

# CALCULATRICE PROGRAMMABLE

## ET-58



*Manuel Utilisateur*

# Calculatrice Programmable ET-58, Manuel Utilisateur

Version 1.1

Décembre 2024

Relatif à la version 201005 de la calculatrice



website: [http://www.breatharian.eu/hw/et58/index\\_en.html](http://www.breatharian.eu/hw/et58/index_en.html)

# Sommaire

		Caractéristiques	11
		Description	13
		Comment utiliser la calculatrice	14
		Différences avec la TI-58/59	16
		Format des nombres	22
		Clavier	23
		Disposition du clavier	24
		Indicateurs à l'écran	26
		Signalement d'erreur	28
		Édition de nombres	30
		Expressions numériques	31
		Adressage indirect	32
		Programmation	35
		Boutons et instructions	39
00...09	<b>0...9</b>	Chiffres de base	39
0A...0F	<b>0A...0F</b>	Chiffres hexadécimaux	39
10...1F	<b>A...F</b>	Labels alphabétiques	40
20	<b>OFF</b>	Éteindre la calculatrice	42
21	<b>2nd</b>	Fonction alternative	43
22	<b>INV</b>	Inverse d'une fonction	43
23	<b>In</b>	Logarithme naturel et exposant	44
24	<b>CE</b>	Effacement erreur	45
25	<b>CLR</b>	Effacer l'affichage	45
26	<b>SBR Ind</b>	Sous-programme avec adresse indirecte	46
27	<b>HIR Ind</b>	Instruction interne indirecte	46
28	<b>log</b>	Logarithme décimal et exposant	47
29	<b>CP</b>	Effacement du programme et du registre T	48
2B	<b>code</b>	Saisie d'un code d'instruction	48
2C	<b>log2</b>	Logarithme binaire et exposant	49
2D	<b>rand</b>	Générateur de nombres aléatoires	50
30	<b>tan</b>	Tangente	51
31	<b>LRN</b>	Mode programmation	52
32	<b>x&lt;-&gt;t</b>	Échange des registres X et T	52
33	<b>x^2</b>	Carré d'un nombre	53
34	<b>√x</b>	Racine carrée d'un nombre	53
35	<b>1/x</b>	Inverse d'un nombre	53
36	<b>Pgm</b>	Sélection d'un programme de bibliothèque	54

37	<b>P-&gt;R</b>	Conversion de coordonnées cartésiennes et polaires	55
38	<b>sin</b>	Sinus	56
39	<b>cos</b>	Cosinus	57
3A	<b>Temp</b>	Température	57
3B	<b>x&lt;&gt;y</b>	Échange des registres X et Y	58
3C	<b>sinh</b>	Sinus hyperbolique	59
3D	<b>cosh</b>	Cosinus hyperbolique	59
3E	<b>tanh</b>	Tangente hyperbolique	60
40	<b>Ind</b>	Adressage indirect	60
41	<b>SST</b>	Avance d'un pas	60
42	<b>STO</b>	Stockage d'un nombre dans un registre	61
43	<b>RCL</b>	Rappel d'un nombre depuis un registre	61
44	<b>SUM</b>	Addition et soustraction d'un nombre dans un registre	62
45	<b>y^x</b>	Puissance et racine	62
46	<b>Ins</b>	Insertion d'un pas vide dans le programme	63
47	<b>CMs</b>	Effacement des registres de données	63
48	<b>Exc</b>	Échange d'un nombre avec un registre	64
49	<b>Prd</b>	Multiplier et diviser dans un registre	64
4A	<b>Bat</b>	Détection de la tension de la batterie	65
4B	<b>x!</b>	Factorielle	65
4C	<b>Inx!</b>	Logarithme naturel d'une factorielle	66
4D	<b>logx!</b>	Logarithme décimal d'une factorielle	66
4E	<b>mod2</b>	Modulo (arrondi inférieur)	67
50	<b> x </b>	Valeur absolue	68
51	<b>BST</b>	Recul d'un pas	68
52	<b>EE</b>	Mode exposant (notation scientifique)	69
53	<b>(</b>	Parenthèse gauche	69
54	<b>)</b>	Parenthèse droite	70
55	<b>:</b>	Division	70
56	<b>Del</b>	Suppression d'un pas du programme	70
57	<b>Eng</b>	Mode exposant (notation ingénieur)	70
58	<b>Fix</b>	Arrondi	71
59	<b>Int</b>	Entier	72
5A	<b>LCD</b>	Réglage du contraste de l'affichage	73
5B	<b>SHL</b>	Décaler vers la gauche	73
5C	<b>SHR</b>	Décaler vers la droite	74
5D	<b>round</b>	Arrondi	74
5E	<b>mod</b>	Modulo (troncature)	75
60	<b>Deg</b>	Degrés	76
61	<b>GTO</b>	Saut	76

62	<b>Pgm Ind</b>	Sélection indirecte d'un programme de bibliothèque	77
63	<b>Exc Ind</b>	Echange indirect avec le contenu d'un registre	78
64	<b>Prd Ind</b>	Multiplication et division indirectes dans un registre	78
65	<b>x</b>	Multiplication	79
66	<b>Pause</b>	Délai d'attente	79
67	<b>x=t</b>	Test d'égalité registres x et t	79
68	<b>Nop</b>	Pas d'opération	81
69	<b>Op</b>	Opérations spéciales	81
6A	<b>REL</b>	Saut relatif	82
6B	<b>Inc Ind</b>	Incrémentation indirecte dans un registre	83
6C	<b>Reg Ind</b>	Opérations indirectes de registre à registre	83
6D	<b>IF Ind</b>	Condition indirecte	84
6E	<b>AND</b>	ET entre deux opérandes binaires	84
70	<b>Rad</b>	Radians	85
71	<b>SBR</b>	Appel sous-programme	86
72	<b>STO Ind</b>	Stockage indirect d'un nombre dans un registre	87
73	<b>RCL Ind</b>	Rappel indirect d'un nombre depuis un registre	87
74	<b>SUM Ind</b>	Addition indirecte d'un nombre dans un registre	87
75	<b>-</b>	Soustraction	88
76	<b>Lbl</b>	Etiquette dans un programme	88
77	<b>x&gt;=t</b>	Test si supérieur ou égal	89
78	<b>Stat</b>	Saisie données statistiques	90
79	<b>Mean</b>	Moyenne statistique	91
7A	<b>IF</b>	Saut conditionnel	92
7E	<b>XOR</b>	OU exclusif entre deux opérandes binaires	95
80	<b>Grad</b>	Grades	95
81	<b>RST</b>	Réinitialisation pointeur de programme	96
82	<b>HIR</b>	Gestion registres internes	96
83	<b>GTO Ind</b>	Saut indirect	100
84	<b>OP Ind</b>	Opération spéciale indirecte	101
85	<b>+</b>	Addition	101
86	<b>StFlg</b>	Positionnement drapeau	101
87	<b>IfFlg</b>	Test drapeau	102
88	<b>DMS</b>	Conversions minutes et secondes / décimal	104
89	<b>pi</b>	Nombre de Ludolf [constante pi]	105
8A	<b>Reg</b>	Opérations de registre à registre	105
8B	<b>HEX</b>	Mode hexadécimal	107
8C	<b>BIN</b>	Mode binaire	108
8D	<b>OCT</b>	Mode octal	108
8E	<b>OR</b>	OU entre deux opérandes binaires	109

91	<b>R/S</b>	Démarrer et arrêter le programme	110
92	<b>RTN</b>	Retour de sous-programme	110
93	.	Séparateur décimal	110
94	<b>+/-</b>	Changement de signe	111
95	<b>=</b>	Réalisation du calcul	111
97	<b>Dsz</b>	Boucle de programme	112
9A	<b>phi</b>	Nombre d'or [constante phi]	114
9B	<b>DEC</b>	Mode décimal	114
9C	<b>Inc</b>	Incrémentation/décrémentation dans un registre	115
9D	<b>NOT</b>	Inversion de bits	116
9E	<b>%</b>	Pourcentage	116
		Opérations spéciales Op	118
<b>Op 00</b>		Efface les registres d'impression 1 à 4	118
<b>Op 01..04</b>		Stocke dans registre d'impression 1 à 4	118
<b>Op 09</b>		Charge un programme de bibliothèque	119
<b>Op 0A</b>		Affichage registres d'impression 1 et 2 [ligne 1]	119
<b>Op 0B</b>		Affichage registre d'impression 1 et registre X	120
<b>Op 0C</b>		Affichage registre d'impression 1 et registre X	121
<b>Op 0D</b>		Affichage registres d'impression 3 et 4 [ligne 2]	121
<b>Op 0E</b>		Affichage registre d'impression 3 et registre X	121
<b>Op 0F</b>		Affichage registre d'impression 3 et registre X	122
<b>Op 10</b>		Test de signe	122
<b>Op 11</b>		Variance	122
<b>Op 12</b>		Coefficients de régression linéaire	123
<b>Op 13</b>		Coefficient de corrélation	124
<b>Op 14</b>		Régression linéaire de Y sur X	125
<b>Op 15</b>		Régression linéaire de X sur Y	125
<b>Op 16, 17</b>		Gestion de la mémoire [inopérant]	126
<b>Op 18</b>		Lève le drapeau 7 si pas d'erreur	126
<b>Op 19</b>		Lève le drapeau 7 si erreur	126
<b>Op 1A</b>		Affichage registres impression 1 et 2 [à l'arrêt]	126
<b>Op 1B</b>		Affichage registre d'impression 1 et registre X	127
<b>Op 1C</b>		Affichage registre d'impression 1 et registre X	127
<b>Op 1D</b>		Active le mode d'affichage 'Indicateurs' sur 1ère ligne	127
<b>Op 1E</b>		Active le mode d'affichage 'Registre T' sur 1ère ligne	128

<b>Op 1F</b>	Active le mode d'affichage 'Texte' sur 1ère ligne	128
<b>Op 20...2F</b>	Incrémentation du registre R00 à R15	128
<b>Op 30...3F</b>	Décrémentation du registre R00 à R15	128
<b>Op 40</b>	Affiche le code de la touche	129
<b>Op 41</b>	Test d'appui sur une touche	129
<b>Op 42</b>	Affiche un caractère à l'écran	130
<b>Op 43</b>	Chargement de la police	132
<b>Op 44</b>	Pause paramétrée	136
<b>Op 45</b>	Afficher le pointeur à gauche de la 1ère ligne	136
<b>Op 46</b>	Afficher le pointeur à gauche de la 2ème ligne	137
<b>Op 47</b>	Afficher le texte avec le pointeur à gauche à l'arrêt	137
<b>Op 48</b>	Afficher le pointeur à droite de la 1ère ligne	138
<b>Op 49</b>	Afficher le pointeur à droite de la 2ème ligne	139
<b>Op 4A</b>	Afficher le texte avec le pointeur à droite à l'arrêt	139
<b>Op 4B</b>	Affichage d'un graphique à barres lors de l'exécution	140
<b>Op 4C</b>	Affichage d'un graphique à barres avec texte	141
<b>Op 4D</b>	Paramètres de pixels	143
<b>Op 4E</b>	Effacer un pixel	143
<b>Op 4F</b>	Commutation de pixels	143
<b>Op 50</b>	Trouver le plus grand dénominateur commun	144
<b>Op 51</b>	Rappel du registre du générateur aléatoire	144
<b>Op 52</b>	Chargement du registre du générateur aléatoire	144
<b>Op 53</b>	Charger le texte prédéfini	145
<b>Op 54</b>	Ajouter un nombre au texte	146
<b>Op 55</b>	Initialiser la pile de nombres complexes ou de fractions	147
<b>Op 56</b>	Affiche le nombre de nombres complexes dans la pile	148
<b>Op 57</b>	Insérer un nombre dans la pile de nombres complexes	148
<b>Op 58</b>	Rappel un nombre de la pile de nombres complexes	148
<b>Op 59</b>	Annule un nombre dans pile de nombres complexes	149
<b>Op 5A</b>	Échange 2 nombres dans pile de nombres complexes	149
<b>Op 5B</b>	Duplique un nombre dans pile de nombres complexes	149
<b>Op 5C</b>	Addition de deux nombres complexes X+Y	149
<b>Op 5D</b>	Différence de deux nombres complexes X-Y	149

<b>Op 5E</b>	Produit de deux nombres complexes $X*Y$	150
<b>Op 5F</b>	Quotient de deux nombres complexes $X/Y$	150
<b>Op 60</b>	Exponentiation de deux nombres complexes $X^Y$	150
<b>Op 61</b>	Racine de deux nombres complexes $X^{(1/Y)}$	150
<b>Op 62</b>	Logarithme de deux nombres complexes $\log Y(X)$	150
<b>Op 63</b>	Le carré du nombre complexe $X^2$	151
<b>Op 64</b>	La racine carrée du nombre complexe $\sqrt{X}$	151
<b>Op 65</b>	L'inverse du nombre complexe $1/X$	151
<b>Op 66</b>	L'exposant naturel du nombre complexe $e^X$	151
<b>Op 67</b>	Le logarithme naturel du nombre complexe $\ln(X)$	151
<b>Op 68</b>	Sinus du nombre complexe $\sin(X)$	151
<b>Op 69</b>	Cosinus d'un nombre complexe $\cos(X)$	151
<b>Op 6A</b>	Tangente du nombre complexe $\tan(X)$	152
<b>Op 6B</b>	Arcsinus du nombre complexe $\text{asin}(X)$	152
<b>Op 6C</b>	Arccosinus du nombre complexe $\text{acos}(X)$	152
<b>Op 6D</b>	Arctangente du nombre complexe $\text{atan}(X)$	152
<b>Op 6E</b>	Convertir un nombre complexe en un nombre polaire	152
<b>Op 6F</b>	Convertir nombres polaires en nombres complexes	152
<b>Op 70</b>	Trouver le passage par zéro de $A'$	153
<b>Op 71</b>	Intégrale de Simpson de la fonction $A'$	153
<b>Op 72</b>	Convertir un angle de l'unité d'angle en cours en radians	154
<b>Op 73</b>	Convertir un angle en radians à l'unité d'angle en cours	154
<b>Op 74</b>	Distribution de probabilité normale $Z(x)$	154
<b>Op 75</b>	Distribution gaussienne complémentaire $Q(x)$	154
<b>Op 76</b>	Distribution normale cumulative de $P(x)$	154
<b>Op 77</b>	Maximum	155
<b>Op 78</b>	Minimum	155
<b>Op 79</b>	Réinitialiser tout les registres HIR	155
<b>Op 7A</b>	Conversion décimale en fraction	155
<b>Op 7B</b>	Conversion d'une fraction en un nombre décimal	155
<b>Op 7C</b>	La somme de la fraction $X+Y$	155
<b>Op 7D</b>	Différence de fraction $X-Y$	156
<b>Op 7E</b>	Produit de deux fractions de $X*Y$	156



<b>Op 7F</b>	Quotient de deux fractions $X/Y$	156
<b>Op 80</b>	Temps de pause de 10 ms	156
<b>Op 81</b>	Temps de pause de 100 ms	156
<b>Op 82</b>	Suppression des chiffres masqués	157
<b>Op 83</b>	Début du chronométrage	157
<b>Op 84</b>	Détection du temps écoulé	157
<b>Op 85</b>	Détermination du nombre de registres de données	157
<b>Op 86</b>	Détermination de l'état des commutateurs utilisateur	158
<b>Op 87</b>	Calcul de la somme de contrôle ROM	158
<b>Op 88</b>	Réglage du délai d'extinction de la calculatrice	158
<b>Op 89</b>	Détection du délai d'extinction de la calculatrice	158
<b>Op 8A</b>	Affichage de la version du firmware de la calculatrice	158
<b>Op 8B</b>	Réinitialiser la calculatrice	159
	Table des caractères	160
	Programmes de la bibliothèque	162
<b>ML-01</b>	Diagnostic	162
<b>ML-02</b>	Matrices: Inversion, déterminant, équations	164
<b>ML-03</b>	Addition et multiplication de matrices	168
<b>ML-04</b>	Arithmétique complexe	172
<b>ML-05</b>	Fonctions complexes	175
<b>ML-06</b>	Fonctions trigonométriques complexes	178
<b>ML-07</b>	Calcul d'un polynôme	181
<b>ML-08</b>	Zéros d'une fonction	182
<b>ML-09</b>	Approximation de Simpson (continue)	184
<b>ML-10</b>	Approximation de Simpson (discrète)	187
<b>ML-11</b>	Résolution d'un triangle (1)	188
<b>ML-12</b>	Résolution d'un triangle (2)	191
<b>ML-13</b>	Calcul d'arcs de cercles	194
<b>ML-14</b>	Distribution normale	196
<b>ML-15</b>	Génération de nombres aléatoires	198
<b>ML-16</b>	Combinaisons, permutations, factorielles	199
<b>ML-17</b>	Moyennes mobiles	202
<b>ML-18</b>	Intérêts composés (Méthode US)	203

<b>ML-19</b>	Annuités	206
<b>ML-20</b>	Nombre de jour entre deux dates, jour de la semaine	210
<b>ML-21</b>	Jeu du nombre mystérieux HILO	212
<b>ML-22</b>	Vérification de relevés bancaires	213
<b>ML-23</b>	Opérations sexagésimales	215
<b>ML-24</b>	Conversions (1)	217
<b>ML-25</b>	Conversions (2)	217
<b>ML-26</b>	Arithmétique des fractions	218
<b>ML-27</b>	Générateur astable avec circuit 555	222
<b>ML-28</b>	(EE-07) Conversion de ratios	225
<b>ML-29</b>	(EE-11) Diagramme de réactance	226
<b>ML-30</b>	(EE-12) Conversion d'impédance série/parallèle	227
<b>ML-31</b>	(EE-13) Filtres actifs	229
<b>ML-32</b>	(EE-14) Filtres passifs	233
<b>ML-33</b>	(EE-15) Convolution du signal	235
<b>ML-34</b>	(EE-17) Transformation de Fourier discrète	238
<b>ML-35</b>	Loi d'Ohm	240
<b>ML-36</b>	Connexions série et parallèle	242
<b>ML-37</b>	(AV-23) Changement de fuseau horaire	242
<b>ML-38</b>	(MU-06) Tri	246
<b>ML-39</b>	(MU-09) Décomposition en facteurs premiers	248
<b>ML-40</b>	(MU-21) Arithmétique avec variables	249
<b>ML-41</b>	(MU-14) Interpolation	250
<b>ML-42</b>	(MU-16) Minimax	252
<b>ML-43</b>	Somme de contrôle	254
<b>ML-44</b>	(LE-09) Jeu Mastermind	258
<b>ML-45</b>	(LE-12) Jeu Acey-Deucey	259
<b>ML-46</b>	(LE-13) Jeu de craps	261
<b>ML-47</b>	(LE-14) Jeu Atterrir sur Mars	263
<b>ML-48</b>	Jeu de nim	265
<b>ML-49</b>	Mesure du temps de réaction	266
<b>ML-50</b>	(LE-20) Jeu de bataille navale	266

# 1. Caractéristiques

- Processeur ATmega328P (4 MHz, 32 KB ROM, 2 KB RAM)
- Tension d'alimentation 2,6 V (depuis la batterie) jusqu'à 5,5 V (depuis le connecteur USB).
- Précision des calculs 19 chiffres
- Affichage jusqu'à 14 chiffres significatifs
- Exposant 4 chiffres, plage +- 9863
- 1000 pas de programme utilisateur en EEPROM (conserve le programme même sans batterie)
- 110 registres de données
- 16 registres de contrôle HIR
- Écran LCD à deux lignes (2 x 16 caractères alphanumériques)
- 45 touches
- Code de calculatrice entièrement écrit en assembleur AVR
- Bibliothèque intégrée de 50 programmes avec une longueur totale de près de 10 000 pas de programme
- Fonctions exponentielles et logarithmiques
- Fonctions trigonométriques
- Fonctions hyperboliques
- Factorielle de nombres décimaux et grands
- Générateur de nombres aléatoires
- Accès indexé aux variables
- Paramètres indirects des fonctions
- Mode d'affichage scientifique et technique avec exposant
- Mode d'affichage HEX, OCT a BIN, y compris les décimales
- Opérations binaires AND, OR, XOR, NOT, décalages
- Affichage pseudographique de graphiques et d'indicateurs
- Affichage programmatique de texte
- Saisie dynamique au clavier pendant l'exécution du programme
- Fonctions statistiques et régression linéaire
- Adressage absolu, étiquettes, sauts relatifs
- Calculs avec des matrices
- Nombres complexes
- Fractions
- Énumération de polynômes
- Recherche numérique de racines de fonctions
- Calcul numérique d'intégrales
- Interpolation et approximation

- Calculs de triangles
- Conversions d'unités
- Secteurs circulaires
- Combinaisons, permutations
- Moyenne flottante
- Intérêts et remboursements
- Transferts horaires, fuseaux horaires
- Intervalle entre les dates, jour de la semaine
- Jeux (Hi-Lo, Codebreaker, Acey-Deucey, Lander et plus)
- Générateur astable avec 555
- Réactance des condensateurs et des inducteurs
- Connexion série et parallèle des composants
- Calculs de filtres actifs et passifs
- Convolution du signal
- Transformée de Fourier discrète
- Loi d'Ohm
- Tri des nombres et médiane
- Recherche de facteurs premiers
- Recherche du minimum et du maximum de la fonction
- Mesure du temps de réaction

## 2. Description

La calculatrice ET-58 est destinée principalement aux bricoleurs et à ceux qui s'intéressent à la technologie informatique rétro. Conceptuellement, elle est basée sur la calculatrice populaire TI-58/59, développée en 1978 par la société Texas Instruments. Elle étend ses fonctionnalités, tout en essayant de préserver autant que possible les fonctionnalités de la calculatrice d'origine.

La calculatrice ET-58 est destinée aux bricoleurs, sous forme de kit. Elle met l'accent sur la conception la moins chère possible et un assemblage facile du kit, elle utilise donc une construction la plus simple possible avec de simples boutons et avec un minimum de composants CMS.

### 3. Comment utiliser la calculatrice

La calculatrice ET-58 est équipée d'un écran LCD alphanumérique à 2 rangées, de 45 micro-interrupteurs, d'un processeur et d'une batterie.

La calculatrice est mise sous tension en appuyant sur le bouton **CLR** (fonction alternative **OFF**). Elle s'éteint soit en appuyant sur **OFF** (c'est-à-dire en appuyant sur **2nd CLR**) soit automatiquement après un temps d'inactivité de la calculatrice sélectionné (réglé par défaut à 30 secondes - il peut être modifié par **Op 88**). Si la batterie n'est pas retirée pendant plus de quelques minutes (et l'alimentation externe déconnectée), le contenu des registres de mémoire, le nombre affiché et les opérations initiées (c'est-à-dire le contenu de la mémoire RAM) seront conservés inchangés. Pour la calculatrice sans batterie insérée, n'appuyez pas sur le bouton **CLR** pour conserver le contenu de la RAM après la mise sous tension, sinon la calculatrice se réinitialisera après la mise sous tension et oubliera le contenu de la RAM. La perte du contenu de la mémoire n'affecte pas le programme en ROM (bibliothèque) et EEPROM (programme utilisateur). Ils restent inchangés même sans batterie.

Si la pile est retirée alors que la calculatrice n'est pas éteinte, le contenu de la RAM peut être immédiatement perdu. De cette façon, vous pouvez réinitialiser la calculatrice à son état par défaut si elle présente un comportement incorrect ou si vous souhaitez réinitialiser la calculatrice aux paramètres par défaut. Il est également possible de rétablir les paramètres par défaut de la calculatrice par le biais de la sous-routine **CE** du programme de bibliothèque 1 (pour ce faire, saisissez **Pgm 0 1 SBR CE RST**).

Le contenu du programme utilisateur et de la bibliothèque utilisateur téléchargée ne sera pas perdu même après un retrait prolongé de la pile.

Attention - Ne retirez pas la pile ou ne débranchez pas l'alimentation pendant les opérations d'écriture de programme ou pendant le chargement de la bibliothèque en mémoire. Cela pourrait endommager le contenu de la mémoire.

Le contraste de l'écran LCD dépend de la tension d'alimentation. Utilisez le bouton **LCD** pour régler le contraste de l'écran (vous pouvez l'activer en appuyant sur les boutons **2nd 2nd LRN**). Après avoir appuyé sur le

bouton, saisissez le chiffre **0** à **9** dans la calculatrice, qui représente le contraste souhaité de l'écran LCD. Le chiffre 0 signifie le contraste minimum (les caractères sur l'écran sont gris ou même à peine visibles), le chiffre 9 signifie le contraste maximum (il y a des rectangles sombres sous les caractères sur l'écran). En modifiant le contraste de l'écran, vous pouvez sortir de la plage visible, lorsque vous ne reconnaissez plus le texte sur l'écran. Même dans ce cas, essayez de modifier le contraste avec le bouton **LCD**. Si aucun caractère n'est visible sur l'écran (la calculatrice semble éteinte), essayez de régler un contraste d'affichage plus élevé à partir du clavier en utilisant un nombre plus élevé. Alternativement, appuyez d'abord sur **CLR**, la calculatrice peut être éteinte. Inversement, s'il y a des rectangles noirs sur l'écran, définissez une valeur inférieure du contraste de l'écran. Si vous ne savez pas dans quel état se trouvait la calculatrice (elle est peut-être maintenant en mode programmation), retirez la pile, appuyez sur **CLR** pour réinitialiser la calculatrice. Insérez ensuite la pile, essayez d'allumer la calculatrice avec le bouton **CLR** et réglez le contraste de l'écran. Si cela échoue, essayez une nouvelle pile ou une alimentation externe, peut-être que la pile est faible.

Après avoir réinitialisé la calculatrice (la première fois qu'elle est allumée après avoir inséré la pile), la calculatrice affichera son nom pendant 2 secondes, ainsi qu'un code à 6 chiffres représentant la date de la version du micrologiciel de la calculatrice. Par exemple, « ET-58 201005 » signifie date du micrologiciel (build) 5/10/2020. Pour afficher à nouveau la version du micrologiciel, retirez et réinsérez la pile dans la calculatrice sans éteindre la calculatrice. Cela réinitialisera la calculatrice. Si vous retirez la pile de la calculatrice alors qu'elle est éteinte, la calculatrice peut conserver les données pendant une heure sans effectuer de réinitialisation après la mise sous tension. Dans ce cas, appuyez sur le bouton **CLR** et la réinitialisation se produira même lorsqu'elle est éteinte. Vous pouvez également utiliser l'**Op 8A** pour afficher la version du micrologiciel.

Chaque fois que la calculatrice est allumée, elle vérifie l'intégrité de la ROM interne à l'aide d'une somme de contrôle 16 bits (CRC-XModem). Si une erreur de mémoire interne se produit, la calculatrice affiche un avertissement « Erreur CRC » pendant 2 secondes. La calculatrice peut toujours être utilisée, mais son processeur est manifestement défectueux et peut présenter une activité imprévisible.

## 4. Différences avec les TI-58/59

Bien que le logiciel de la calculatrice ET-58 s'efforce d'assurer une compatibilité maximale avec la série TI-58/59 d'origine, une compatibilité totale ne peut être garantie et certains programmes peuvent alors devoir être modifiés lors de l'importation.

### **Précision accrue**

Les calculatrices TI-58/59 d'origine fonctionnaient avec des nombres en code BCD (1 octet = 2 chiffres décimaux) et avec une précision interne de 13 chiffres. La taille du nombre était de 10 octets. Bien que l'ET-58 utilise également des nombres de 10 octets, elle utilise un code binaire, ce qui lui permet d'atteindre une plus grande précision avec une mantisse de 19 chiffres et des calculs plus rapides. Une plus grande précision n'est généralement pas un problème, mais l'utilisation d'un code binaire au lieu d'un code BCD peut entraîner une perte de précision pour les nombres avec un nombre fini de chiffres. La calculatrice essaie de corriger les déviations, mais elles peuvent parfois apparaître d'elles-mêmes et il peut être nécessaire d'en tenir compte.

Un exemple typique est celui des entiers utilisés comme compteur dans les boucles. Après des milliers de cycles d'addition/soustraction de la valeur 1, un petit écart par rapport au nombre entier peut se produire, qui peut ne pas apparaître sur l'écran, mais il peut se produire lors de la comparaison du nombre à une correspondance. La calculatrice traite cet écart de sorte que la fonction de boucle (DSZ, DJNZ, DJZ) arrondit le nombre à un entier après chaque décrémentation.

Bien que la calculatrice prenne en compte une certaine tolérance pour l'écart, il peut être nécessaire de prendre en compte un écart croissant par rapport à la valeur exacte supposée pour les calculs répétés et d'utiliser l'arrondi à un entier ou de comparer dans l'intervalle avec tolérance avant de comparer à une correspondance.

De plus, la précision différente se prouve dans le générateur de nombres aléatoires. Le générateur de nombres aléatoires TI-58/59 d'origine effectue des opérations d'addition et de multiplication et déplace les chiffres cachés vers des positions visibles. Cela se reflète dans le fait que le calcul du nombre aléatoire de l'ET-58 s'écarte complètement du générateur aléatoire de la calculatrice TI-58/59 d'origine après quelques étapes. Habituellement, ce fait ne devrait pas avoir d'importance. De plus, dans les programmes



ET-58, le générateur aléatoire d'origine n'est plus utilisé, la calculatrice dispose d'un générateur aléatoire interne avec une distribution aléatoire nettement meilleure.

## Code hexadécimal

Le code du programme est stocké dans le code BCD de la calculatrice TI-58/59 d'origine. Chaque octet est divisé en moitiés contenant 2 chiffres d'une valeur de 0 à 9. Les deux chiffres représentent ainsi la valeur décimale de 0 à 99. En revanche, le code de la calculatrice ET-58 est stocké en mémoire en code hexadécimal. Il est à nouveau exprimé à l'aide de 2 chiffres, mais dans la plage de 0 à 15. Les chiffres d'une valeur de 10 à 15 sont exprimés par les lettres A à F. La notation hexadécimale prend en charge à la fois la signification d'origine, c'est-à-dire l'écriture de la valeur 00 à 99 (soit 100 valeurs) à l'aide de 2 chiffres de 0 à 9, ainsi qu'une signification hexadécimale avec une valeur de 00 à FF (soit 256 valeurs). L'interprétation du code dépend de la signification de l'octet.

Les codes d'origine des boutons sont conservés. Par ex. le bouton **CLR** porte ici aussi le code 25, comme sur la TI-58/59 d'origine, qu'il s'agisse d'une notation binaire ou hexadécimale. Les caractères hexadécimaux étendent le jeu de codes par des codes avec des caractères A à F.

L'adresse absolue et le numéro de registre sont exprimés par 2 chiffres décimaux de 0 à 9, comme sur la TI-58/59 d'origine, et ont une valeur de 0 à 99.



Dans certains cas particuliers, le code de programme a la signification de 2 chiffres hexadécimaux. C'est par exemple le cas du code d'instruction HIR. Le premier chiffre de 0 à F représente le code d'opération, le deuxième chiffre de 0 à F représente le numéro de registre HIR de 0 à 15.

## Précision boucles

La boucle DSZ de la calculatrice d'origine décrémente la valeur du registre de 1, et si elle n'atteint pas zéro, elle revient en arrière. Comme mentionné ci-dessus, en raison de l'expression binaire du nombre, la valeur du registre de l'ET-58 peut s'écarter légèrement du nombre entier, ce qui entraînerait une imprécision de la boucle. La calculatrice traite cela en arrondissant le résultat à un nombre entier après chaque décrémentation. Cela peut se produire si le programme TI-58/59 d'origine envisage l'utilisation de nombres décimaux.

Il en va de même pour les instructions DJNZ et DJZ (groupe d'instructions HIR), dont la calculatrice d'origine ne dispose cependant pas.

## Répétition des calculs

La calculatrice ET-58 répète la dernière opération arithmétique saisie après avoir appuyé sur la touche . Certains programmes de la TI-58/59 d'origine ne comptent pas sur la répétition des opérations, ils utilisent la touche  plusieurs fois et cela peut entraîner un calcul incorrect.




## Vitesse accrue


La calculatrice ET-58 utilise un processeur plus moderne et plus puissant, avec une fréquence d'horloge plus élevée. Par conséquent, les calculs sont sensiblement plus rapides qu'avec la calculatrice d'origine. Cette caractéristique ne pose généralement pas de problème. Elle peut se produire dans des programmes qui supposent la vitesse de la calculatrice d'origine, comme un programme de test de perception. Mais il existe un nombre négligeable de programmes de ce type.

## Registres HIR

La calculatrice TI-58/59 d'origine possède des instructions HIR internes spéciales, utilisées par le code interne et non publiées dans le manuel d'utilisation. Les instructions HIR utilisaient à l'origine les registres d'opérations arithmétiques 0 à 9. Pour l'ET-58, les instructions HIR sont validées comme des instructions valides et sont même utilisées par les bibliothèques comme registres de travail principaux. Contrairement à la calculatrice d'origine, elles ne sont pas partagées avec l'unité arithmétique. Il s'agit de registres à part entière, non réécrits par d'autres opérations.

## Arrondi Fix

Pour la calculatrice TI-58/59 d'origine, la touche  permet de régler le nombre de décimales affichées entre 0 et 8. Une valeur de 9 désactive l'arrondi, le nombre est affiché avec le nombre maximum de décimales.   a la même signification.

La calculatrice ET-58 permet d'afficher des nombres jusqu'à 13 décimales. Par conséquent, le paramètre de l'instruction  est étendu aux valeurs de 0 à 9 et de A à D, représentant l'arrondi de 0 à 13 décimales. Le code

**Fix** **OF** est utilisé pour désactiver l'arrondi, tout comme la commande **INV** **Fix**.

Pour les programmes plus anciens, **Fix** **9** est souvent utilisé pour désactiver l'arrondi. Dans de tels cas, le code doit être modifié en **INV** **Fix** ou **Fix** **OF**.

## Indication d'erreur

Sur la calculatrice TI-58/59 d'origine, le clignotement de l'affichage est parfois utilisé pour indiquer une erreur de programmation. La séquence **0** **1/X** est utilisé à cet effet. Ce code fonctionnera sans changement pour l'ET-58.

Le deuxième cas, parfois utilisé, est la séquence **+** **=**. Pour la calculatrice d'origine, cette séquence assure que les données affichées sur l'écran clignotent. Pour ET-58, cette séquence n'active pas d'indication d'erreur. L'indication peut être remplacée soit par la séquence **X/T** **0** **1/X** **X/T** soit en utilisant **STF** **OF**, qui active le drapeau 15, qui est aussi l'état de l'indication d'erreur.

## Fonction INV

La calculatrice TI-58/59 d'origine utilise parfois le préfixe **INV** (fonction inverse) sur plusieurs instructions. La calculatrice suppose que les instructions qui n'utilisent pas ce préfixe ne modifient pas son état. Avec la calculatrice ET-58, de nombreuses fonctions sont étendues par une fonction inverse ou alternative, de sorte que la plupart des commandes traitent le code **INV** et ne l'enregistrent pas.

L'utilisation la plus courante de la fonction **INV** concerne les étiquettes.

Si une première étiquette est suivie de la commande **INV** elle-même, suivie d'une deuxième étiquette et d'une première commande. Dans ce cas, l'exécution du programme à partir de la première étiquette invoque la fonction inverse de la commande, l'exécution du programme à partir de la deuxième étiquette invoque la fonction non inversée. Dans ce cas, l'état du bouton **INV** est également conservé derrière l'instruction **LBL** de la première étiquette sur la calculatrice ET-58.

## Séquence 2nd INV

Pour les TI-58/59 d'origine, l'ordre d'appui sur les touches **2nd** et **INV** n'avait pas d'importance. Dans les deux cas, une fonction alternative inverse était exécutée. Lors de l'écriture dans le programme, dans un cas, le code **INV** et le code alternatif de la touche étaient écrits, dans l'autre cas, le code **2nd INV** (code 26) et le code normal du bouton étaient écrits - le changement vers le code alternatif était effectué jusqu'à l'exécution du programme.

Cette fonction ne peut pas être utilisée avec la calculatrice ET-58 et il est nécessaire de maintenir l'ordre d'appui sur les boutons **INV 2nd ...**. Le bouton **INV** définit d'abord le préfixe de la fonction inverse et le bouton **2nd ...** écrit un code de bouton alternatif dans le programme. Lorsqu'il est enfoncé dans la direction opposée, **2nd INV**, une fonction alternative du bouton **INV** est exécutée, c'est-à-dire la fonction **HIR**.

## Table des caractères

L'ET-58 utilise une table de caractères différente pour l'impression et l'affichage que la TI-58/59. Les caractères sont saisis sous forme de 2 chiffres d'un nombre décimal compris entre 00 et 99 et (sauf de petites différences) ils correspondent à des caractères ASCII diminués de 32. Lors de l'importation de programmes, il est nécessaire de modifier le contenu des registres **Op 01** à **Op 04**. Pour plus de détails, voir le chapitre Table de caractères.

## Fonction HIR 20

La fonction **HIR 20** était utilisée pour brancher le programme interne du microcode sur les TI-58/59 d'origine. Elle permettait, selon un registre interne prédéfini, d'effectuer soit un saut relatif, soit la terminaison d'une fonction. Sur ET-58 elle perd sa signification et est donc utilisée à d'autres fins (arrondi du registre HIR).

## Organisation de la mémoire

Dans les TI-58/59 d'origine, la RAM était divisée entre la mémoire programme et la mémoire registre de données. La partition pouvait être modifiée par l'opération **Op 17**. La partition ne peut pas être définie avec

l'ET-58 et correspond toujours à un maximum de 1000 pas de programme (dans la mémoire EEPROM) et 100 registres de données ou plus (dans la mémoire RAM).

## 5. Format des nombres

L'utilisateur ne rencontre généralement pas le format de nombre interne. Il peut être rencontré si le contenu des registres est enregistré sur la carte SD ou si la mantisse est affichée avec la commande **INV DEC**.

Les nombres dans la calculatrice sont stockés dans 10 octets de mémoire. Les deux premiers octets contiennent l'exposant, l'octet supérieur étant à l'adresse inférieure. L'exposant est un entier non signé, avec un biais de 0x8000 (32768) et avec une plage de valeurs valides de 0x0001 à 0xFFFE. L'exposant avec une valeur de 0x0000 est un cas particulier et représente un nombre nul (le contenu de la mantisse n'a pas d'importance). Le deuxième cas particulier est la valeur 0xFFFF, représentant l'infini (dépassement de capacité). Après conversion en nombre décimal, l'exposant a une valeur comprise entre -9863 et +9863.

8 octets sont réservés à la mantisse. La mantisse est un nombre binaire, dont l'octet le plus significatif se trouve à l'adresse mémoire inférieure (c'est-à-dire l'offset 2). Le bit le plus élevé de la mantisse (c'est-à-dire le bit 7 du premier octet à l'offset 2) a toujours la valeur 1 pour un nombre non nul et n'est donc pas exprimé dans le nombre, il est masqué. Sa position sert à indiquer le signe du nombre (1 signifie nombre négatif).

La précision de la mantisse est de 19,27 chiffres décimaux. L'écran affiche un maximum de 14 chiffres, soit 5 chiffres cachés et servent à maintenir la précision des calculs de la calculatrice. Le test trigonométrique populaire peut être utilisé pour tester la précision de la calculatrice :

**9 sin cos tan INV tan INV cos INV tan**

Si le calcul est correct, le résultat devrait à nouveau être le nombre 9. Le calcul perd rapidement en précision et un écart se produit généralement dans les calculatrices. Pour l'ET-58, l'écart reste dans les chiffres cachés et le résultat affiché sera à nouveau le chiffre 9.

Plus d'informations sur la précision de la calculatrice :  
<http://www.datamath.org/Forensics.htm>

## 6. Le clavier

La calculatrice peut être utilisée soit en mode direct, où les codes des touches sont exécutés immédiatement, soit en mode programmation, où les codes des touches sont uniquement écrits dans le programme mais ne sont pas exécutés.

La calculatrice est contrôlée par un ensemble de 45 boutons, disposés en 9 rangées et 5 colonnes. Les rangées sont numérotées de haut en bas, dans l'ordre de 1 à 9. Les colonnes sont numérotées de gauche à droite, avec des numéros de 1 à 5.

Après avoir appuyé sur le bouton **2nd**, la fonction alternative du bouton est utilisée, indiquée par le numéro de colonne 6 à 10 (le numéro 10 est remplacé par le chiffre 0 dans le code). Après la deuxième pression sur le bouton **2nd** (c'est-à-dire **3rd**), la deuxième fonction alternative est utilisée. Il est indiqué par les colonnes A à E.

Lors de l'écriture d'un programme dans la mémoire (en utilisant la touche LRN), le code de touche est écrit dans le programme sous forme de paire de chiffres, où le premier chiffre représente la rangée de la touche (généralement de 1 à 9) et le deuxième chiffre la colonne de la touche (généralement de 1 à 5 pour la fonction de base ou de 6 à E pour une fonction alternative).

Les codes des touches numériques de **0** à **9** ne sont pas stockés dans le programme à l'aide des coordonnées de la touche, mais sous forme de valeur décimale de 00 à 09.

En plus des codes de touches répertoriés, il peut y avoir des codes de fonction de touche secondaires et des commandes spéciales dans le programme qui utilisent des codes qui ne sont pas directement accessibles depuis le clavier. Par exemple, la séquence **INV SBR** est stockée dans le programme sous forme de commande **RTN** avec le code 92.

**Remarque** : dans le texte du manuel, les noms des boutons sont donnés sans aucun deuxième préfixe, qui peut être nécessaire pour appeler la fonction du bouton. Par exemple, le code de la touche **rand** (nombre aléatoire) est appelé en appuyant sur les boutons **2nd 2nd CE**.

## 7. Disposition du clavier

Pour chaque bouton, la signification de base est donnée sur la 1ère ligne, la première signification alternative sur la 2ème ligne (après avoir appuyé sur le bouton **2nd**) et la deuxième signification alternative sur la 3ème ligne (après avoir appuyé deux fois sur le bouton **2nd**, c'est-à-dire **3rd**).

[11] <b>A</b>	[12] <b>B</b>	[13] <b>C</b>	[14] <b>D</b>	[15] <b>E</b>
[16] <b>A'</b>	[17] <b>B'</b>	[18] <b>C'</b>	[19] <b>D'</b>	[10] <b>E'</b>
[1A] <b>A''</b>	[1B] <b>B''</b>	[1C] <b>C''</b>	[1D] <b>D''</b>	[1E] <b>E''</b>
[21] <b>2nd</b>	[22] <b>INV</b>	[23] <b>In x</b>	[24] <b>CE</b>	[25] <b>CLR</b>
[ ] <b>3rd</b>	[82] <b>HIR</b>	[28] <b>log x</b>	[29] <b>CP</b>	[20] <b>OFF</b>
[ ]	[2B] <b>code</b>	[2C] <b>log2</b>	[2D] <b>rand</b>	[2E]
[31] <b>LRN</b>	[32] <b>x&lt;&gt;t</b>	[33] <b>x^2</b>	[34] <b>Vx</b>	[35] <b>1/x</b>
[36] <b>Pgm</b>	[37] <b>P-&gt;R</b>	[38] <b>sin</b>	[39] <b>cos</b>	[30] <b>tan</b>
[3A] <b>Temp</b>	[3B] <b>x&lt;&gt;y</b>	[3C] <b>sinh</b>	[3D] <b>cosh</b>	[3E] <b>tanh</b>
[41] <b>SST</b>	[42] <b>STO</b>	[43] <b>RCL</b>	[44] <b>SUM</b>	[45] <b>y^x</b>
[46] <b>Ins</b>	[47] <b>CMs</b>	[48] <b>Exc</b>	[49] <b>Prd</b>	[40] <b>Ind</b>
[4A] <b>Bat</b>	[4B] <b>x!</b>	[4C] <b>In x!</b>	[4D] <b>log x!</b>	[4E] <b>mod2</b>
[51] <b>BST</b>	[52] <b>EE</b>	[53] <b>(</b>	[54] <b>)</b>	[55] <b>:</b>
[56] <b>Del</b>	[57] <b>Eng</b>	[58] <b>Fix</b>	[59] <b>Int</b>	[50] <b>Ixl</b>
[5A] <b>LCD</b>	[5B] <b>SHL</b>	[5C] <b>SHR</b>	[5D] <b>round</b>	[5E] <b>mod</b>
[61] <b>GTO</b>	[07] <b>7</b>	[08] <b>8</b>	[09] <b>9</b>	[65] <b>x</b>
[66] <b>Pause</b>	[67] <b>x=t</b>	[68] <b>Nop</b>	[69] <b>Op</b>	[60] <b>Deg</b>
[6A] <b>REL</b>	[0D] <b>0D</b>	[0E] <b>0E</b>	[0F] <b>0F</b>	[6E] <b>AND</b>
[71] <b>SBR</b>	[04] <b>4</b>	[05] <b>5</b>	[06] <b>6</b>	[75] <b>-</b>
[76] <b>Lbl</b>	[77] <b>x&gt;=t</b>	[78] <b>Stat</b>	[79] <b>Mean</b>	[70] <b>Rad</b>
[7A] <b>IF</b>	[0A] <b>0A</b>	[0B] <b>0B</b>	[0C] <b>0C</b>	[7E] <b>XOR</b>
[81] <b>RST</b>	[01] <b>1</b>	[02] <b>2</b>	[03] <b>3</b>	[85] <b>+</b>
[86] <b>StFlg</b>	[87] <b>IfFlg</b>	[88] <b>DMS</b>	[89] <b>pi</b>	[80] <b>Grad</b>
[8A] <b>Reg</b>	[8B] <b>HEX</b>	[8C] <b>BIN</b>	[8D] <b>OCT</b>	[8E] <b>OR</b>
[91] <b>R/S</b>	[00] <b>0</b>	[93] <b>.</b>	[94] <b>+/-</b>	[95] <b>=</b>
[96] <b>Write</b>	[97] <b>Dsz</b>	[98] <b>Adv</b>	[99] <b>Prt</b>	[90] <b>Lst</b>
[9A] <b>phi</b>	[9B] <b>DEC</b>	[9C] <b>Inc</b>	[9D] <b>NOT</b>	[9E] <b>%</b>



	A	B	C	D	E
A'	●	B'	C'	D'	E'
A''	●	B''	C''	D''	E''
2nd	INV	Inx	CE	CLR	ON
3rd	HIR code	log log2	CP rand	OFF	
LRN	x↔t	x <sup>2</sup>	√x	1/x	
Pgm Temp	P→R x↔y	sin sinh	cos cosh	tan tanh	
SST	STO	RCL	SUM	y <sup>x</sup>	
Ins Bat	CMs x!	Exc Inx!	Prd logx!	Ind mod2	
BST	EE	( )	÷		
Del LCD	Eng SHL	Fix SHR	Int round	x  mod	
GTO	7	8	9	x	
Pause REL	x=t 0D	Nop 0E	Op 0F	Deg AND	
SBR	4	5	6	-	
Lbl IF	x≥t 0A	Stat 0B	Mean 0C	Rad XOR	
RST	1	2	3	+	
StFlg Reg	IfFlg HEX	D.MS BIN	pi OCT	Grad OR	
R/S	0	.	+/-	=	
Write phi	Dsz DEC	Adv Inc	Prt NOT	Lst %	

## 8. Indicateurs à l'écran

L'écran LCD contient 2 rangées de 16 caractères alphanumériques. La première rangée est généralement utilisée pour afficher les indicateurs, la deuxième rangée pour afficher le nombre en cours (registre X) : saisie, résultat d'opération...

La signification de la première ligne de l'écran peut être modifiée à l'aide d'instructions :

**Op 1D** ... indicateurs (mode par défaut)  
**Op 1E** ... registre T  
**Op 1F** ... texte (ce mode peut être désactivé en utilisant **CLR**)



Indicateurs sur la première ligne de l'écran :

**Deg/Rad/Grd** - indication de l'unité des angles en degrés, radians ou grades.  $360^\circ = 2 * \text{PI radians} = 400 \text{ grades}$ . Le commutateur peut être changé avec les boutons **Deg**, **Rad** ou **Grad**.

**Hex/Bin/Oct** - indique le système numérique sélectionné : décimal (non indiqué), hexadécimal (Hex), binaire (Bin) et octal (Oct). Le système numérique peut être sélectionné avec les boutons **DEC**, **HEX**, **BIN** ou **OCT**.

**F0 to FD** - indique l'arrondi sélectionné des nombres de 0 à 13 décimales. Il est réglé avec les boutons **Fix 0** à **Fix 0D**. La séquence **INV Fix (Fix 0F)** a la même signification) désactive l'arrondi du résultat affiché. Dans ce cas, l'arrondi n'est pas indiqué sur l'écran (il est remplacé par des espaces).

**EE/Eng** - indique comment l'exposant est affiché. Après avoir appuyé sur **EE**, le nombre s'affiche sous forme scientifique, au format mantisse et exposant. Le mode peut être annulé en appuyant sur **CLR** ou **INV EE**.

Après avoir appuyé sur **2nd** **Eng**, le nombre s'affiche sous forme technique ("ingénieur"), où l'exposant est un multiple de 3. Le mode **Eng** est désactivé en appuyant sur **INV** **Eng**. Si les modes exposant sont désactivés, rien n'est indiqué sur l'écran.

**2n/3d** - indique la première ou la deuxième pression de la touche **2nd**, la touche de fonction alternative. Si une touche est enfoncée après avoir appuyé une fois sur la touche **2nd**, une fonction alternative (visible sur le clavier dans la deuxième ligne) est exécutée à la place de sa fonction de base. Après avoir appuyé une deuxième fois sur la touche **2nd**, 3d apparaît sur l'écran. Dans ce cas, la deuxième fonction alternative de la touche sélectionnée est exécutée (visible sur la troisième ligne du clavier). Si la touche **2nd** n'est pas utilisée, ou si elle est utilisée 3 fois consécutives, la fonction alternative n'est pas active, la fonction de base du bouton est exécutée. Cet état n'est pas indiqué sur l'écran (des espaces apparaissent à la position).

**In** - indication de la pression sur la touche **INV**, activant la fonction inverse. Si la touche **INV** (fonction alternative inverse) doit être utilisée en même temps que la touche **2nd**, la touche **INV** doit être pressée avant d'appuyer sur la touche **2nd**.

**Operation courante** - la dernière position de la 1ère ligne est destinée à indiquer l'opération arithmétique sélectionnée : + addition, - soustraction, \* multiplication, : division, & AND [bit], | OR [bit], ~ XOR [bit], \ modulo troncature (mod), / modulo plancher (mod2), % pourcentage, < décalage à gauche, > décalage à droite, ^ puissance, racine V, ( parenthèse ouverte.

## 9. Signalement d'erreur

Si une erreur se produit pendant le fonctionnement de la calculatrice, l'erreur est signalée par un clignotement de l'écran. En même temps, un caractère E ou F apparaît au début de la deuxième ligne, indiquant le type d'erreur.

L'erreur de type **E** (Error) est une erreur logicielle. Elle est créée, par exemple, en divisant un nombre par zéro. Le programme peut continuer à s'exécuter et l'erreur ne sera pas indiquée tant que le programme n'aura pas terminé son exécution. L'indication peut être annulée avec le bouton **CE** ou **CLR**.

Le comportement du programme en cas d'erreur peut être influencé par le drapeau 8. Si le drapeau 8 est activé (instruction **StFlg 8**), le programme s'arrêtera après une erreur logicielle. Si le drapeau 8 n'est pas activé (état par défaut), le programme effectue la correction la plus justifiée et continue de s'exécuter. Dans ce cas, l'erreur ne sera indiquée qu'après la fin du programme.

L'état de l'indication d'erreur peut être détecté par programmation à l'aide des commandes **Op 18** et **Op 19**, ainsi que du drapeau 7. En appelant **Op 18**, le drapeau 7 est activé si l'erreur n'est pas indiquée. Sinon, l'état du drapeau ne change pas. L'appel de **Op 19** active le drapeau 7 si une erreur est indiquée. Sinon, l'état du drapeau ne change pas.

La troisième façon de travailler avec l'indication d'erreur est le drapeau 15. Il est directement connecté à l'indication d'erreur et peut être à la fois détecté (avec l'instruction **IfFlg 0F**) et défini (**StFlg 0F**) ou réinitialisé (**INV StFlg 0F**).

*Remarque : Après avoir réinitialisé l'indication d'erreur en désactivant le drapeau 15, le caractère E peut rester allumé sur l'écran. Il ne s'agit pas d'un défaut, le caractère disparaît après le premier changement du contenu de l'écran.*

Vous pouvez également utiliser la méthode utilisée dans les programmes de calculatrice TI-58/59 pour déclencher l'indication d'erreur. La séquence de touches **CLR 1/X** fait clignoter l'écran avec le nombre 9,9999+9999.

L'erreur de type **F** (fatale) est une erreur matérielle qui empêche le

programme de continuer à s'exécuter et entraîne son arrêt immédiat. Elle est créée par un débordement du pointeur du programme derrière la fin de la mémoire, par un débordement de la profondeur de la pile du programme ou de la pile des opérations, par l'utilisation d'une étiquette inexistante ou par l'utilisation d'un numéro de registre non valide.

## 10. Edition des nombres

Le nombre saisi ainsi que le résultat des calculs apparaissent sur la deuxième ligne de l'écran. La mantisse est affichée avec une précision de 14 chiffres max.

Une position pour un signe est réservée devant la mantisse. Le caractère '-' est affiché ici pour les nombres négatifs, laissant un espace pour les nombres positifs.

Un exposant est affiché derrière la mantisse (si le mode exposant est actif). L'exposant est séparé de la mantisse par un signe + ou -. L'exposant est affiché avec 1 à 4 chiffres.

La mantisse peut inclure un point décimal. En mode exposant avec notation scientifique (mantisse et exposant), le point décimal est toujours affiché après le premier chiffre. En mode technique (ingénierie), l'exposant est multiple de 3. Un à trois chiffres sont affichés avant le point décimal. Si le mode exposant n'est pas actif, un point décimal est affiché après le chiffre de l'unité. S'il n'y a pas de chiffres après la virgule décimale, la virgule décimale n'apparaîtra pas.

Si le mode d'affichage hexadécimal ou binaire est actif, l'exposant est un multiple de 4. Un à quatre chiffres sont affichés avant la virgule décimale.

Si un système numérique non décimal est actif, les chiffres de la mantisse sont affichés dans le système numérique sélectionné, mais l'exposant est toujours affiché sous la forme d'un nombre décimal.

La touche **CE** supprime le dernier caractère de la mantisse ou de l'exposant (selon l'écriture en cours des chiffres).

La touche **EE** permet de commencer à saisir l'exposant. Vous pouvez revenir à la saisie de la mantisse en appuyant sur la touche point **.** ou **INV EE**. La touche **EE** est également utilisée pour commencer à éditer le résultat affiché de l'opération. Cela peut être utilisé pour supprimer les chiffres cachés d'un nombre. D'autres alternatives sont **INV .** et **Op 82**.


# 11. Expressions numériques



Lors des calculs, la calculatrice maintient la priorité des opérations sur 3 niveaux :

1. + addition, - soustraction, & AND [bit], | OR [bit], ~ XOR [bit]
2. \* multiplication, ÷ division, \ modulo tronqué (mod), / modulo plancher (mod2), % pourcentage, < shift left (SHL), > shift right (SHR)
3. ^ puissance, √ racine

Dans les calculs, le niveau (3) de puissance et de racine est évalué en premier, puis (2) la multiplication et la division, et enfin (1) l'addition et la soustraction.

Vous pouvez utiliser des parenthèses dans l'expression de manière arbitraire, jusqu'au niveau 15.

La touche  est utilisée pour échanger les premier et deuxième opérandes de l'opération.

Après avoir effectué le calcul, le niveau de calcul le plus bas peut être répété en appuyant à nouveau sur la touche . La saisie d'un nombre et l'appui sur  répètent l'opération, où le deuxième opérande sera le dernier deuxième nombre.

Exemple :

    [5]

  [6]

  [12]

## 12. Adressage indirect

En plus de la saisie directe des paramètres des instructions (leur valeur numérique), les paramètres peuvent également être saisis indirectement, à l'aide de registres de données. Dans ce cas, le numéro de registre à partir duquel la calculatrice doit extraire le paramètre est spécifié comme paramètre. Utilisez le bouton **Ind** pour changer le pointeur vers une adresse indirecte. Dans certains cas, un autre code d'instruction (destiné à l'adressage indirect) est stocké dans le programme, dans d'autres cas, le code **Ind** est stocké comme indicateur d'adressage indirect.

Pour les instructions de saut, le registre contient l'adresse de saut absolue sous forme de nombre décimal compris entre 0 et 999. En plus de l'adresse de saut absolue, un symbole d'étiquette peut également être stocké dans le registre. L'étiquette est stockée dans le registre sous forme de code d'étiquette, exprimé par un nombre décimal et multiplié par 256 (le code d'étiquette est dans l'octet supérieur du nombre). Par exemple, le bouton **1/x** a un code de programme hexadécimal de 35. Cela correspond au nombre décimal 53 ( $3 \times 16 + 5 = 53$ ). La multiplication par 256 donne une valeur de 13568 (ou 3500 en code HEX).

Le simple fait de placer la commande **Ind** dans le programme exécute une instruction dont le code est stocké dans le registre de données, dont le code est donné après l'instruction **Ind**. L'instruction **Ind** elle-même n'est normalement pas destinée à fonctionner avec son propre paramètre, et il est donc nécessaire soit d'utiliser un code de registre à un chiffre (chiffres de 0 à 9) ou de faire un pas en arrière et de corriger le numéro de registre. Le code d'instruction doit être stocké sous forme décimale dans le registre. Si l'instruction nécessite des paramètres, ils sont lus à partir de la partie du programme qui suit le code **Ind**.

Exemple, adressage indirect du registre :

**5 STO 01** ... stocke la valeur 5 dans le registre R01 (adressage direct)

**8 STO 05** ... stocke la valeur 8 dans le registre R05 (adressage direct)

**RCL Ind 01** [8] ... lit la valeur 5 du registre R01 et l'utilise comme adresse du registre R05, à partir duquel il lit la valeur résultante 8

Au lieu du code 43 01 (instruction **RCL 01**), le code 73 01 (instruction



**RCL** **Ind** **01**) est stocké dans le programme.

Exemple, adresse absolue indirecte :

**1** **2** **3** **STO** **0** **1** ... stocke la valeur 123 dans le registre R01

**GTO** **Ind** **0** **1** ... lit la valeur 123 du registre R01 et l'utilise comme saut vers la nouvelle adresse

**LRN** [123 FF ...empty] ... vérifie l'adresse du pas courant.

Exemple, adressage indirect d'un label :

**RST** **LRN** ... activation du mode programmation

**Lbl** **1/x** ... saisie du label **1/x**

**LRN** ... sortie du mode programmation

**RST** ... Retour au pas 000 (pour contrôle du saut lors de l'exécution)

**1** **3** **5** **6** **8** **STO** **0** **1** ... stocke la valeur du label **1/x** dans le registre R01 (code de 1/x = 35 hex = 53 decimal, \* 256 = 13568)

**GTO** **Ind** **0** **1** ... choix du label depuis le registre R01 et saut au label **1/x**

**LRN** (002 FF ...empty) ... vérification exécution du saut (après **1/x**)

Exemple, adressage indirecte vers une instruction:

**RST** **LRN** ... activation du mode programmation

**Ind** **1** **2** ... instruction indirecte depuis registre R01 avec paramètre '2'

**LRN** ... sortie du mode programmation

**4** **5** **STO** **0** **2** ... stocke la valeur 45 dans le registre R02

**6** **7** **STO** **0** **1** ... stocke la valeur décimale du code de l'instruction **RCL** dans le registre R01

**RST** ... Retour au pas 000

**SST** [45] ... exécute l'instruction à l'adresse 000. Récupère le code de l'instruction **RCL** (43 hex = 67 dec) depuis le registre R01, l'exécute, l'instruction **RCL** lit le paramètre suivant l'instruction **Ind** **1**, c'est à dire la valeur '2', et l'instruction **RCL** **02** est exécutée, laquelle affiche le contenu

du registre R02 (dans notre cas la valeur 45).

**LRN** [003 FF ...empty] ... vérification exécution (après séquence **Ind 01**  
**02**).

## 13. Programmation

L'écriture d'une séquence de touches dans la mémoire du programme est appelée un programme. Avec un programme, la calculatrice devient un outil puissant. Il existe un programme utilisateur, inscriptible dans la mémoire EEPROM (c'est-à-dire la mémoire dont le contenu est enregistré dans le processeur même après le retrait de la batterie), et une bibliothèque de programmes, qui est stockée dans la mémoire ROM. La bibliothèque de programmes ne peut pas être écrasée par l'éditeur de programmes.

Le mode de programmation est activé par le bouton **LRN**. Le contenu du programme est affiché sur deux lignes de l'écran. Sur la ligne inférieure à gauche, il y a 3 chiffres, représentant l'adresse de l'étape de programme en cours dans la mémoire. L'adresse est comprise entre 000 et 999 (soit 1000 étapes de programme).



Le code HEX à deux chiffres de l'octet de programme à l'adresse donnée est affiché après l'adresse. Le code d'octet est suivi du nom du bouton ou de la fonction qui correspond à ce code. La calculatrice ne peut pas distinguer dans l'éditeur de programme s'il s'agit du début d'une instruction ou d'un paramètre, et affiche donc le texte approprié du nom du bouton pour tous les codes. Il appartient à l'utilisateur de distinguer selon le code de l'instruction précédente, s'il s'agit d'un code d'instruction ou d'un paramètre.

La ligne supérieure affiche le contenu des 4 octets adjacents du programme. Cela facilite la navigation dans le programme. L'octet de programme actuel (qui est sélectionné dans la ligne inférieure) est affiché au milieu de la ligne supérieure et est en outre encadré par des crochets..

Le mode de programmation peut être activé même si un programme de la bibliothèque est actif (par exemple en sélectionnant **Pgm 02**). Dans ce cas, l'écran n'affiche pas le programme utilisateur (comme sur la TI-58/59

d'origine), mais le contenu du programme de la bibliothèque. Le programme peut uniquement être parcouru, il ne peut en aucun cas être modifié. Le programme de la bibliothèque se distingue du programme principal par le fait qu'un astérisque \* est affiché entre l'adresse et le code octet du programme (comme indication du mode "Lecture seule").

## Touches utiles à la programmation

**SST** (Single Step) - Augmente le pointeur de programme de 1 (pas suivant). Le bouton **SST** peut également être utilisé en mode normal (exécution). Après avoir appuyé dessus, le code de l'instruction sur laquelle le pointeur de programme est placé est exécuté.

**BST** (Back Step) - Diminue le pointeur de programme de 1 (pas précédent).

**Ins** (Insert) - Insère un octet vide à la position actuelle du programme et décale la partie suivante du programme vers le bas. Un octet vide avec la valeur FF et est marqué avec l'étiquette "... vide" est inséré dans l'éditeur. Cet octet a une fonction similaire à celle de l'instruction **Nop** lors de l'exécution (c'est-à-dire que rien n'est fait), mais il a une signification particulière lors de l'édition du programme. Les boutons **Ins** et **Del** déplacent le reste de la mémoire du programme derrière le pointeur du programme actuel. Cependant, si un code FF est rencontré, l'opération est interrompue. Le code FF agit ainsi comme une sorte d'espace flexible séparant les différentes parties du programme. Il garantit que les sections suivantes du programme, séparées par l'espace FF, restent en place. Cela convient, par exemple, dans les cas d'adressage absolu. De plus, il accélère les opérations **Ins** et **Del**, qui peuvent prendre plusieurs secondes lorsque la mémoire est pleine. L'espace séparateur est appliqué jusqu'à ce qu'il disparaisse complètement (en insérant plus d'octets). Les sections fusionneront et continueront à se déplacer ensemble.

*Notez que certaines instructions vous permettent de saisir un paramètre dans le code HEX et d'insérer un octet valide avec une valeur de 0FF dans le code. Si possible, évitez une telle valeur d'octet, car elle serait interprétée comme un espace vide et endommagée lors du déplacement du programme en mémoire.*

**Del** (Delete) - Supprime l'octet à la position actuelle du programme et décale la partie suivante du programme vers le haut. Pendant l'opération, des octets FF vides sont appliqués, comme décrit dans l'instruction **Ins**.

Il convient de noter que les instructions **Ins** et **Del** ne corrigent pas les adresses de saut absolues dans le programme. Pour cette raison, il est préférable d'utiliser soit des sauts relatifs, soit, mieux encore, des étiquettes. Avec la calculatrice TI-58/59 d'origine, les adresses absolues étaient préférées en raison de leur vitesse plus élevée, car la recherche d'étiquettes dans le programme peut prendre beaucoup de temps (même quelques secondes). En revanche, l'ET-58 recherche très rapidement les étiquettes du programme et peut donc être utilisée comme un remplacement à part entière des adresses absolues.

Si quelqu'un a du mal à déterminer quels boutons il peut encore utiliser comme étiquettes dans un programme, il peut utiliser à cet effet les codes numériques des boutons. Les codes 0A0 à 0FE conviennent à cet effet, car ils ne sont attribués à aucun bouton et peuvent également être utilisés comme étiquettes de programme. Vous pouvez utiliser le bouton **code** pour entrer à partir du clavier, suivie des deux chiffres du code d'étiquette.

**LRN** (Learn) - Quitte le mode programmation et ramène la calculatrice au mode exécution.

**GTO** (Go To) - Le bouton **GTO** ne peut pas être utilisé directement comme touche de commande en mode programmation, car son code est inséré dans le programme lorsqu'il est enfoncé. Cependant, il peut être utilisé en mode exécution en désactivant temporairement le mode de programmation en appuyant sur **LRN**, en appuyant sur **GTO** et en saisissant une adresse numérique à 3 chiffres ou un libellé de bouton, et en appuyant sur **LRN** pour revenir au mode programmation. Le pointeur de programme sera déplacé vers l'adresse spécifiée.

**R/S** (Run/Stop) - Démarrage ou arrêt du programme (utilisé en mode exécution).

**RST** (Reset) - Comme **GTO**, le bouton **RST** ne peut pas être utilisé directement en mode programmation, mais il peut être utilisé en mode exécution pour ramener le pointeur de programme à l'adresse 000. Si un programme de bibliothèque est actif, il est désactivé et ramené au programme utilisateur principal.

Exemple:

**Pgm 02** ... sélectionne le programme de bibliothèque 2

**GTO 1 2 3** ... déplace le pointeur du programme vers l'adresse 123

**LRN** [123\*76 Lbl]... passe en mode programmation pour vérifier l'adresse

**LRN** ... quitte le mode de programmation

**RST** ... réinitialise le pointeur du programme et revient au programme principal

**LRN** [000 ...] ... passe en mode programmation pour vérifier l'adresse

## 14. Boutons et instructions

Pour chaque bouton, le code HEX du programme, le nom du bouton et la séquence d'appuis sur le bouton sont précisés.

### 00 ... 09 Chiffres de base, 0...9

Libellé : **0** ... **9**

Séquence de touches : **0** ... **9**

Les chiffres de base sont utilisés pour saisir des chiffres compris entre 0 et 9, généralement un nombre décimal. Ils sont utilisés pour saisir la mantisse du nombre, pour saisir l'exposant, le numéro du registre de mémoire, l'adresse de saut absolue, etc...

Les chiffres séparés sont stockés dans le programme avec les codes 0 à 9. S'ils font partie d'un nombre composé, tel qu'un numéro de registre ou une adresse de saut absolue, ils sont stockés dans le programme en code BCD (c'est-à-dire 2 chiffres par octet).

Exemple:

La séquence **GTO** **1** **2** **3** sera stockée dans le programme codée sous la forme 61 01 23.

La séquence **STO** **1** **2** sera stockée avec les codes 42 12.

### 0A ... 0F Chiffres hexadécimaux, 0A...0F

Libellé : **0A** ... **0F**

Séquence de touches : **2nd** **4** ... **2nd** **9**

Les chiffres hexadécimaux peuvent être utilisés pour saisir des chiffres de 10 à 15 lorsque cela est possible :

- par exemple saisir la mantisse du nombre sous forme

hexadécimale (pas l'exposant qui est toujours saisi sous forme décimale)

- ou aussi saisir les paramètres des instructions **HIR** et **Op**.

Un cas particulier concerne les paramètres qui attendent une forme décimale du nombre. Il s'agit généralement du nombre de registres de mémoire. Par exemple, l'instruction **RCL 2 3** est stockée dans le programme sur 2 octets, avec le code 43 23. Lors de l'interprétation du programme, le numéro de registre est lu en composant les deux chiffres comme  $2*10 + 3 = 23$ .

## 10 ... 1F Labels alphabétiques, A...F

Libellé : **A** ... **E**, **A'** ... **E'**, **A''** ... **E''**, **F**

Séquence de touches : **A** ... **E**, **2nd A** ... **2nd E**, **2nd 2nd A** ... **2nd 2nd E**

Les boutons **A** à **E** permettent d'appeler rapidement des sous-programmes, généralement par simple pression. Le sous-programme est marqué dans le programme par une étiquette portant le symbole spécifié, par exemple **Lbl A**. Après avoir appuyé sur la touche **A**, le sous-programme concerné est appelé.

Les boutons alphabétiques sont disponibles en 3 ensembles. L'ensemble de base, **A** to **E**, est activé par simple pression sur le bouton **A** à **E**, sans préfixe **2nd**. Le second ensemble, **A'** à **E'**, est activé par pression de **2nd** suivi d'un bouton **A** à **E**. Le troisième ensemble, **A''** à **E''**, est activé par deux pressions successives de **2nd** (c'est à dire **3rd**) suivi d'un bouton **A** à **E**.

Un cas particulier est l'instruction **F**, avec le code de programme 1F. Le code n'est pas disponible directement depuis le clavier sous forme de bouton. Il peut être saisi de manière plus complexe, en utilisant l'instruction **code** : **2nd 2nd INV 0 2nd 2nd 0F**. Une telle utilisation serait peu pratique. L'utilisation réelle se fait à l'intérieur du programme, pour invoquer une sous-routine en utilisant uniquement le code d'instruction de 1 octet.



L'appel des sous-programmes avec les touches **A** à **F** s'effectue de la même manière que l'appel d'un sous-programme avec l'instruction **SBR**. L'adresse suivant le code de la touche est d'abord stockée dans la pile d'adresses. Le contrôle du sous-programme est ensuite transmis. La pile d'adresses a une capacité limitée à 15 sous-programmes.

Le sous-programme est terminé par l'instruction **RTN**. Cela permettra de revenir du sous-programme. L'adresse d'origine après l'instruction **A** à **F** est extraite de la pile d'adresses et le contrôle est transmis à cette adresse. Si le sous-programme a été démarré à partir du clavier, le programme s'arrête.

Les sous-programmes peuvent également être appelés à partir d'un autre programme de bibliothèque. Le programme contient l'instruction **Pgm** suivie du numéro de programme et du code de touche **A** à **F** ou l'appel du sous-programme via **SBR**. Si l'instruction **Pgm** est utilisée dans le programme, le programme actif n'est pas commuté de manière permanente (comme ce serait le cas lors d'une utilisation à partir du clavier), le changement n'est que temporaire pendant la durée de l'appel d'un sous-programme suivant. À la fin du sous-programme, le contrôle est transféré vers le programme d'origine.

Exemple:

**RST** **LRN** ... active le mode de programmation

**Lbl** **A** ... création de l'étiquette A

**BST** **code** **1** **0F** ... retour au pas précédent et saisie du code de l'instruction 1F

**[** **CE** **+** **1** **]** **RTN** ... le sous-programme incrémente la valeur du registre X de 1

(l'instruction **RTN** est obtenue par saisie de **INV** **SBR**)

**Lbl** **A** **code** **1** **0F** **RTN** ... le programme de test appelle le sous programme 1F

**LRN** ... sortie du mode programmation

**1** **A** **[2]** **A** **[3]** **A** **[4]** ... test, chaque appui sur **A** incrémente X de 1

## 20 Éteindre la calculatrice, OFF

Libellé : **OFF**

Séquence de touches : **2nd CLR**

L'instruction **OFF** éteint la calculatrice et la met en veille. Mais la calculatrice s'éteint aussi automatiquement après une certaine période d'inactivité. Le temps d'inactivité peut être réglé avec l'opération **Op 88** (par défaut 30 secondes). La réactivation est possible en appuyant sur le bouton **CLR**.

Lors de l'arrêt de la calculatrice, la mémoire RAM, qui contient les registres de données, les registres de travail (X, Y, T) et les opérations en attente est préservée. La condition de stockage est que la batterie ne soit pas retirée de la calculatrice pendant plus de quelques minutes et que le bouton **CLR** ne soit pas enfoncé. Sinon, le bouton **CLR** allumerait la calculatrice qui, sans la batterie, serait réinitialisée. La batterie peut être remplacée pendant que la calculatrice est éteinte. Si la batterie est retirée de la calculatrice allumée et qu'aucune alimentation externe n'est connectée, la calculatrice se réinitialise lorsque la nouvelle batterie est insérée et alors le contenu de la mémoire RAM est perdu (le contenu des registres est remis à zéro)..

La pile ne doit pas être retirée de la calculatrice pendant les opérations d'écriture dans la mémoire EEPROM ou ROM, c'est-à-dire pendant la programmation du programme utilisateur (par exemple, une opération **Ins** en cours) ou pendant le chargement de la bibliothèque utilisateur de la carte SD vers la ROM. Dans ce cas, le contenu de la mémoire peut être endommagé.

Si la calculatrice est alimentée par une source externe (connecteur d'alimentation USB), l'écran restera allumé même lorsque la calculatrice est éteinte.

## 21 Fonction alternative function, 2nd

Libellé : **2nd**

Séquence de touches : **2nd**

Le bouton **2nd** permet de changer la signification du bouton suivant en fonction alternative. La plupart des boutons ont 2 fonctions alternatives. Après avoir appuyé une fois sur le bouton **2nd**, la première fonction alternative est exécutée. Après avoir appuyé deux fois sur le bouton **2nd**, la deuxième fonction alternative est exécutée. Trois pressions consécutives, soit **2nd 2nd 2nd**, ramènent aux fonctions de base.

Le code du bouton **2nd** n'est pas enregistré dans le programme. C'est le code de la fonction alternative qui est enregistré.

Exemple:

**2 In x** [.6931...] ... calcule le logarithme naturel du nombre

**2 2nd In x** [.3010...] ... logarithme décimal du nombre (instruction **log**)

**2 2nd 2nd In x** [1] ... logarithme binaire du nombre (instruction **log2**)

## 22 Inverse d'une fonction, INV

Libellé : **INV**

Séquence de touches : **INV**

Le bouton INV, pressé avant un autre code de fonction, invoque la fonction inverse. Dans certains cas, il s'agit d'une fonction alternative supplémentaire.

Le code correspondant au bouton INV est stocké dans le programme avec la valeur 22. La plupart des instructions utilisent directement le préfixe INV qui les précède ou au moins le réinitialisent. Dans le cas des étiquettes Lbl, l'état du préfixe INV est conservé. Cela peut être utilisé pour distinguer la fonction du programme en utilisant l'instruction INV avant l'étiquette du programme.

Exemple:

**RST** **LRN** ... active le mode programmation

**Lbl** **B** ... label de la première routine

**INV** ... l'instruction suivant **Lbl** sera inversée

**Lbl** **A** ... label de la deuxième routine

**sin** ... calcule le sinus

**RTN** ... retour de la routine (saisi avec les boutons **INV** **SBR**)

**LRN** ... revient au mode exécution

**3** **0** **A** [0.5] **B** [30] ... la routine **A** calcule le sinus de l'angle spécifié, la routine **B** calcul l'arcsinus (**INV** **SIN**) de la valeur spécifiée.

## 23 Logarithme naturel et exposant, $\ln x$

Libellé : **ln x**

Séquence de touches : **ln x**

La touche **ln x** calcule le logarithme naturel du nombre affiché. Le logarithme naturel utilise la constante d'Euler avec la valeur 2,718281828459 comme base. Si la touche **INV** est pressée en premier, la fonction inverse, l'exposant naturel, est exécutée.

L'argument de la fonction **ln x** doit être un nombre positif, différent de zéro. Dans le cas d'un nombre nul, l'affichage clignote avec la valeur 9,9999+9999 comme indicateur d'erreur. Dans le cas d'un nombre négatif, la valeur absolue du nombre est calculée et l'affichage clignote à nouveau avec une indication d'erreur.

L'argument de la fonction **INV** **ln x** peut être un nombre positif ou négatif, compris entre -23025 et +23025. Un nombre en dehors de cette plage entraînera un débordement des données et indiquera une erreur.

Exemple:

**5** **ln x** [1.6094...] ... calcule le logarithme naturel de 5 (ie 1.6094 ...)

**INV** **ln x** [5] ... calcule l'exposant naturel du nombre affiché (ie 5)

## 24 Effacement erreur, CE

Libellé : **CE**

Séquence de touches : **CE**

La touche **CE** permet d'annuler l'indication d'erreur E, qui se fait par clignotement de l'écran. Parallèlement à la réinitialisation de l'indication d'erreur, les registres de travail X, T et Y sont corrigés de manière à ce qu'ils ne contiennent pas de nombre avec une indication de dépassement - c'est-à-dire que le nombre indiqué 9,9999+9999 passe à une valeur d'environ 7,0773+9863, qui est la valeur maximale affichable par la calculatrice.

Lors de la saisie d'un nombre sur l'écran, le dernier caractère du nombre saisi est supprimé avec la touche **CE**. Si la mantisse est en cours d'édition, le dernier caractère de la mantisse est supprimé. Si l'exposant est en cours d'édition, le dernier caractère de l'exposant est supprimé. Si un exposant avec une valeur de 0 est supprimé, l'exposant est annulé et la mantisse redevient en cours d'édition.

## 25 Effacer affichage, CLR

Libellé : **CLR**

Séquence de touches : **CLR**

Le bouton **CLR** effectue plusieurs opérations d'initialisation. Il réinitialise les opérations arithmétiques démarrées, réinitialise l'indication d'erreur, désactive le mode texte de l'affichage (activé par la commande Op 1F), renvoie les polices d'affichage par défaut, désactive le mode exposant EE, efface le registre X et démarre l'édition d'un nouveau nombre avec une valeur par défaut de 0.

Le bouton **CLR** ne réinitialise pas le registre T, les registres de données

ou les registres HIR.

Une fonction spéciale du bouton **CLR** permet d'allumer la calculatrice. Le bouton **CLR** est également utilisé pour éteindre la calculatrice si la touche **2nd** (fonction **OFF**) est appuyée avant.

## 26 Sous-programme avec adresse indirecte, SBR Ind

Libellé : **SBR Ind**

Séquence de touches : **SBR 2nd y^x**

La commande **SBR Ind** est suivie d'un numéro de registre contenant une adresse de saut ou un symbole d'étiquette. La commande appelle une sous-routine avec une adresse ou une étiquette contenue dans le registre. (voir chapitre Adressage indirect).

Sinon, la commande fonctionne de la même manière que la commande **SBR** (voir 71 Sous-routine, SBR), c'est-à-dire qu'elle stocke l'adresse actuelle dans la pile du programme (la profondeur maximale de la pile est de 15 adresses) et continue à l'adresse stockée d'origine une fois la sous-routine terminée (fin de la sous-routine avec l'instruction **RTN** 92 Retour de la sous-routine, RTN).

## 27 Instruction interne indirecte, HIR Ind

Libellé : **HIR Ind**

Séquence de touches : **2nd INV 2nd y^x** (seulement dans programme)

L'instruction HIR Ind fonctionne de manière similaire à l'instruction HIR, mais au lieu du registre HIR de travail, elle utilise un registre de données dont l'index est contenu dans le registre HIR spécifié comme paramètre d'instruction.

**Remarque** : Pour des raisons internes, la fonction **HIR Ind** ne peut pas être appelée directement depuis le clavier avec la séquence **HIR Ind**. La fonction ne peut être utilisée que de cette manière à l'intérieur du programme. Il est possible d'appeler la fonction depuis le clavier en

saisissant son code à l'aide de la commande **code** (c'est-à-dire en appuyant sur **2nd 2nd INV 2 7**).

Exemple:

**4 5 STO 01** ... stocke la valeur 45 dans le registre R01

**1 HIR 05** ... stocke la valeur 1 dans le registre HIR H5

**code 27 15** [45] ... Invoque la fonction **HIR Ind** avec le code 27. La fonction lit la valeur 1 from register H5 and reads contents 45 from register R01.

Le paramètre **15** signifie : **1=RCL**, **5=registre HIR H5**.

**HIR Ind 15** est donc l'équivalent de **RCL IND H5** soit **RCL 01** puisque H5 contient 1.

(voir 82 Gestion registres internes, HIR)

## 28 Logarithme décimal et exposant, log

Libellé : **log**

Séquence de touches : **2nd ln x**

La touche **log** calcule le logarithme décimal du nombre affiché. Le logarithme décimal utilise le nombre 10 comme base. Si la touche **INV** est pressée en premier, la fonction inverse, l'exposant décimal, est exécutée.

L'argument de la fonction **log** doit être un nombre positif, différent de zéro. En cas de zéro, l'écran clignote avec la valeur 9,9999+9999 comme indication d'erreur. Pour un nombre négatif, la valeur absolue du nombre est calculée et l'écran clignote à nouveau avec une indication d'erreur.

L'argument de la fonction **log INV** peut être soit un nombre positif ou un nombre négatif, dans la plage d'environ -9863 à +9863. Un nombre en dehors de cette plage entraînera un débordement des données et indiquera une erreur.

Exemple:

**5** **log** [.69897...] ... logarithme décimal de 5 (ie 0.69897...)

**INV** **log** [5] ... exposant décimal du nombre affiché (ie 5)

## 29 Effacement du programme et du registre T, CP

Libellé : **CP**

Séquence de touches : **2nd** **CE**

Le bouton **CP** a deux fonctions : supprimer le programme utilisateur et effacer le registre T.

Si la touche **CP** est utilisée en mode exécution, le programme utilisateur est supprimé. Il vous sera demandé de confirmer l'opération avant de supprimer. Appuyez sur le bouton **1** pour confirmer l'opération et le programme utilisateur sera supprimé. Le registre T est supprimé même si l'opération de suppression du programme n'est pas confirmée. Dans ce cas, la commande est utilisée uniquement pour effacer le registre T.

Si la touche **CP** est utilisée dans le programme, seul le registre T est effacé.

## 2B Saisie d'un code d'instruction, Code

Libellé : **code**

Séquence de touches : **2nd** **2nd** **INV**

Utilisez l'instruction **code** pour saisir directement le code numérique hexa d'une fonction. Cela vous permet de saisir des codes de touches difficiles ou pas du tout accessibles depuis le clavier. Le code de touche à deux chiffres est saisi comme paramètre d'instruction. Le code de touche est exécuté comme si la touche avait été enfoncée sur le clavier. De cette manière, des codes de touches non standard peuvent également être saisis dans le programme pendant la programmation. Le code d'instruction



2B lui-même est ignoré pendant l'exécution du programme.

Exemple:

**5** **STO** **01** ... stockage de la valeur de test 5 dans le registre R01

**CLR** [0] ... effacer l'affichage

**code** **43** **01** [5] ... saisie du code de la fonction RCL et exécution de l'instruction RCL 01 (renvoie le contenu 5)

## 2C Logarithme binaire et exposant, $\log_2$

Libellé : **log2**

Séquence de touches : **2nd** **2nd** **ln x**

La touche **log2** calcule le logarithme binaire du nombre affiché. Le logarithme binaire utilise le nombre 2 comme base. Si la touche **INV** est pressée en premier, la fonction inverse, l'exposant binaire, est exécutée.

L'argument de la fonction **log2** doit être un nombre positif, différent de zéro. Dans le cas de zéro, l'affichage clignote avec la valeur 9,9999+9999 comme indication d'erreur. Pour un nombre négatif, la valeur absolue du nombre est calculée et l'affichage clignote à nouveau avec une indication d'erreur.

L'argument de la fonction **INV** **log2** peut être à la fois un nombre positif et un nombre négatif, dans la plage -32767 à +32767. Un nombre en dehors de cette plage entraînera un débordement des données et indiquera une erreur.

Exemple:

**5** **log2** [2.3219...] ... calcule le logarithme binaire de 5 (ie 2.3219 ...)

**INV** **log2** [5] ... calcule l'exposant binaire du nombre affiché (ie 5)

## 2D Générateur de nombres aléatoires, rand

Libellé : **rand**

Séquence de touches : **2nd** **2nd** **CE**

La fonction **rand** calcule un nombre aléatoire dans la plage de 0 à 1 (ou jusqu'au nombre limite spécifié, avec le préfixe **INV**), y compris la valeur 0, mais excluant la valeur 1. Le générateur LCG (générateur congruentiel linéaire) avec la formule  $\text{RandSeed} = \text{RandSeed} * 214013 + 2531011$  est utilisé pour calculer le nombre aléatoire. La graine du générateur a une plage de 32 bits. Le nombre généré est converti en nombre flottant en le divisant par  $2^{32}$ . Cela garantit que le nombre aléatoire résultant est compris entre 0 et 1, y compris zéro, mais excluant la valeur 1. Le nombre limité de chiffres du nombre de base (9 chiffres et demi, plage de nombres de 0 à 4294967295) garantit une telle granularité des données que même après multiplication, il ne déborde pas jusqu'à la valeur limite 1.

Si le préfixe **INV** est pressé avant l'instruction **rand**, le nombre aléatoire généré est multiplié par la valeur du registre X. Cela permet de générer des nombres dans une plage donnée, de zéro au nombre maximum spécifié, mais en excluant le nombre maximum spécifié.

Le générateur de nombres aléatoires compte en continu à chaque fois qu'il est utilisé, ce qui garantit que les séquences de nombres générées ne se répètent pas. De plus, à chaque réinitialisation de la calculatrice, la valeur du générateur est lue dans l'EEPROM et une nouvelle valeur est stockée. Cela garantit que les séquences générées ne se répètent pas même après la réinitialisation de la calculatrice (ou après le retrait de la batterie).

Par non-répétition, on entend une séquence de petits nombres générés. Si de grands nombres couvrant la plage du générateur (9,5 chiffres) sont générés, la séquence générée ne pourra être répétée que longtemps après.

La semence du générateur est contrôlée par **Op** **51** et **Op** **52**.

Exemple d'un programme de lancer de dés :

**Lbl** **A** **(** **6** **INV** **rand** **Int** **+** **1** **)** **RTN** ... Appuyer sur **A** génère un nombre



## 31 Programmation, LRN

Libellé : **LRN**

Séquence de touches : **LRN**

Le bouton **LRN** permet d'activer ou de désactiver le mode de programmation. Le mode de programmation est décrit plus en détail dans la section Programmation.

## 32 Exchange of registers X and T, x<>t

Libellé : **x<>t**

Séquence de touches : **x<>t**

La touche **x<>t** permet d'échanger les contenus des registres X et T. Le registre X est un registre de travail ou également le contenu de l'affichage. Le registre T est un registre auxiliaire (temporaire). Il est utilisé pour comparer des nombres, pour convertir des coordonnées polaires et cartésiennes et pour calculer des nombres complexes.

Le registre X est réinitialisé par les touches **CLR** et **CE**. Le registre T est réinitialisé par le bouton **CP** (attention, il est également utilisé pour supprimer le contenu du programme utilisateur).

Le registre T n'est normalement pas visible sur l'écran. En utilisant l'instruction **Op 1E**, il est possible d'activer un mode d'affichage spécial, où le registre T est affiché sur la ligne supérieure de l'écran. Le retour au format d'affichage standard avec une rangée de commutateurs est possible avec l'instruction **Op 1D**.

### 33 Carré d'un nombre, $x^2$

Libellé :  $x^2$

Séquence de touches :  $x^2$

La touche  $x^2$  calcule le carré du nombre, c'est-à-dire le multiple du nombre par lui-même.

### 34 Racine carrée d'un nombre, $\sqrt{x}$

Libellé :  $\sqrt{x}$

Séquence de touches :  $\sqrt{x}$

Utilisez la touche  $\sqrt{x}$  pour calculer la racine carrée du nombre. Le nombre ne doit pas être négatif. Si un nombre négatif est calculé, la racine carrée de la valeur absolue du nombre est calculée et l'indication d'erreur E est définie (l'affichage clignote).

### 35 Inverse d'un nombre, $1/x$

Libellé :  $1/x$

Séquence de touches :  $1/x$

Utilisez le bouton  $1/x$  pour calculer l'inverse du nombre. Si le nombre est zéro, la valeur 9,9999+9999 s'affiche et l'indication d'erreur E est définie (l'affichage clignote). Cette fonction est souvent utilisée dans les programmes pour activer l'indication d'erreur et pour indiquer un dysfonctionnement du programme.

## 36 Sélection d'un programme de bibliothèque, Pgm

Libellé : **Pgm**

Séquence de touches : **2nd** **LRN**

Le bouton **Pgm** permet de sélectionner un programme dans la bibliothèque. Le paramètre est le numéro à deux chiffres du programme sélectionné, commençant par le numéro 01. Après avoir sélectionné un programme, l'écran affiche brièvement le nom de la bibliothèque, le numéro du programme sélectionné et la longueur du programme en octets. Le nombre de programmes avec la calculatrice ET-58 est limité à 50 programmes (c'est-à-dire le nombre de programmes dans la bibliothèque). Lors de la sélection d'un programme hors de portée, l'indication d'erreur E clignote.

Si un programme de la bibliothèque est sélectionné, les sous-routines de ce programme sont exécutables.

Après avoir activé le mode de programmation avec le bouton **LRN**, le contenu du programme de la bibliothèque sélectionné s'affiche. Vous pouvez parcourir et visualiser le programme, mais vous ne pouvez pas le modifier. Le programme de la bibliothèque est marqué d'un astérisque entre l'adresse et l'octet actuel (indicateur de lecture seule).

La sélection du programme 00 permet de sélectionner le programme utilisateur (le message « Programme principal » apparaît brièvement sur l'écran). Seul le programme utilisateur peut être modifié. La touche **RST** a une fonction similaire, elle commute également la sélection sur le programme utilisateur.

En plaçant le préfixe **INV** devant la touche **Pgm**, le numéro du programme de bibliothèque sélectionné s'affiche à l'écran (registre de travail X) et la première ligne de l'écran affiche brièvement les données sur le programme sélectionné, de manière similaire à la commutation du programme. Dans un programme **INV PGM** permet donc de tester quel PGM est actif.

Si la fonction **Pgm** est utilisée dans un programme en cours d'exécution, le programme ne sera pas commuté de manière permanente. Le changement ne s'applique qu'à l'instruction d'appel de sous-programme suivante (lettres A à F ou sous-programme SBR). De cette manière, vous

pouvez appeler un programme à partir d'un autre programme de bibliothèque, ou même à partir du programme utilisateur principal.

La fonction **Pgm** peut aussi être utilisée en adressage indirect. (voir 62 Sélection indirecte du programme de la bibliothèque, Pgm Ind).

### 37 Conversion de coordonnées cartésiennes et polaires, P->R

Libellé : **P->R**

Séquence de touches : **2nd** **x<>t**

La touche **P->R** convertit les coordonnées de l'expression polaire en coordonnées cartésiennes. Avant l'opération, le registre T (c'est-à-dire le registre auxiliaire) contient le rayon et le registre X (contenu d'affichage) contient l'angle. L'angle est spécifié dans la mesure angulaire actuellement sélectionnée (boutons **Deg**, **Rad** et **Grad**). Après l'opération, le registre T (registre auxiliaire) contient la coordonnée X, le registre X (contenu d'affichage) contient la coordonnée Y. L'angle peut être converti de radians en mesure angulaire actuellement sélectionnée par l'instruction **Op** **73** avant l'opération.

En mettant le préfixe **INV** avant l'instruction **P->R**, l'opération inverse est effectuée : conversion de coordonnées cartésiennes en coordonnées polaires. Avant l'opération, le registre T contient la coordonnée X, le registre X contient la coordonnée Y. Après l'opération, le registre T contient le rayon et le registre X contient l'angle. L'angle est spécifié dans la mesure angulaire actuellement sélectionnée. Si après l'opération il est nécessaire de convertir l'angle en radians, cela peut être fait avec l'instruction **Op** **72**.

L'instruction **Op** **1E** peut être utilisée pour activer un mode d'affichage spécial dans lequel le contenu du registre T est affiché sur la première ligne de l'écran. Cela peut faciliter l'utilisation de la conversion de coordonnées. Utilisez l'instruction **Op** **1D** pour restaurer le mode d'affichage standard.

Exemple.

**1** **0** **x<>t** ... saisie du rayon 10 dans le registre T

**3 0** ... saisie de l'angle of 30 ° dans le register X

**P->R** [5] ... conversion des coordonnées polaires en coordonnées cartésiennes. l'affichage indique la coordonnée Y = 5

**x<>t** [8.6602...] ... échanger X et T affiche la coordonnée X = 8.6602...

**x<>t** [5] ... échange des registres X and T à nouveau

**INV P->R** [30] ... recalcule à l'envers, l'angle de 30 ° est affiché

**x<>t** [10] ... ... échanger X et T affiche le rayon de 10

## 38 Sinus, sin

Libellé : **sin**

Séquence de touches : **2nd** **x^2**

La fonction **sin** calcule le sinus d'un angle. L'angle est spécifié dans les unités définies par les commutateurs **Deg**, **Rad** ou **Grad**. Si un angle calculé en radians est disponible, il peut être converti dans la mesure angulaire actuelle par la fonction **Op 73**.

La saisie du préfixe **INV** avant l'instruction **sin** exécute la fonction opposée arcsinus. Le résultat est un angle dans la mesure angulaire actuellement définie. Si vous devez convertir le résultat en radians, vous pouvez utiliser la fonction **Op 72**.

L'angle calculé par la fonction arcsinus est compris entre -90° et + 90°. La valeur d'entrée de la fonction arcsinus doit être comprise entre -1 et +1. Si elle se situe en dehors de la plage spécifiée, la valeur est limitée à la plage valide et l'erreur E est indiquée (l'affichage clignote).



## 39 Cosinus, cos

Libellé : **cos**

Séquence de touches : **2nd** **Vx**

La fonction **cos** calcule le cosinus d'un angle. L'angle est spécifié dans les unités définies par les commutateurs **Deg**, **Rad** ou **Grad**. Si un angle calculé en radians est disponible, il peut être converti dans la mesure angulaire actuelle par la fonction **Op** **73**.

La saisie du préfixe **INV** avant l'instruction **cos** exécute la fonction opposée arccosinus. Le résultat est un angle dans la mesure angulaire actuellement définie. Si vous devez convertir le résultat en radians, vous pouvez utiliser la fonction **Op** **72**.

L'angle calculé par la fonction arccosinus est compris entre  $0^\circ$  et  $+180^\circ$ . La valeur d'entrée de la fonction arccosinus doit être comprise entre -1 et +1. Si elle se situe en dehors de la plage spécifiée, la valeur est limitée à la plage valide et l'erreur E est indiquée (l'affichage clignote).

## 3A Temperature, Temp

Libellé : **Temp**


Séquence de touches : **2nd** **2nd** **LRN**


Le bouton **Temp** permet de déterminer la température de la puce du processeur et donc la température ambiante (le processeur ne chauffe pratiquement pas en raison de la basse fréquence). La résolution de la mesure de température est de  $1^\circ\text{C}$ . Les données de température sont très imprécises, même après calibrage, elles peuvent différer de plusieurs degrés et ne sont qu'indicatives.

La mesure de température doit d'abord être calibrée avant la première utilisation. Sans calibrage, la valeur par exemple 100 s'affiche. Pour calibrer, entrez la température ambiante actuelle en degrés entiers dans la calculatrice, par exemple 24. Appuyez sur **INV** **Temp** pour enregistrer la valeur définie dans la mémoire EEPROM comme différence entre la



température réelle et la température mesurée. À partir de là, la différence stockée dans la mémoire EEPROM sera ajoutée aux données mesurées. En règle générale, un écart de mesure de température allant jusqu'à 2° C peut être obtenu, mais uniquement à des températures proches de la température d'étalonnage. À des températures plus éloignées, l'écart peut être plus important. L'étalonnage de la température peut être répété à tout moment en utilisant la même procédure.

### 3B Échange des registres X et Y, x<>y

Libellé : 

Séquence de touches :   



Lors de la saisie d'opérations arithmétiques, l'affichage contient la valeur du registre de travail X. Si le calcul est effectué avec deux registres (par exemple des instructions d'addition), la pile d'opérations arithmétiques contient le deuxième registre, Y. La touche x<>y peut être utilisée pour échanger les registres de l'opération. Cela peut être nécessaire pour les opérations avec un ordre d'opérandes important, comme la division ou la soustraction.



La touche  peut également être utilisée lors de la répétition du dernier calcul effectué avec la touche . Dans ce cas, le deuxième paramètre saisi est stocké dans le registre Y, le premier opérande de l'opération est saisi dans le registre X (données d'affichage).

Exemple:

    [9] ... calcule  $3^2 = 9$

  [25] ... calcule  $5^2 = 25$

    [2] ... stocke 0.5 dans le registre Y

  [3] ... calcule  $9^{0.5} = 3$

### 3C Sinus hyperbolique, sinh

Libellé : **sinh**

Séquence de touches : **2nd** **2nd** **x^2**

Le bouton **sinh** est utilisé pour calculer la fonction sinus hyperbolique.

En ajoutant le préfixe **INV** avant l'instruction **sinh**, l'opération inverse, arcsinus hyperbolique, est appliquée.

Exemple:

**1** **sinh** [1.1752...] ... calcul de  $\sinh(1) = 1.1752...$

**INV** **sinh** [1] ... calcul de  $\operatorname{asinh}(1.1752...) = 1$

### 3D Cosinus hyperbolique, cosh

Libellé : **cosh**

Séquence de touches : **2nd** **2nd** **Vx**

Le bouton **cosh** est utilisé pour calculer la fonction cosinus hyperbolique.

En ajoutant le préfixe **INV** avant l'instruction **cosh**, l'opération inverse, arccosinus hyperbolique, est appliquée.

Exemple:

**1** **cosh** [1.5430...] ... calcul de  $\cosh(1) = 1.5430...$

**INV** **cosh** [1] ... calcul de  $\operatorname{acosh}(1.5430...) = 1$

### 3E Tangente hyperbolique, tanh

Libellé : **tanh**

Séquence de touches : **2nd** **2nd** **1/x**

Le bouton **tanh** est utilisé pour calculer la fonction tangente hyperbolique.

En ajoutant le préfixe **INV** avant l'instruction **tanh**, l'opération inverse, arctangente hyperbolique, est appliquée.

Exemple:

**1** **tanh** [.76159...] ... calcul de  $\tanh(1) = 0.76159\dots$

**INV** **tanh** [1] ... calcul de  $\operatorname{atanh}(0.76159\dots) = 1$

### 40 Adressage indirect, Ind

Libellé : **Ind**

Séquence de touches : **2nd** **Ind**

Le bouton **Ind** est utilisé pour saisir le paramètre indirect d'une opération, où les données requises ne sont pas issues du paramètre de l'instruction, mais du contenu du registre de données mentionné en paramètre. Pour plus d'informations, voir le chapitre Adressage indirect.

### 41 Avance d'un pas, SST

Libellé : **SST**

Séquence de touches : **SST**

Le bouton **SST** (Single Step) incrémente le pointeur de l'adresse du programme de 1 en mode programmation.

En mode exécution, l'instruction courante du programme est exécutée, et le pointeur de l'adresse du programme est incrémenté de 1, ce qui permet ainsi de parcourir le programme pas à pas pour le débogage. Si le déroulement du programme est combiné à l'exécution de sous-routines, le retour correct des sous-routines peut ne pas se produire (la calculatrice ne se souviendra pas de l'adresse de retour de la sous-routine).

Pour plus d'informations, consultez le chapitre Programmation.

## 42 Stockage d'un nombre dans un registre, STO

Libellé : **STO**

Séquence de touches : **STO**

Le bouton **STO** (Store) permet de stocker le nombre affiché dans le registre de données. Le code à deux chiffres du numéro de registre, de 00 à 99, est saisi comme paramètre d'instruction.

En saisissant le code **Ind** après l'instruction **STO**, mais avant de saisir le paramètre, l'instruction **STO** se transforme en instruction **STO Ind** avec le code 72 (voir 72 Stockage indirect d'un nombre dans un registre, STO Ind). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 43 Rappel d'un nombre depuis un registre, RCL

Libellé : **RCL**

Séquence de touches : **RCL**

Le bouton **RCL** (Recall) permet de rappeler un nombre depuis un registre vers le registre X (affichage). Le code à deux chiffres du numéro de registre, de 00 à 99, est saisi comme paramètre d'instruction..

En saisissant le code **Ind** après l'instruction **RCL**, mais avant de saisir le paramètre, l'instruction **RCL** se transforme en instruction **RCL Ind** avec le code 73 (voir 73 Rappel indirect d'un nombre depuis un registre, RCL Ind). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage

indirect.

## 44 Addition et soustraction d'un nombre dans un registre, SUM

Libellé : **SUM**

Séquence de touches : **SUM**

Le bouton **SUM** (Summation) permet d'additionner le nombre affiché dans le registre de données. Le code à deux chiffres du numéro de registre, de 00 à 99, est saisi comme paramètre d'instruction.

En saisissant le code **Ind** après l'instruction **SUM**, mais avant de saisir le paramètre, l'instruction **SUM** se transforme en instruction **SUM Ind** avec le code 74 (voir 74 Addition indirecte d'un nombre dans un registre, SUM Ind). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

En plaçant le préfixe **INV** avant l'instruction **SUM**, la fonction opposée est exécutée : soustraire le nombre du registre de données.

## 45 Puissance et racine, y^x

Libellé : **y^x**

Séquence de touches : **y^x**

L'instruction **y^x** élève le nombre Y (premier opérande, dans la pile d'opérations) à la puissance indiquée par le nombre X (deuxième opérande, nombre sur l'écran). Si le préfixe **INV** est pressé en premier, une opération inverse, la racine, est effectuée.

Le premier opérande Y doit normalement être un nombre non négatif. Ce n'est que si l'exposant X est un entier que le premier opérande peut être un nombre négatif. Dans ce cas, le résultat est négatif si l'exposant était un nombre impair.

Exemple:

**3** **+/-** **y^x** **7** **=** [-2187] ... élévation à la puissance  $(-3)^7 = -2187$

## 46 Insertion d'un pas vide dans le programme, **Ins**

Libellé : **Ins**

Séquence de touches : **2nd** **SST**

La touche **Ins** (Insert), utilisée en mode programmation, insère un pas vide avec la valeur FF à la position courante du programme. Les pas suivants sont déplacés, jusqu'à la fin de la mémoire programme ou jusqu'au prochain octet vide FF. Les adresses absolues ne sont pas recalculées lors du déplacement.

Voir le chapitre Programmation pour plus d'informations.

Notez que certaines instructions vous permettent de saisir un paramètre avec un code HEX et d'insérer la valeur 0FF dans le code. Si possible, évitez une telle valeur d'octet, qui serait interprétée comme un espace vide et remplacée lors du déplacement du programme en mémoire.

## 47 Effacement des registres de données, **CMs**

Libellé : **CMs**

Séquence de touches : **2nd** **STO**

Le bouton **CMs** est utilisé pour effacer le contenu de tous les registres de données (il ne s'applique pas aux registres HIR).

Si le préfixe **INV** précède la fonction **CMs**, seuls les registres de données R01 à R06 (registres utilisés pour les fonctions statistiques), le registre T et le registre X sont effacés.

## 48 Échange d'un nombre avec un registre, Exc

Libellé : **Exc**

Séquence de touches : **2nd RCL**

Le bouton **Exc** (Exchange) permet d'échanger le nombre affiché avec le contenu d'un registre de données. Le code à deux chiffres du numéro de registre, de 00 à 99, est saisi comme paramètre d'instruction.

En saisissant le code **Ind** après l'instruction **Exc**, mais avant de saisir le paramètre, l'instruction **Exc** se transforme en instruction **Exc Ind** avec le code 63 (voir 63 Echange indirect avec le contenu d'un registre, Exc Ind). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 49 Multiplier et diviser dans un registre, Prd

Libellé : **Prd**

Séquence de touches : **2nd SUM**

Le bouton **Prd** (Product) permet de multiplier le contenu du registre de données par le nombre affiché. Le code à deux chiffres du numéro de registre, de 00 à 99, est saisi comme paramètre d'instruction.

En saisissant le code **Ind** après l'instruction **Prd**, mais avant de saisir le paramètre, l'instruction **Prd** se transforme en instruction **Prd Ind** avec le code 64 (voir 64 Multiplication et division indirectes dans un registre, Prd Ind). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

En plaçant le préfixe **INV** avant l'instruction **Prd**, la fonction opposée est exécutée : diviser le contenu du registre de données par le nombre affiché.



## 4A Détection de la tension de la batterie, Bat

Libellé : **Bat**

Séquence de touches : **2nd** **2nd** **SST**

Le bouton **Bat** permet de déterminer la tension de la batterie. La tension est affichée en volts. L'état de décharge de la batterie peut être évalué à partir de la tension de la batterie. La pile CR2032 neuve a une tension d'environ 3 V. La calculatrice est capable de fonctionner à partir d'une tension minimale d'environ 2,6 V. Lorsqu'elle est alimentée par le connecteur USB, la tension d'alimentation indiquée sera d'environ 2,9 V (un stabilisateur est utilisé pour réduire la tension).

La saisie du préfixe **INV** avant d'appuyer sur **Bat** affichera la tension de la batterie sous forme de pourcentage. La valeur de 0 à 100 % correspond à une tension de batterie de 2,4 V à 2,9 V.

La précision des données de mesure de la tension de la batterie n'est pas garantie, elles sont fournies à titre indicatif uniquement.

## 4B Factorielle, x!

Libellé : **x!**

Séquence de touches : **2nd** **2nd** **STO**

Le bouton **x!** permet de calculer la factorielle. La factorielle est un nombre créé en multipliant les valeurs 1, 2, 3, ... jusqu'au nombre d'entrée x saisi. La calculatrice calcule la factorielle à l'aide de la fonction d'approximation (fonction Gamma de Stieltjes). Cela permet des calculs rapides et précis, y compris des facteurs décimaux.

Le nombre saisi ne doit pas être négatif. La valeur maximale de la factorielle que la calculatrice peut calculer est 3208 (résultat 8,61680144+9856).

En saisissant le préfixe **INV** avant d'appuyer sur **x!**, une multiplication entière répétée est utilisée au lieu de calculer la fonction d'approximation.

L'entrée peut être un entier compris entre 0 et 3209. Cette méthode n'est ni plus précise ni plus rapide que le calcul utilisant la fonction d'approximation, elle sert uniquement de valeur de référence.

Exemple:

$\boxed{6} \boxed{x!}$  [720] ... factorielle de  $6! = 1*2*3*4*5*6 = 720$

$\boxed{INV} \boxed{6} \boxed{x!}$  [720] ... factorielle par multiplication  $6! = 720$

$\boxed{6} \boxed{.} \boxed{1} \boxed{x!}$  [868.9568...] ... factorielle décimale de  $6.1! = 868.9568...$

## 4C Logarithme naturel d'une factorielle, $\ln x!$

Libellé :  $\boxed{\ln x!}$

Séquence de touches :  $\boxed{2nd} \boxed{2nd} \boxed{RCL}$

Le bouton  $\boxed{\ln x!}$  calcule le logarithme naturel de la factorielle. La valeur d'entrée est un nombre non négatif, y compris les nombres décimaux. Contrairement à la fonction de calcul de la factorielle  $\boxed{n!}$ , cette fonction n'est pas limitée par la plage de la calculatrice.

Exemple:

$\boxed{6} \boxed{\ln x!}$  [6.57925...] ... logarithme naturel  $\ln 6! = 6.57925...$

$\boxed{6} \boxed{n!} \boxed{\ln x}$  [6.57925...] ... contrôle du calcul

## 4D Logarithme décimal d'une factorielle, $\log x!$

Libellé :  $\boxed{\log x!}$

Séquence de touches :  $\boxed{2nd} \boxed{2nd} \boxed{SUM}$

Le bouton  $\boxed{\log x!}$  calcule le logarithme décimal de la factorielle. La valeur d'entrée est un nombre non négatif, y compris les nombres décimaux. Contrairement à la fonction de calcul de la factorielle  $\boxed{n!}$ , cette fonction n'est pas limitée par la plage de la calculatrice. La fonction peut également

être utilisée pour calculer sur de très grandes factorielles.

Exemple 1:

**6** **log x!** [2.85733...] ... logarithm décimal  $\log 6! = 2.85733...$

**6** **n!** **log x** [2.85733...] ... contrôle du calcul

Exemple 2, 123456789!:

**1** **2** **3** **4** **5** **6** **7** **8** **9** **log x!** [945335859.45538] ... logarithme factorielle

**INV** **Int** **INV** **log** [2.8535...] ... exposant partie décimale

... soit  $123456789! = 2.85351252... * 10^{945335859}$

*Remarque : selon la calculatrice, le résultat est  $123456789! = 2,8535125217299 * 10^{945335859}$ . La valeur correcte doit être  $2,8535125219128 * 10^{945335859}$ . La mantisse du résultat correspond à 10 chiffres. Ceci est donné par l'utilisation du logarithme. La précision interne de la calculatrice est de 19 chiffres. Avec une telle précision, la calculatrice calcule le logarithme de la factorielle. La partie entière du logarithme représente 9 chiffres de l'exposant ( $10^{945335859}$ ). Après avoir supprimé 9 chiffres de la partie entière, il reste une mantisse avec une précision de 10 chiffres, ce qui correspond à la précision réalisable de la mantisse résultante. Lors de la division du logarithme en une partie de l'exposant et de la mantisse, il faut prendre en compte la réduction de la précision de la mantisse du résultat.*

## 4E Modulo (arrondi inférieur), mod2

Libellé : **mod2**

Séquence de touches : **2nd** **2nd** **y^x**

L'opération modulo divise le premier opérande Y (dans la pile) par le deuxième opérande X (sur l'écran), convertit le résultat en un entier, multiplie le deuxième opérande X par celui-ci et le soustrait du premier opérande Y. Le résultat est le reste après division.

L'instruction **mod2** est similaire à l'instruction **mod** (voir 5E Modulo (troncature), mod) et donne le même résultat pour les nombres positifs. La différence se reflète dans les nombres négatifs. L'opération **mod2** utilise la fonction *floor* pour arrondir le résultat, c'est-à-dire l'arrondir vers le bas. Le résultat a le même signe que le deuxième opérande (contrairement à la fonction **mod**, qui conserve le signe du premier opérande).

L'instruction **mod2** peut être utilisée, par exemple, pour normaliser un angle dans la plage de 0 à 359°, car elle traite également correctement les angles négatifs.

Exemple:

$$2 \div 2 \text{ mod2 } 0 \div 5 = [0.2] \dots 2.2 \text{ mod2 } 0.5 = 0.2$$

$$2 \div 2 \text{ +/- mod2 } 0 \div 5 = [0.3] \dots -2.2 \text{ mod2 } 0.5 = 0.3$$

$$2 \div 2 \text{ mod2 } 0 \div 5 \text{ +/-} = [-0.3] \dots 2.2 \text{ mod2 } -0.5 = -0.3$$

$$2 \div 2 \text{ +/- mod2 } 0 \div 5 \text{ +/-} = [-0.2] \dots -2.2 \text{ mod2 } -0.5 = -0.2$$

## 50 Valeur absolue, |x|

Libellé : **|x|**

Séquence de touches : **2nd** **|**

La fonction **|x|** ajuste le nombre affiché (registre X) à sa valeur absolue (supprime le signe négatif du nombre).

## 51 Recul d'un pas, BST

Libellé : **BST**

Séquence de touches : **BST**

Utilisé en mode programmation, le bouton **BST** (Back Step) décrémente le pointeur d'adresse du programme de 1 pas.

Pour plus d'informations, voir le chapitre Programmation.

## 52 Mode exposant (notation scientifique), EE

Libellé : **EE**

Séquence de touches : **EE**

L'appui sur la touche **EE** active le mode exposant. Si la touche est enfoncée pendant la saisie d'un nombre, l'exposant sera saisi. En même temps, le mode d'affichage en notation scientifique avec l'exposant est activé.

Si la touche est enfoncée en dehors de l'édition du nombre, le mode d'affichage en notation scientifique avec l'exposant est activé et l'édition de l'exposant du nombre est lancée. Cette fonction est souvent utilisée pour supprimer les chiffres cachés du nombre, car lorsque vous commencez l'édition, seuls les chiffres affichés sont chargés dans l'éditeur, pas la valeur exacte complète du nombre.

Appuyer sur le préfixe **INV** avant d'appuyer sur **EE** permet de quitter le mode d'affichage de l'exposant. Une autre façon de quitter le mode d'affichage de l'exposant est d'appuyer sur la touche **CLR**.

Exemple:

**pi** **Fix** **4** **EE** **INV** **EE** **INV** **Fix** [3.1416] ... arrondi à 4 décimales


## 53 Parenthèse gauche, (


Libellé : **(**


Séquence de touches : **(**

Bouton **(** ouvre le calcul pour une partie de l'expression.


## 54 Parenthèse droite, )


Libellé : 



Séquence de touches : 

Le bouton  termine le calcul d'une partie de l'expression.


## 55 Division, ÷



Libellé : 

Séquence de touches : 

Le bouton  divise le premier opérande par le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur .

## 56 Suppression d'un pas du programme, Del


Libellé : 

Séquence de touches :  

La touche Del (Delete) utilisée en mode programmation supprime le pas de la position courante du programme. Les données suivantes sont décalées en remontant les pas jusqu'à la fin de la mémoire programme ou jusqu'au prochain pas FF vide. Les adresses absolues ne sont pas recalculées lors du déplacement.

Voir le chapitre Programmation pour plus d'informations.

## 57 Mode exposant (notation ingénieur), Eng

Libellé : 

Séquence de touches : **2nd** **EE**

Le bouton **Eng** (Engineer) permet d'activer le mode exposant (ingénieur) d'affichage d'un nombre. Le nombre est affiché avec un exposant multiple de 3 (mode d'affichage décimal et octal) ou de 4 (mode d'affichage hexadécimal et binaire).

Le mode exposant **Eng** ne se désactive pas en appuyant sur **CLR**. Pour le désactiver, il faut utiliser la séquence **INV** **Eng**.

Exemple:

**5** **|** ... prenons une tension de 5 V

**2** **EE** **3** **+/-** ... divisons la tension par 2 mA

**=** **Eng** [2.5+3] ... la valeur de la résistance sera de 2,5 kOhm

## 58 Arrondi, Fix

Libellé : **Fix**

Séquence de touches : **2nd** **|**

La touche **Fix** sert à arrondir le nombre affiché à l'écran au nombre de décimales spécifié. Un nombre représentant le nombre de décimales après la virgule est saisi comme paramètre. Le chiffre peut être compris entre 0 et 9 (représentant 0 à 9 décimales), mais également le chiffre hexadécimal 0A à 0D, ce qui signifie 10 à 13 décimales.

En mode arrondi, le nombre est complété vers la droite avec des zéros, jusqu'au nombre de décimales spécifié. La saisie de **INV** **Fix** ou **Fix** **0F** désactive l'arrondi. Dans ce cas, le nombre est affiché en pleine précision et les zéros non significatifs de fin sont supprimés.

La calculatrice TI-58/59 d'origine utilisait la séquence **Fix** **9** pour désactiver l'arrondi. Ceci est considéré comme un réglage 9 décimales valide pour l'ET-58. Lors de l'importation d'un programme depuis la TI-58/59, il est nécessaire d'effectuer une correction et de remplacer **Fix** **9**

par le code **INV** **Fix** ou **Fix** **OF**.

L'arrondi n'affecte que l'affichage du nombre. En interne, le nombre est toujours calculé avec une précision totale. Si des chiffres cachés doivent vraiment être supprimés, cela peut être fait avec la touche **EE** (voir 52 Mode exposant, EE).

Le mode d'arrondi défini affecte également la manière d'afficher les très petits nombres. Si l'arrondi est activé et que le mode exposant n'est pas activé, l'écran affichera des zéros pour les petits nombres, même si des chiffres valides ont dépassé la bordure droite de l'écran. Si l'arrondi n'est pas activé, la calculatrice passe à l'affichage de l'exposant si l'exposant est inférieur à -3.

## 59 Entier, Int

Libellé : **Int**

Séquence de touches : **2nd** **Int**

Le bouton **Int** (Integer) peut être utilisé pour supprimer les chiffres après la virgule décimale, c'est-à-dire pour réduire le nombre à un entier. La fonction a la même signification que l'arrondi à zéro.

Si le préfixe **INV** est utilisé avant la commande **Int**, la fonction inverse est exécutée - en supprimant la partie entière du nombre et en laissant la partie décimale (fraction).

Exemple:

$$\mathbf{2} \mathbf{|} \mathbf{3} \mathbf{|} \mathbf{Int} [2] \dots \text{int}(2.3) = 2$$

$$\mathbf{2} \mathbf{|} \mathbf{3} \mathbf{|} \mathbf{+/-} \mathbf{|} \mathbf{Int} [-2] \dots \text{int}(-2.3) = -2$$

$$\mathbf{2} \mathbf{|} \mathbf{3} \mathbf{|} \mathbf{INV} \mathbf{|} \mathbf{Int} [0.3] \dots \text{frac}(2.3) = 0.3$$

$$\mathbf{2} \mathbf{|} \mathbf{3} \mathbf{|} \mathbf{+/-} \mathbf{|} \mathbf{INV} \mathbf{|} \mathbf{Int} [-0.3] \dots \text{frac}(-2.3) = -0.3$$



## 5A Réglage du contraste de l'affichage, LCD

Libellé : **LCD**

Séquence de touches : **2nd** **2nd** **BST**

Le bouton **LCD** sert à régler le contraste de l'écran. Après avoir appuyé sur le bouton, l'instruction vous demandera d'entrer le chiffre de 0 à 9. Le chiffre 0 représente le contraste minimum (les caractères sont gris ou à peine visibles), le chiffre 9 est le contraste maximum (il y a des rectangles sombres sous les caractères).

Le contraste de l'écran LCD dépend de la tension d'alimentation. En réglant le contraste, il est possible de sortir de la plage visible. Dans ce cas, il peut être nécessaire de mémoriser le contraste de l'écran. Si aucun caractère n'est visible sur l'écran (la calculatrice semble éteinte), essayez de régler un contraste d'affichage plus élevé à partir du clavier en utilisant un chiffre plus élevé. Alternativement, appuyez d'abord sur CLR, peut-être que la calculatrice est éteinte. Inversement, s'il y a des rectangles noirs sur l'écran, définissez une valeur inférieure pour le contraste de l'écran.

En saisissant le préfixe **INV** avant l'instruction **LCD**, la valeur actuellement définie du contraste de l'écran peut être trouvée (la valeur 0 à 9 est renvoyée dans le registre X).

## 5B Décaler vers la gauche, SHL

Libellé : **SHL**

Séquence de touches : **2nd** **2nd** **EE**

La fonction **SHL** permet de déplacer le premier argument Y (dans la pile d'opérations) vers la gauche du nombre de bits donné par le deuxième argument X (nombre sur l'écran). L'opération de décalage vers la gauche de 1 bit correspond à l'opération de multiplication par le nombre 2.

Exemple:

**1** **2** **3** **SHL** **4** **=** [1968] ...  $123 \ll 4 = 123 * 2^4 = 1968$

ou en code HEX :  $123 \ll 4 = 0x7B \ll 4 = 0x7B0 = 1968$

ou en code BIN :  $123 = 1111011$  décalé de  $\ll 4 = 11110110000 = 1968$

## 5C Décaler vers la droite, SHR

Libellé : **SHR**

Séquence de touches : **2nd** **2nd** **|**

La touche **SHR** permet de déplacer le premier argument Y (dans la pile d'opérations) vers la droite du nombre de bits donné par le deuxième argument X (numéro sur l'écran). L'opération de décalage vers la droite de 1 bit correspond à l'opération de division par le nombre 2.

Exemple:

**1** **2** **3** **SHR** **4** **=** [7.6875] ...  $123 \gg 4 = 123 / 2^4 = 7.6875$

ou en code HEX :  $123 \gg 4 = 0x7B \gg 4 = 0x7.B = 7.6875$

ou en code BIN :  $123 = 1111011$  décalé de  $\gg 4 = 111,1011 = 7.6875$

## 5D Arrondi, round

Libellé : **round**

Séquence de touches : **2nd** **2nd** **|**

Le bouton **round** arrondit le nombre affiché au nombre entier le plus proche. Si la partie décimale est supérieure ou égale à 0,5, le nombre est arrondi au nombre supérieur. Si la partie décimale est inférieure à 0,5, le nombre est arrondi au nombre inférieur.

Contrairement à la fonction **Fix**, l'arrondi est effectué avec un nombre réel,

pas seulement pour l'affichage.

En plaçant le préfixe **INV** avant la fonction **round**, la partie entière du nombre est supprimée en arrondissant le nombre vers le bas (plancher). Cette opération donnera un nombre qui sera toujours positif et sera compris entre 0 (inclus) et 1 (exclu). Pour les nombres positifs, le résultat est le même que la fonction **INV Int** (voir 59 Entier, Int). Pour les nombres négatifs, 1 est ajouté au résultat différent de zéro.

Exemple:

$$\boxed{2} \boxed{.} \boxed{3} \boxed{\text{round}} [2] \dots \text{round}(2.3) = 2$$

$$\boxed{2} \boxed{.} \boxed{5} \boxed{\text{round}} [3] \dots \text{round}(2.5) = 3$$

$$\boxed{2} \boxed{.} \boxed{3} \boxed{+/-} \boxed{\text{round}} [-2] \dots \text{round}(-2.3) = -2$$

$$\boxed{2} \boxed{.} \boxed{5} \boxed{+/-} \boxed{\text{round}} [-3] \dots \text{round}(-2.5) = -3$$

$$\boxed{2} \boxed{.} \boxed{6} \boxed{+/-} \boxed{\text{round}} [-3] \dots \text{round}(-2.6) = -3$$

$$\boxed{2} \boxed{.} \boxed{3} \boxed{\text{INV}} \boxed{\text{round}} [0.3] \dots 2.3 - \text{floor}(2.3) = 2.3 - 2 = 0.3$$

$$\boxed{2} \boxed{.} \boxed{6} \boxed{\text{INV}} \boxed{\text{round}} [0.6] \dots 2.6 - \text{floor}(2.6) = 2.6 - 2 = 0.6$$

$$\boxed{2} \boxed{.} \boxed{3} \boxed{+/-} \boxed{\text{INV}} \boxed{\text{round}} [0.7] \dots -2.3 - \text{floor}(-2.3) = -2.3 - (-3) = 0.7$$

$$\boxed{2} \boxed{.} \boxed{6} \boxed{+/-} \boxed{\text{INV}} \boxed{\text{round}} [0.4] \dots -2.6 - \text{floor}(-2.6) = -2.6 - (-3) = 0.4$$

## 5E Modulo (troncature), mod

Libellé : **mod**

Séquence de touches : **2nd** **2nd** **:**

L'opération **mod** (modulo) divise le premier opérande Y (dans la pile) par le deuxième opérande X (sur l'écran), convertit le résultat en un entier, multiplie le deuxième opérande X par celui-ci et le soustrait du premier opérande Y. Le résultat est le reste après la division.

L'instruction **mod** est similaire à l'instruction **mod2** (voir 4E Modulo floor, mod2) et donne le même résultat pour les nombres positifs. La différence

se reflète dans les nombres négatifs. L'opération **mod** utilise la troncature pour arrondir le résultat. Le résultat a le même signe que le premier opérande (contrairement à la fonction **mod2** qui conserve le signe du deuxième opérande).

Exemple:

$$2 \lfloor 2 \bmod 0.5 = [0.2] \dots 2.2 \bmod 0.5 = 0.2$$

$$2 \lfloor 2 +/- \bmod 0.5 = [-0.2] \dots -2.2 \bmod 0.5 = -0.2$$

$$2 \lfloor 2 \bmod 0.5 +/- = [0.2] \dots 2.2 \bmod -0.5 = 0.2$$

$$2 \lfloor 2 +/- \bmod 0.5 +/- = [-0.2] \dots -2.2 \bmod -0.5 = -0.2$$

## 60 Degrés, Deg

Libellé : **Deg**

Séquence de touches : **2nd** **x**

Le bouton **Deg** bascule les calculs des fonctions trigonométriques en degrés (l'angle total est de 360°).

Si les calculs doivent être effectués indépendamment de la mesure angulaire sélectionnée, les fonctions **Op 72** et **Op 73** peuvent être utilisées pour les conversions.

## 61 Saut, GTO

Libellé : **GTO**

Séquence de touches : **GTO**

Le bouton **GTO** (Go To) permet d'effectuer un saut inconditionnel dans un programme. Le paramètre est saisi soit avec le code numérique à 3 chiffres de l'adresse de destination absolue, soit avec l'instruction utilisée comme étiquette dans le programme.

En utilisant la touche **Ind** après l'instruction **GTO**, mais avant de saisir l'adresse de saut, l'instruction se transforme en une instruction avec adressage indirect, **GTO Ind** (code d'instruction 83, voir 83 Saut indirect, GTO Ind).

Si l'instruction **GTO** est utilisée en mode exécution, le pointeur du programme est placé sur l'adresse sélectionnée. Ce saut en mode exécution peut être utilisé pour déplacer le pointeur pendant la programmation (voir Programmation pour plus de détails).

Pour plus d'informations sur les méthodes d'adressage, voir le chapitre Adressage indirect.

Le bouton **GTO** a une autre fonction spéciale. Si vous maintenez le bouton **GTO** enfoncé pendant l'exécution du programme, le contenu courant de l'affichage (contenu du registre X) clignotera sur la ligne inférieure de l'affichage et l'adresse courante du programme avec la commande en cours d'exécution clignotera sur la ligne supérieure. Cependant, cette surveillance ralentit le programme considérablement.

## 62 Sélection indirecte programme de bibliothèque, Pgm Ind

Libellé : **Pgm Ind**

Séquence de touches : **2nd LRN 2nd y^x**

L'instruction **Pgm Ind** permet de sélectionner un programme de bibliothèque de la même manière que l'instruction **Pgm** (voir 36 Sélection d'un programme de bibliothèque, Pgm), mais au lieu de le faire à partir du paramètre d'instruction, le numéro de bibliothèque est extrait du registre de données.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **Pgm**, mais avant d'entrer le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, consultez le chapitre Adressage indirect.

Exemple:

**2 STO 01** ... le numéro de programme 02 est stocké dans le registre R01

**Pgm Ind 01** [ML-02 (875)] ... le programme de bibliothèque 02 est activaté

## 63 Echange indirect avec le contenu d'un registre, Exc Ind

Libellé : **Exc Ind**

Séquence de touches : **2nd RCL 2nd y^x**

L'instruction **Exc Ind** peut être utilisée pour échanger le numéro sur l'écran avec le registre de données de la même manière que l'instruction **Exc** (voir 48 Échange de numéro avec le registre de données, Exc), mais au lieu du paramètre d'instruction, le numéro de registre est extrait du registre de données.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **Exc**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 64 Multiplication et division indirectes dans un registre, Prd Ind

Libellé : **Prd Ind**

Séquence de touches : **2nd SUM 2nd y^x**

L'instruction **Prd Ind** peut être utilisée pour multiplier ou diviser le contenu du registre de données par le nombre affiché de la même manière que l'instruction **Prd** (voir 49 Multiplier et diviser le registre de données, Prd), mais au lieu du paramètre d'instruction, le numéro de registre est extrait du registre de données.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **Prd**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre

Adressage indirect.

## 65 Multiplication, x

Libellé : **x**

Séquence de touches : **x**

La touche **x** multiplie le premier opérande par le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur **=**.

## 66 Délai d'attente, Pause

Libellé : **Pause**

Séquence de touches : **2nd GTO**

La commande **Pause** spécifiée dans le programme suspend l'exécution du programme pendant 0,25 seconde et affiche le contenu courant de l'écran (contenu du registre X).

Voir aussi la surveillance de l'exécution du programme avec la touche **GTO** (saut GTO 61).

Les commandes **Op 80** et **Op 81** permettent de suspendre brièvement le programme sans afficher le contenu de l'écran.

## 67 Test d'égalité registres x et t, x=t

Libellé : **x=t**

Séquence de touches : **2nd 7**

L'instruction **x=t** permet de comparer le registre X (contenu affiché) avec le registre auxiliaire T (défini par le bouton **x<>t**). Si les registres sont identiques, un saut est effectué vers l'adresse spécifiée en paramètre de

l'instruction. L'adresse peut être une adresse absolue ou une étiquette.

L'instruction **x=t** n'a pas de code spécial pour l'adressage indirect. Néanmoins, un adressage indirect est possible de telle sorte que le code **Ind** soit stocké dans le programme après le code **x=t**. Si la condition est remplie, l'adresse absolue ou le code d'étiquette est lu à partir du registre saisi, comme décrit plus en détail dans le chapitre Adressage indirect.

Si le préfixe **INV** est spécifié avant le code **x=t**, la fonction inverse est effectuée - le saut vers l'adresse spécifiée est effectué au contraire en cas de non-concordance de registre.

Lors du test d'accord, l'accord absolu de la mantisse entière n'est pas testé, car les nombres peuvent différer légèrement en raison des calculs. Il est calculé avec une petite tolérance autorisée. Après un grand nombre d'opérations, l'écart peut sortir de la tolérance autorisée et la conformité ne sera pas évaluée correctement. Un cas typique est la soustraction répétée d'un nombre entier. Si possible, il est recommandé d'arrondir les nombres comparés ou de tester dans un intervalle avec une tolérance plus grande.

#### Exemple adressage indirect:

**RST** **LRN** ... active le mode de programmation

**Lbl** **A** **x=t** **Ind** **01** **CLR** **RTN** ... s'il y a une correspondance, il passe à l'adresse contenue dans R01, sinon renvoie la valeur 0

**1** **RTN** ... si branchement à l'adresse 7 renvoie une valeur de 1

**LRN** ... retour au mode exécution

**7** **STO** **01** ... stockage de l'adresse de saut '7' dans le registre R01

**5** **x/t** **2** **A** [0] ... les nombres 2 et 5 ne correspondent pas, ce qui donne une valeur de 0

**5** **A** [1] ... les nombres 5 et 5 correspondent, la valeur 1 est renvoyée



## 68 Pas d'opération, Nop

Libellé : **Nop**

Séquence de touches : **2nd 8**

La commande **Nop** (No Operation) est une commande vide qui n'effectue aucune opération. Il sert uniquement à remplir l'emplacement d'un pas inutilisé dans le programme. De nombreux autres codes de programme ont une fonction "nulle", par exemple, les codes de A0 à FE, utilisés seuls. Mais ils peuvent servir d'étiquette de programme.

## 69 Opérations spéciales, Op

Libellé : **Op**

Séquence de touches : **2nd 9**

L'instruction **Op** assure l'exécution d'une commande spéciale de la calculatrice. L'instruction est suivie d'un code paramètre de 1 octet.

L'ajout du code **Ind** après le code **Op**, mais avant de spécifier le paramètre, transforme l'instruction en instruction d'adressage indirect, **Op Ind**, avec le code 84 (voir 84 Opération spéciale indirecte Op Ind). Dans le cas d'un adressage indirect, le paramètre d'instruction n'est pas lu dans le programme, mais dans le registre de données dont le numéro est spécifié comme paramètre d'opération.

L'instruction **Op** est une instruction volumineuse comportant de nombreuses opérations, elle est donc couverte dans un chapitre séparé, "Opérations spéciales Op".

## 6A Saut relatif, REL

Libellé : **REL**

Séquence de touches : **2nd** **2nd** **GTO**

L'instruction **REL** effectue un saut relatif en fonction de la valeur du paramètre suivant. Le paramètre de l'instruction est un nombre décimal de 00 à 99. La valeur du paramètre est ajoutée à l'adresse suivant l'instruction **REL** et un saut vers l'adresse donnée est effectué. Par exemple, le paramètre 01 signifie sauter 1 pas à partir du pas suivant, le paramètre 00 signifie continuer le programme sans faire de saut.

Si le préfixe **INV** est utilisé avant l'instruction **REL**, un saut vers l'arrière est effectué. La valeur du paramètre est soustraite de l'adresse suite à l'instruction **REL**. Par exemple le paramètre 03 signifie revenir au début de l'instruction **INV REL** 03, le paramètre 00 signifie continuer le programme sans effectuer de saut.

Le saut relatif **REL** fonctionne de la même manière que l'adressage absolu, c'est-à-dire un saut rapide est effectué sans avoir besoin d'utiliser une étiquette. Dans le même temps, l'adresse de saut est indépendante de la position en mémoire, le code peut être déplacé arbitrairement dans le programme. La commande **REL** est utile pour les sauts courts dans un programme.

Exemple:

**RST** **LRN** ... active le mode de programmation

**Lbl** **A** **+** **1** **=** **Pause** **INV** **REL** **0** **7**

**LRN** ... revient au mode exécution

- Après avoir appuyé sur **A**, le programme fait un cycle et augmente à plusieurs reprises le nombre affiché sur l'écran.

## 6B Incrémentation indirecte dans un registre, Inc Ind

Libellé : **Inc Ind**

Séquence de touches : **2nd 2nd | 2nd y^x**

L'instruction **Inc Ind** incrémente/décrémente le contenu du registre de données de la même manière que l'instruction **Inc** (voir 9C Incrémentation et décrémentation du registre Inc), seulement au lieu du paramètre d'instruction, le numéro de registre est extrait du registre de données mentionné en paramètre.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **Inc**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. De plus amples informations sur l'adressage indirect sont disponibles dans le chapitre Adressage indirect.

## 6C Opérations indirectes de registre à registre, Reg Ind

Libellé : **Reg Ind**

Séquence de touches : **2nd 2nd RST nn 2nd y^x**

L'instruction **Reg Ind** effectue une opération entre les registres de la même manière que l'instruction **Reg** (voir 8A Opérations de registre à registre Reg), seulement au lieu du deuxième paramètre de l'instruction, le numéro du registre source est extrait du registre de données mentionné en paramètre. Le registre cible n'est pas modifié par le code **Ind**, il est déterminé par le premier paramètre de l'instruction **Reg**.

L'instruction est créée en appuyant sur la touche **Ind** après le premier paramètre de l'instruction **Reg**, mais avant de saisir le deuxième paramètre (remarque : **Ind** ne peut pas être saisi immédiatement après l'instruction **Reg**). Le premier paramètre spécifie l'opérande cible et le code d'opération et est le même pour l'instruction **Reg** que pour l'instruction **Reg Ind**. Le deuxième opérande de l'instruction **Reg** est le numéro du registre utilisé comme opérande source. Pour l'instruction **Reg**

**Ind**, le deuxième paramètre est le numéro de registre contenant le numéro de registre de l'opérande source. De plus amples informations sur l'adressage indirect sont disponibles dans le chapitre Adressage indirect.

## 6D Condition indirecte, IF Ind

Libellé : **IF Ind**

Séquence de touches : **INV 2nd 2nd SBR**

L'instruction **IF Ind** effectue un saut conditionnel tout comme l'instruction **IF** (voir 7A Saut conditionnel IF), seulement au lieu de comparer directement les registres de données, les indices des registres sont extraits du code de l'instruction, leur contenu est lu et utilisé comme registre de données. indices de comparaison. L'adressage indirect s'applique aux deux opérandes, premier et deuxième. L'un des registres ne peut être adressé directement et l'autre indirectement. L'adresse de saut peut être indirecte - dans ce cas, le code **Ind** est utilisé avant de saisir l'adresse de saut (avant le 3ème opérande de l'instruction **IF**).

L'instruction **IF Ind** est créée dans une procédure quelque peu non standard par rapport aux autres instructions indirectes (car le code **Ind** a ici la signification d'un paramètre valide). Tout d'abord, on appuie sur le préfixe **INV**, puis sur la touche **IF**. Alors le code **IF Ind** est stocké à la place du code **IF**. Les paramètres suivants sont saisis de la même manière que pour l'instruction **IF**. De plus amples informations sur l'adressage indirect sont disponibles dans le chapitre Adressage indirect.

## 6E ET entre deux opérandes binaires, AND

Libellé : **AND**

Séquence de touches : **2nd 2nd x**

L'instruction **AND** effectue une opération ET au niveau du bit (produit au niveau du bit) entre le premier et le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre

premier opérande en appuyant plusieurs fois sur  $\boxed{=}$ .

Un produit au niveau du bit signifie que le résultat de l'opération est un bit «1» uniquement si les deux bits comparés sont «1».

Exemple:

$\boxed{BIN}$  ... passage à l'affichage en mode binaire

$\boxed{1}\boxed{0}\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{1}\boxed{1}$  ... saisie du premier opérande (10110011 = 179 décimal)

$\boxed{AND}$  ... le produit au niveau du bit est effectué

$\boxed{0}\boxed{0}\boxed{1}\boxed{0}\boxed{0}\boxed{1}\boxed{1}\boxed{0}$  ... saisie du deuxième opérande (100110 = 38 décimal)

$\boxed{=}$  [100010] ... calcul du résultat (100010 = 34 décimal)

$$\begin{array}{r} 10110011 \\ \text{AND } 100110 \\ \hline = 100010 \end{array}$$

## 70 Radians, Rad

Libellé :  $\boxed{Rad}$

Séquence de touches :  $\boxed{2nd}\boxed{}$

Le bouton  $\boxed{Rad}$  bascule les calculs des fonctions trigonométriques en radians (un angle total est  $2 \cdot \pi = 6,283185\dots$ ).

Si les calculs doivent être effectués indépendamment de la mesure angulaire sélectionnée, les fonctions  $\boxed{Op}\boxed{72}$  et  $\boxed{Op}\boxed{73}$  peuvent être utilisées pour les conversions.

## 71 Appel sous-programme, SBR

Libellé : **SBR**

Séquence de touches : **SBR**

Le bouton **SBR** (Subroutine) permet d'appeler un sous-programme. Le paramètre suivant peut être soit une adresse absolue de destination (adresse de pas numérique sur 3 chiffres), soit une étiquette du programme (code d'une touche). Si l'instruction **SBR** est utilisée en mode exécution, le sous-programme est exécuté immédiatement.

Lors de l'appel d'un sous-programme, l'adresse qui suit le code d'instruction **SBR** est d'abord stockée dans la pile d'adresses. Le contrôle est ensuite transmis au sous-programme. La pile d'adresses a une capacité limitée à 15 sous-programmes.

Le sous-programme se termine par l'instruction **RTN** (voir 92 Retour du sous-programme RTN). L'instruction **RTN** est invoquée en appuyant sur les touches **INV SBR** et assure un retour du sous-programme. L'adresse d'origine après l'instruction **SBR** est extraite de la pile d'adresses et transmise à cette adresse de contrôle. Si le sous-programme a été lancé à partir du clavier, la calculatrice s'arrête.

Les sous-programmes peuvent également être appelés depuis un autre programme de bibliothèque. Une instruction **Pgm** suivie du numéro de programme et du code du bouton **A** à **F** ou en appelant le sous-programme via **SBR** est répertoriée dans le programme. Si l'instruction Pgm est utilisée dans un programme, le programme actif n'est pas commuté de manière permanente (comme ce serait le cas lors de son utilisation à partir du clavier), le changement n'est que temporaire pour la durée d'un appel de sous-programme ultérieur. Une fois le sous-programme terminé, le contrôle est transféré au programme d'origine.

En utilisant la touche **Ind** après le code **SBR**, mais avant de spécifier l'adresse du sous-programme, change l'instruction en une instruction d'adressage indirect, **SBR Ind** (voir 26 Sous-programme d'adressage indirect SBR). Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 72 Stockage indirect d'un nombre dans un registre, STO Ind

Libellé : **STO** **Ind**

Séquence de touches : **STO** **2nd** **y<sup>x</sup>**

L'instruction **STO** **Ind** stocke le contenu de l'affichage dans le registre de données de la même manière qu'avec l'instruction **STO** (voir 42 Stockage d'un nombre dans un registre STO), seulement le paramètre de l'instruction est le numéro de registre qui contient le numéro de registre destinataire.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **STO**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 73 Rappel indirect d'un nombre depuis un registre, RCL Ind

Libellé : **RCL** **Ind**

Séquence de touches : **RCL** **2nd** **y<sup>x</sup>**

L'instruction **RCL** **Ind** rappelle un nombre depuis le registre de données de la même manière qu'avec l'instruction **RCL** (voir 43 Rappel d'un nombre depuis un registre RCL), seulement le paramètre de l'instruction est le numéro de registre qui contient le numéro de registre d'origine.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **RCL**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 74 Addition indirecte d'un nombre dans un registre, SUM Ind

Libellé : **SUM** **Ind**

Séquence de touches : **SUM** **2nd** **y<sup>x</sup>**

L'instruction **SUM Ind** ajoute (ou soustrait avec **INV**) un nombre au registre de données de la même manière qu'avec l'instruction **SUM** (voir 44 Addition et soustraction d'un nombre dans un registre SUM), seulement le paramètre de l'instruction est le numéro de registre qui contient le numéro de registre destinataire.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **SUM**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 75 Soustraction, -

Libellé : **-**

Séquence de touches : **-**

Le bouton **-** soustrait le deuxième opérande du premier opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur **=**.

## 76 Etiquette dans un programme, Lbl

Libellé : **Lbl**

Séquence de touches : **2nd SBR**

L'instruction **Lbl** peut être utilisée pour marquer un endroit dans le programme sous forme d'étiquette. Le code **Lbl** est suivi en paramètre d'un code d'une instruction utilisé comme label. N'importe quelle touche peut être utilisée sauf **2nd**, sauf les chiffres **0** à **F** et sauf **Ind**.

L'emplacement du programme indiqué par l'étiquette peut être accédé à l'aide d'instructions de saut avec l'adresse donnée, telles que **GTO**, **SBR**, **Dsz** et autres... Dans la calculatrice TI-58/59 d'origine, l'adressage absolu était préféré pour des raisons de vitesse, car un saut d'adresse absolue est effectué immédiatement, tandis qu'une recherche d'étiquette dans le programme peut prendre un certain temps. Pour la calculatrice ET-58, il est



préférable d'utiliser une balise car sa recherche est rapide et il est préférable de pouvoir déplacer facilement le code (passer à une autre adresse) sans avoir à corriger les adresses de saut absolues.

Si la saisie d'une instruction comme étiquette ne convient pas, les codes d'étiquette peuvent être saisis numériquement, à l'aide de la fonction code. 2 chiffres HEX sont saisis comme paramètre de code. Le tag peut avoir un code de 10 à FE (sauf 40 = **Ind**). Les codes A0 à FE, qui ne sont pas utilisés par la calculatrice et ont le sens d'opérations vides, sont particulièrement adaptés pour cela. Cependant, l'instruction **code** ne peut pas être utilisée immédiatement après avoir appuyé sur **Lbl** ou après avoir appuyé sur **GTO**, car elle pourrait être considérée comme un code d'étiquette valide. Il faut d'abord utiliser un tag de remplacement (par exemple A), revenir en arrière avec le **BST**, puis saisir le code héxa avec le bouton **code**.

Pour plus d'informations sur l'adressage, voir le chapitre Adressage indirect.

## 77 Test si supérieur ou égal, x>=t

Libellé : **x>=t**

Séquence de touches : **2nd** **4**

L'instruction **x>=t** permet de comparer le registre X (contenu affiché) avec le registre auxiliaire T (renseigné par le bouton **x<>t**). Si le registre X est supérieur ou égal au registre T, un saut est effectué vers l'adresse qui est spécifiée en paramètre de l'instruction. L'adresse peut être une adresse absolue ou une étiquette.

L'instruction **x>=t** n'a pas de code spécial pour l'adressage indirect. Néanmoins, un adressage indirect est possible de telle sorte que le code **Ind** soit stocké dans le programme après le code **x>=t**. Si la condition est remplie, l'adresse absolue ou le code d'étiquette est lu à partir du registre spécifié, comme décrit plus en détail dans le chapitre Adressage indirect.

Si le préfixe **INV** est spécifié avant le code **x>=t**, la fonction inverse est exécutée - un saut vers l'adresse spécifiée est effectué si le registre X est

plus petit que le registre T.

Pour plus d'informations, voir l'instruction 67 Test d'égalité registres x et t  $x=t$ . La recommandation pour le test d'égalité s'applique également à l'instruction `x>=t`.

## 78 Saisie données statistiques, Stat

Libellé : `Stat`

Séquence de touches : `2nd 5`

L'instruction `Stat` est utilisée pour saisir des données lors de la réalisation de calculs statistiques (moyenne, corrélation, variation) et lors du calcul de régression linéaire (ajustement avec une droite d'approximation). L'instruction utilise les registres de données R01 à R06 pour stocker les calculs intermédiaires. Avant utilisation, les registres doivent d'abord être réinitialisés avec la commande `INV Cms`, qui réinitialise R01 à R06, ainsi que X et T. Une autre alternative est le sous-programme CLR du programme de bibliothèque ML-01 (appelé avec `Pgm 01 SBR CLR`).

Lors de la saisie de données statistiques par paires (x, y), la valeur 'x' est écrite en premier et en appuyant sur `x<>t` elle est transférée au registre T. Ensuite, la valeur 'y' est écrite et en appuyant sur `Stat` à la fois x et y les valeurs sont enregistrées. L'écran (dans le registre X) indiquera le nombre d'éléments 'n' insérés jusqu'à présent. Le contenu du registre T (valeur x) est augmenté de 1 avec l'instruction `Stat` car si les valeurs x doivent différer de 1, il n'est pas nécessaire de les insérer, il suffit d'insérer le x initial. valeur dans le registre T, puis écrivez uniquement les valeurs y. S'il n'est pas nécessaire d'évaluer des couples de valeurs (x, y), il suffit de saisir uniquement la valeur y.

Si le préfixe `INV` est donné avant l'instruction `Stat`, la valeur saisie est à la place soustraite. De cette manière, une valeur saisie par erreur peut être corrigée : la valeur x des données erronées est saisie, `x<>t` est enfoncé, la valeur y des données erronées est saisie et les données erronées sont soustraites en appuyant sur `INV Stat`. Vous pouvez alors continuer avec les nouvelles données correctes. S'il n'est pas nécessaire d'évaluer des couples de valeurs (x, y), il suffit de saisir uniquement la valeur y des

données erronées.

Registres utilisés :

<b>R01</b>	somme des y
<b>R02</b>	somme des y <sup>2</sup>
<b>R03</b>	nombre d'éléments n

<b>R04</b>	somme des x
<b>R05</b>	somme des x <sup>2</sup>
<b>R06</b>	somme des x*y

## 79 Moyenne statistique, Mean

Libellé : **Mean**

Séquence de touches : **2nd** **6**

L'instruction **Mean** calcule la moyenne des valeurs « x » et « y » saisies à l'aide de la fonction statistique **Stat** (voir 78 Saisie données statistiques Stat pour plus de détails). L'écran affiche la moyenne des valeurs « y ». Après avoir appuyé sur **x<>t**, la moyenne des valeurs « x » est affichée à partir du registre T.

Préfixer **INV** avant la fonction **Mean** calcule l'écart type. L'écran affiche l'écart type des valeurs 'y' **devy = sqrt((sum(y<sup>2</sup>) - sum(y)<sup>2</sup>/N)/(N-1))**, et stocke dans le registre T l'écart type des valeurs 'x' **devx = sqrt((sum(x<sup>2</sup>) - sum(x)<sup>2</sup>/N)/(N-1))**.

Exemple:

**INV** **CMS** [0] ... réinitialiser les registres

**9** **6** **Stat** [1] ... 1. saisie valeur y 96

**8** **1** **Stat** [2] ... 2. saisie valeur y 81

**9** **7** **Stat** [3] ... 3. saisie valeur y erronée

**9** **7** **INV** **Stat** [2] ... 3. annulation saisie valeur y

**8** **7** **Stat** [3] ... 3. saisie valeur y 87

**7** **0** **Stat** [4] ... 4. saisie valeur y 70

**9 3 Stat** [5] ... 5. saisie valeur y 93

**7 7 Stat** [6] ... 6. saisie valeur y 77

**Mean** [84] ... moyenne des valeurs saisies = 84

**INV Mean** [9.87927...] ... écart type = 9.87927...

**Op 11** [81.333...] ... variance = 81.333...

**RCL 01** [504] ... somme de toutes les valeurs = 504

## 7A Saut conditionnel, IF

Libellé : **IF**

Séquence de touches : **2nd 2nd SBR**

L'instruction **IF** est un saut conditionné à la comparaison des registres. Contrairement aux instructions **x=t** et **x>=t**, le nombre affiché n'est pas utilisé ici, mais les registres sont comparés entre eux. Il est possible d'utiliser des registres de données ou des registres HIR comme opérands, en adressage direct et indirect.

Le premier paramètre, après le code **IF**, est un nombre HEX à deux chiffres. Le premier chiffre (le plus élevé) représente le code de l'opération en cours. Le deuxième chiffre (inférieur) représente l'index du premier opérande. Le premier opérande peut être le registre de données R00 à R15, le registre HIR H0 à H15, soit sous forme de registres directs, soit sous forme de registres indirects. S'il s'agit d'un adressage indirect avec le registre HIR, l'index du registre est lu à partir du registre HIR, mais le registre de données (et non le registre HIR) est utilisé comme registre indexé. Le registre HIR est utilisé comme pointeur vers la mémoire utilisateur principale des registres de données.

Le deuxième paramètre est un nombre décimal représentant un numéro de registre ou une constante décimale. Le registre peut être le registre de données R00 à R99 ou le registre HIR H0 à H15. S'il s'agit d'un adressage indirect, le registre de données est toujours adressé, indépendamment du fait que l'index puisse être lu à partir du registre HIR.

Le troisième paramètre de l'instruction est l'adresse cible à laquelle le programme saute lorsque la condition est remplie. L'adresse peut être une

adresse absolue, d'étiquette ou indirecte. L'adressage sera indirect si le code **Ind** suivi du numéro de registre est donné comme troisième paramètre. Une adresse indirecte peut être utilisée indépendamment du fait qu'une instruction **IF** directe ou une instruction **IF Ind** indirecte soit utilisée.

Si la touche de préfixe **INV** est enfoncée avant l'instruction **IF**, l'instruction **IF** est remplacée par une instruction **IF Ind** avec adressage indirect (voir 6D Condition indirecte IF Ind pour plus de détails).

Codes opération **IF** (premier paramètre) :

Le code d'opération se compose de 2 caractères hexadécimaux. Le premier caractère indique en même temps le type de registre (registres de données pour les codes 0 à 7, registres HIR pour les codes 8 à F) et le type d'opération : inférieur ou égal, inférieur, égal, supérieur...

Le type de registre s'applique aux deux registres de comparaison. Il n'est pas possible de comparer le registre de données (en adressage direct) avec le registre HIR.

Premier paramètre (cible de l'opération)		
Reg. Rxx	Reg. Hxx	Opération
0n	8n	<= (*)
1n	9n	<
2n	An	=
3n	Bn	<=
4n	Cn	>
5n	Dn	<>
6n	En	>=
7n	Fn	> (*)

$n$  = numéro de registre exprimé en hexa :

00 à 15 soit 0 à F

(\*) constante en deuxième paramètre

Attention, l'instruction **IF** permet de saisir des paramètres en code HEX et ainsi d'insérer un octet de valeur 0FF dans le code. Si possible, évitez une telle valeur d'octet, car elle serait interprétée comme un espace vide et serait corrompue lors d'un changement de programme en mémoire avec **Ins** ou **Del**.

Exemple - recherche d'une valeur >8 dans les registres de données

**CMs** **9** **STO** **13** ... remise à zéro des registres, stocke le chiffre 9 dans le registre R13

**3** **0** **HIR** **02** ... (30 STO H2) stockage du nombre de registres (30) dans H2

**RST** **LRN** ... active le mode de programmation

**CLR** **HIR** **01** ... (0 STO H1) préparation de l'index initial 0 dans H1

**Lbl** **Inx** ... étiquette boucle de recherche

**IF** **Ind** **F1** **08** **=** ... si contenu indirect H1 > 8, aller à l'étiquette =

... Note: **IF** **Ind** s'écrit en séquence **INV** **IF**

**HIR** **71** ... (Inc H1) incrémente l'index de 1 dans H1

**IF** **91** **02** **Inx** ... si H1 < H2 aller à l'étiquette Inx

**CLR** **1/x** **R/S** ... erreur et indication d'arrêt (non trouvé)

**Lbl** **=** ... étiquette traitement si trouvé

**HIR** **11** **R/S** ... affichage de l'index du registre trouvé et arrêt

**LRN** ... revient au mode exécution

**RST** **R/S** [13] ... TEST 1 : remise à zéro pointeur programme puis exécution, retourne la valeur 13 (numéro du registre trouvé).

**8** **STO** **13** ... stocke la valeur 8 dans R13

**RST** **R/S** [9.999+9999] ... TEST 2 : non trouvé, erreur clignotante

## 7E OU exclusif entre deux opérandes binaires, XOR

Libellé : **XOR**

Séquence de touches : **2nd** **2nd** **|**

L'instruction **XOR** effectue une opération XOR (somme exclusive) au niveau du bit entre le premier et le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur **|**.

La somme exclusive au niveau du bit signifie que le résultat de l'opération est un bit «1» uniquement si les deux bits d'entrée sont différents.

Exemple:

**BIN** ... passage à l'affichage en mode binaire

**1** **0** **1** **1** **0** **0** **1** **1** ... saisie du premier opérande (10110011 = 179 décimal)

**XOR** ... la somme exclusive au niveau du bit est effectué

**0** **0** **1** **0** **0** **1** **1** **0** ... saisie du deuxième opérande (100110 = 38 décimal)

**|** [10010101] ... calcul du résultat (10010101 = 149 décimal)

$$\begin{array}{r} 10110011 \\ \text{XOR } 00100110 \\ = 10010101 \end{array}$$

## 80 Grades, Grad

Libellé : **Grad**

Séquence de touches : **2nd** **+**

Le bouton **Grad** bascule les calculs des fonctions trigonométriques en grades (un angle total est 400).

Si les calculs doivent être effectués indépendamment de la mesure

angulaire sélectionnée, les fonctions **Op 72** et **Op 73** peuvent être utilisées pour les conversions.

## 81 Réinitialisation pointeur de programme, RST

Libellé : **RST**

Séquence de touches : **RST**

Le bouton **RST** est utilisé pour réinitialiser le pointeur de programme, c'est-à-dire en plaçant le pointeur sur le pas 0. Lorsqu'un programme d'un module de bibliothèque est sélectionné, il est désactivé et revient au programme principal de l'utilisateur. Si le programme s'exécute à partir d'un module de bibliothèque, le programme s'arrête. Si le programme principal est en cours d'exécution, il continue à partir de l'adresse 0.

L'instruction **RST** réinitialise les états des interrupteurs, sauf l'interrupteur 15 qui indique une erreur E.

## 82 Gestion registres internes, HIR

Libellé : **HIR**

Séquence de touches : **2nd INV**

Dans la calculatrice TI-58/59 d'origine, l'instruction **HIR** était une instruction cachée, non documentée dans les manuels. Elle utilisait des registres de pile d'opérations arithmétiques pour son fonctionnement.

Pour la calculatrice ET-58, l'instruction **HIR** est devenue l'une des instructions principales des programmes de bibliothèque. Il utilise un ensemble distinct de 16 registres de contrôle H0 à H15 qui (contrairement à la TI-58/59 d'origine) ne sont pas partagés avec la pile d'opérations arithmétiques, ce sont des registres complètement séparés et indépendants. Ils sont principalement destinés à servir de registres de contrôle fonctionnels, tandis que les registres de données R00 à R99 restent entièrement disponibles pour les



données utilisateur. L'ensemble des fonctions de l'instruction **HIR** a été considérablement élargi et comprend des options similaires aux instructions qui sont utilisées avec les registres de données.

Le code **HIR** de l'instruction est suivi d'un octet codé HEX en paramètre. Le premier chiffre HEX (supérieur) représente le code d'instruction, le deuxième chiffre HEX (inférieur) est le numéro de registre HIR H0 à H15 sur lequel opérer.

Placer le code **Ind** après l'instruction **HIR**, mais avant de spécifier le paramètre, transforme l'instruction **HIR** en une instruction **HIR Ind** indirecte (voir 27 Instruction interne indirecte HIR Ind). L'instruction indirecte fonctionne avec le registre de données R00 à R99 dont l'index est contenu dans le registre HIR H0 à H15 donné par le deuxième chiffre du paramètre de l'instruction **HIR**. (voir chapitre Adressage indirect).

Codes opération des instructions HIR (1er chiffre du paramètre) :

0 ... **STO**, stocke le contenu de l'affichage dans le registre HIR

1 ... **RCL**, récupère le contenu du registre HIR

2 ... **round**, arrondit le registre HIR à l'entier le plus proche

3 ... **SUM**, ajoute l'affichage au registre HIR

4 ... **Prd**, multiplie le registre HIR par l'affichage

5 ... **INV SUM**, soustrait l'affichage du registre HIR

6 ... **INV Prd**, divise le registre HIR par l'affichage

7 ... **Inc**, incrémente (augmente de 1) le registre HIR

8 ... **Dec**, décrémente (diminuer de 1) le registre HIR

9 ... **Exc**, échange l'affichage avec le registre HIR

A ... **GTO**, saut à l'adresse contenue dans le registre HIR (absolue

ou étiquette)

B ... **SBR**, appel du sous-programme selon le registre HIR

C *adr* ... **x<=0**, saut à l'adresse *adr* si le registre HIR  $\leq 0$

D *adr* ... **x>0**, saut à l'adresse *adr* si le registre HIR  $> 0$

E *adr* ... **DJNZ** (**D**ecrement and **J**ump if **N**ot **Z**ero, équivalent à **DSZ**), décrémente/incrémente le registre HIR et saute à l'adresse *adr* si le registre HIR n'est pas 0.

F *adr* ... **DJZ** (**D**ecrement and **J**ump if **Z**ero, équivalent à **INV DSZ**), décrémente/incrémente le registre HIR et saute à l'adresse *adr* si le registre HIR est 0.

L'instruction **HIR 20** a été utilisée sur la TI-58/59 d'origine pour brancher le programme de microcode interne. Il permettait, selon un registre interne prédéfini, d'effectuer soit un saut relatif, soit la terminaison de la fonction. Avec ET-58, il perd son sens et est donc utilisé à d'autres fins - arrondi du registre HIR. L'arrondi est utile dans le cas de nombreux calculs répétés, où les erreurs d'arrondi peuvent s'accumuler et la comparaison du résultat de l'opération de mise en correspondance peut échouer. Un exemple typique sont les cycles dans lesquels une constante est ajoutée ou soustraite à plusieurs reprises à un registre. En accumulant l'erreur d'inexactitude, la valeur résultante peut s'écarter du nombre de cycles testé. L'arrondi continu du résultat en nombres entiers corrigera l'erreur.

Les fonctions A (GTO) et B (SBR) utilisent le contenu du registre HIR comme adresse cible. L'adresse peut être absolue, c'est-à-dire un nombre décimal compris entre 0 et 999 ou un code d'étiquette. S'il s'agit d'un code tag, la valeur décimale du code du bouton tag est utilisée, multipliée par le nombre 256. Pour une description plus détaillée, voir le chapitre Adressage indirect.

Les fonctions C ( $x \leq 0$ ) à F (DJZ) ont également un deuxième paramètre, l'adresse de saut. L'adresse peut être absolue ou une étiquette. En ajoutant le code **Ind**, l'adresse peut être indirecte, contenue dans le

registre de données indiqué.

Les fonctions E (DJNZ) et F (DJZ) ont une fonction similaire aux instructions **DSZ** et **INV DSZ** (voir 97 Boucle de programme Dsz), mais contrairement à eux, ils utilisent un registre HIR. Si le contenu du registre avant l'opération est supérieur à 0, la valeur du registre est diminuée de 1. S'il est inférieur à 0, la valeur est augmentée de 1. Si le contenu du registre était 0, la valeur est pas changé. Si le résultat de l'opération est zéro ou si l'opération a franchi la limite zéro, l'opération pour zéro est effectuée selon la fonction sélectionnée. Enfin, la valeur du registre est arrondie à un nombre entier, ce qui évite l'accumulation d'erreurs lors de la répétition des opérations.

**Attention**, l'instruction **HIR** permet de saisir un paramètre en code HEX et ainsi d'insérer un octet de valeur OFF dans le code (opération DJZ H15). Si possible, évitez une telle valeur d'octet, car elle serait interprétée comme un espace vide et serait corrompue lors d'un déplacement de programme en mémoire avec **Ins** ou **Del**.

Exemple - remplissage des registres R00 à R99 avec les valeurs 100 à 199

**RST** **LRN** ... activation du mode programmation

**Lbl** **A** ... étiquette du sous-programme

**0** **HIR** **01** ... (≈STO H1) index initial des registres = 0

**1** **0** **0** ... valeur stockée (et aussi nombre de registres)

**HIR** **02** ... (≈STO H2) compteur de boucle = 100

**Lbl** **=** ... étiquette de début de boucle

**(** **HIR** **Ind** **01** ... (≈STO Ind H1) stockage dans le registre adressée par H1

**+** **1** **)** ... incrémenter le numéro sur l'écran

**HIR** **71** ... (≈Inc H1) incrémenter l'indice H1

**HIR** **E2** **=** ... (≈DJNZ H2 =) décrémenter H2 et passer à = si différent de 0

**RTN** ... fin du sous-programme (**RTN** contraction de **INV** **SBR**)

**LRN** ... sortir du mode programmation

**A** [200] ... exécution du programme

**RCL** **99** [199] ... vérifier le registre R99 pour voir s'il contient 199

**RCL** **13** [113] ... vérifier le registre R13 pour voir s'il contient 113

## 83 Saut indirect, GTO Ind

Libellé : **GTO** **Ind**

Séquence de touches : **GTO** **2nd** **y^x**

L'instruction **GTO** **Ind** effectue un saut dans le programme de la même manière que l'instruction **GTO** (voir 61 Saut GTO), seulement au lieu du paramètre d'instruction, l'adresse de saut est extraite du registre de données. Le registre peut contenir à la fois une adresse absolue (000 à 999) et une étiquette (code d'étiquette décimal \* 256).

L'instruction est créée en appuyant sur la touche **Ind** après la touche **GTO**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

### Exemple

**RST** **LRN** ... active le mode de programmation

**GTO** **Ind** **01** ... saut indirect à l'adresse du registre R01

**1** **R/S** ... adresse 2, affichera le numéro 1

**2** **R/S** ... adresse 4, affichera le numéro 2

**3** **R/S** ... adresse 6, affichera le numéro 3

**LRN** .. revient au mode exécution

**2** **STO** **01** **RST** **R/S** [1] ... test 1 : passe à l'adresse 2 et affiche 1

**CLR** **4** **STO** **01** **RST** **R/S** [2] ... test 2 : passe à l'adresse 4 et affiche 2

**CLR** **6** **STO** **01** **RST** **R/S** [3] ... test 3 : passe à l'adresse 6 et affiche 3

## 84 Opération spéciale indirecte, Op Ind

Libellé : **Op Ind**

Séquence de touches : **2nd Op 2nd y^x**

L'instruction **Op Ind** effectue une opération spéciale tout comme l'instruction **Op** (voir chapitre 69 Opérations spéciales OP), seulement au lieu du paramètre d'instruction, le code d'opération est extrait du registre de données.

L'instruction est créée en appuyant sur la touche **Ind** après la touche **Op**, mais avant de saisir le paramètre, qui sera un numéro de registre à 2 chiffres. Pour plus d'informations sur l'adressage indirect, voir le chapitre Adressage indirect.

## 85 Addition, +

Libellé : **+**

Séquence de touches : **+**

Le bouton **+** ajoute le deuxième opérande au premier opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur **=**.

## 86 Positionnement drapeau, StFlg

Libellé : **StFlg**

Séquence de touches : **2nd RST**

L'instruction **StFlg** (Set Flag) lève le drapeau 0 à 15, donné en paramètre de l'instruction. Le paramètre est un nombre de 00 à 0F. L'utilisation du préfixe **INV** avant que l'instruction **StFlg** effectue l'opération inverse : baisser le drapeau.

L'instruction **StFlg** n'a pas de code pour l'adressage indirect. L'adressage

indirect peut être obtenu en remplaçant le numéro de registre 00..0F par le code **Ind** suivi du numéro de registre contenant le numéro de drapeau.

L'instruction **RST** réinitialise les états des drapeaux, sauf le drapeau 15 qui indique une erreur E.

Certains drapeaux ont une signification particulière :

**7** ... le drapeau peut être réglé par les opérations **Op 18** et **Op 19** en fonction de la condition d'erreur

**8** ... si le drapeau 8 est activé, le programme s'arrête lorsqu'une erreur logicielle E se produit. S'il n'est pas activé, la calculatrice indique une erreur, mais continue le calcul. Lorsqu'une erreur matérielle F se produit, le programme s'arrête toujours.

**15** ... le drapeau 15 est directement connecté à l'indication d'erreur E. L'état d'erreur peut être testé, activé et désactivé avec le commutateur 15.

*Remarque : Après avoir réinitialisé l'indication d'erreur par la mise à zéro du drapeau 15, le caractère E peut rester allumé sur l'afficheur. Ce n'est pas un défaut, le caractère disparaît après le premier changement du contenu de l'affichage.*

## 87 Test drapeau, IfFlg

Libellé : **IfFlg**

Séquence de touches : **2nd 1**

L'instruction **IfFlg** (If Flag) saute à l'adresse donnée par le troisième paramètre si le drapeau 0 à 15 spécifié par le deuxième paramètre est levé. Placer le préfixe **INV** avant l'instruction **IfFlg** exécute la fonction opposée : le saut est effectué au cas où le drapeau serait désactivé.

L'instruction **IfFlg** n'a pas de code pour l'adressage indirect. L'adressage indirect du numéro de drapeau peut être réalisé en remplaçant le deuxième paramètre par le code **Ind** suivi du numéro de registre qui contiendra le numéro de drapeau. De même, un adressage par saut indirect peut être réalisé - au lieu du troisième paramètre, le code **Ind** suivi du numéro du

registre contenant l'adresse ou l'étiquette de destination est donné. Pour plus d'informations sur l'adressage indirect, consultez le chapitre Adressage indirect.

L'instruction **RST** réinitialise les états des drapeaux, sauf le drapeau 15 qui indique une erreur E.

Exemple - affichage indirect de l'état du drapeau:

**RST** **LRN** ... activation du mode programmation

**Lbl** **IfFlg** ... étiquette de début de sous-programme

**INV** **IfFlg** **Ind** **01** **=** ... si le drapeau indiqué par le registre R01 n'est pas levé aller à **=**

**1** ... le résultat sera 1 (levé)

**RTN** ... fin du sous-programme (**INV** **SBR**)

**Lbl** **=** ... saute ici si l'interrupteur est éteint

**CLR** ... le résultat sera 0 (baissé)

**RTN** ... fin du sous-programme (**INV** **SBR**)

**Lbl** **A** ... étiquette de test de drapeau 1

**1** **STO** **01** ... stocke le numéro de drapeau 1 dans R01

**SBR** **IfFlg** ... appel du test d'état du drapeau 1

**RTN** ... fin du sous-programme (**INV** **SBR**)

**Lbl** **B** ... étiquette de test de drapeau 2

**2** **STO** **01** ... stocke le numéro de drapeau 2 dans R01

**SBR** **IfFlg** ... appel du test d'état du drapeau 2

**RTN** ... fin du sous-programme (**INV** **SBR**)

**LRN** ... sortie du mode programmation

**StFlg** **2** ... lève le drapeau 2

**A** [0] ... test 1 : le drapeau 1 est baissé

**B** [1] ... test 2 : le drapeau 2 est levé

## 88 Conversions minutes et secondes / décimal, DMS

Libellé : **DMS**

Séquence de touches : **2nd** **2**

L'instruction **DMS** peut être utilisée pour convertir un temps ou un angle exprimé en minutes et secondes (secondes) en un nombre décimal. L'entrée de la fonction est un nombre exprimé sous la forme DD.MMSS, ayant le nombre heures ou de degrés dans la partie entière, suivi du nombre de minutes aux deux premières décimales et du nombre de secondes (secondes) aux deux décimales suivantes. Les secondes décimales peuvent être renseignées sous forme de chiffres décimaux supplémentaires. Le résultat de la fonction est un nombre décimal représentant le nombre d'heures ou de degrés DD.DDDD, exprimé sous forme de nombre décimal.

Mettre le préfixe **INV** avant l'instruction **DMS** effectue l'opération inverse : le temps ou l'angle exprimé à l'aide d'un nombre décimal est converti en minutes et secondes (secondes). L'entrée de la fonction est un nombre décimal représentant le nombre d'heures ou de degrés DD.DDDD. Le résultat est le nombre DD.MMSS, avec le nombre d'heures ou de degrés dans la partie entière, suivi du nombre de minutes aux deux premières décimales et du nombre de secondes (secondes) aux deux décimales suivantes. Si le résultat n'est pas un nombre entier de secondes, les décimales des secondes sont arrondies sous forme de chiffres décimaux supplémentaires.

Exemple, somme de temps:

**1** **2** **.** **3** **0** **.** **2** **3** ... heure à 12:30:23 (12 heures, 30 minutes et 23 secondes)

**DMS** [12.50638...] ... conversion en heures exprimées en décimales

+ **3** **.** **4** **5** **.** **1** **2** **DMS** [3.7533...] ... plus 3 h 45 mn et 12 secondes

**=** **INV** **DMS** [16.1535] ... temps résultant 16 h 15 mn et 35 secondes



## 89 Nombre de Ludolf [constante pi], pi

Libellé : **pi**

Séquence de touches : **2nd** **3**

Le bouton **pi** permet de saisir la constante PI «nombre de Ludolf», qui a la valeur 3.141592653589793238..

## 8A Opérations de registre à registre, Reg

Libellé : **Reg**

Séquence de touches : **2nd** **2nd** **RST**

L'instruction **Reg** permet une manipulation directe des registres, sans avoir besoin d'utiliser le contenu de l'affichage (registre X).

L'instruction **Reg** est suivie de deux paramètres. Le premier paramètre est au format HEX. Son premier chiffre (le plus élevé) représente le code d'opération 0 à 0F. Le deuxième chiffre (inférieur) indique l'opérande cible de l'opération. L'opérande cible est le registre de données R00 à R15 ou le registre HIR H0 à H15. L'opérande cible peut également être indirect. Le deuxième paramètre spécifie l'opérande source. Il peut s'agir du registre de données R00 à R99 ou du registre HIR H0 à H15.

Préfixer **INV** avant l'instruction **Reg** effectue une opération inverse ou alternative.


En précisant le code **Ind** après le premier paramètre de l'instruction **Reg**, mais avant de saisir le deuxième paramètre, l'instruction se transforme en instruction indirecte **Reg Ind** (code 6C, voir chapitre 6C Fonctionnement indirect avec les registres Reg Ind). L'instruction indirecte **Reg Ind** permet l'adressage indirect du deuxième paramètre de l'opération (le registre source). L'adressage indirect n'affectera pas la manière dont le premier opérande (cible) est adressé, il est toujours adressé par le type d'instruction. Et cela n'affecte pas non plus la constante, elle est toujours chargée à partir du paramètre d'instruction. De plus amples informations sur l'adressage indirect sont disponibles dans le chapitre Adressage

indirect.


### Codes d'opération:

Le code d'opération est déterminé par le premier chiffre (supérieur) du premier paramètre. Les bits 0 et 1 déterminent le numéro d'opération. Le bit 2 indique l'adressage indirect du premier opérande (cible). Le bit 3 indique les registres HIR. Dans le cas des registres HIR, les registres HIR seront utilisés comme deux opérandes, une opération directe entre les données et les registres HIR ne peut pas être effectuée. Si l'un des opérandes est adressé indirectement, l'index peut être stocké soit dans le registre de données, soit dans le registre HIR, mais le registre adressé est toujours le registre de données. Le registre HIR est utilisé comme pointeur vers les registres de données.

Dans le tableau suivant, les codes d'opération **0** à **3** représentent un registre de données directes (colonne 1), les codes **4** à **7** représentent un registre de données indirectes (colonne 2), les codes **8** à **B** représentent un registre HIR direct (colonne 3), les codes **C** à **F** représentent un registre HIR indirect, c'est-à-dire que le registre HIR est un pointeur vers un registre (colonne4).

Premier paramètre (cible de l'opération)					
Reg. Rxx	Reg. Rxx Ind	Reg. Hxx	Reg. Hxx Ind	Opération	
<b>0</b> <i>n</i>	<b>4</b> <i>n</i>	<b>8</b> <i>n</i>	<b>C</b> <i>n</i>	Copie	Echange
<b>1</b> <i>n</i>	<b>5</b> <i>n</i>	<b>9</b> <i>n</i>	<b>D</b> <i>n</i>	Addition	Soustraction
<b>2</b> <i>n</i>	<b>6</b> <i>n</i>	<b>A</b> <i>n</i>	<b>E</b> <i>n</i>	Multiplication	Division
<b>3</b> <i>n</i>	<b>7</b> <i>n</i>	<b>B</b> <i>n</i>	<b>F</b> <i>n</i>	Constante	- Constante

*n* = numéro de registre exprimé en hexa : 00 à 15 soit 0 à F

Dans le cas de la définition d'une constante, la constante est le 2ème paramètre de l'instruction, sous forme de nombre décimal BCD (par exemple, l'octet 99 signifie la valeur 99). La signification n'est pas affectée par l'adressage indirect, même dans ce cas, la constante utilisée est directement le 2ème paramètre. Lorsque le préfixe  est utilisé, la constante est stockée sous forme de nombre négatif.

Remarque : L'instruction **Reg** permet de saisir un paramètre avec une valeur de OFF (opération OFF = chargement d'une constante dans le registre HIR H15). Évitez d'utiliser l'octet OFF si possible. Lors d'un déplacement de mémoire avec **Ins** ou **Del**, le paramètre OFF serait traité comme un espace vide et le code serait corrompu.

Exemple:

**10** **HIR** **01** ... (STO H1) stocke le numéro 10 dans le registre HIR H1

**Reg** **B2** **13** ... stocke la constante 13 dans le registre HIR H2

**Reg** **A1** **02** ... (Prd H1 H2) Le registre HIR H1 multiplie le registre HIR H2

**HIR** **11** [130] ... (RCL H1), le contenu du registre HIR H1 est de 130

## 8B Mode hexadécimal, HEX

Libellé : **HEX**

Séquence de touches : **2nd** **2nd** **1**

L'instruction **HEX** fait passer le mode d'affichage standard (décimal) en affichage hexadécimal. La mantisse du nombre est affichée et saisie en mode hexadécimal, chiffres 0 à F, y compris la partie fractionnaire de la mantisse. L'exposant est toujours un nombre décimal.

La saisie du préfixe **INV** avant l'instruction **HEX** active le mode de débogage auxiliaire, où la première ligne de l'écran affiche la mantisse du nombre affichée au format interne, sous forme de 16 chiffres hexadécimaux. De cette façon, même les décimales cachées de la mantisse peuvent être affichées. Le premier chiffre en partant de la gauche est le chiffre le plus élevé. Le bit de poids fort de la mantisse ayant une valeur de «1» est masqué et remplacé par le bit de signe. Le mode débogage est désactivé en sélectionnant le mode d'affichage sans le préfixe **INV**.

En mode exposant Eng (ingénieur), l'exposant est un multiple de 4.

## 8C Mode binaire, BIN

Libellé : **BIN**

Séquence de touches : **2nd** **2nd** **2**

L'instruction **BIN** fait passer le mode d'affichage standard (décimal) en affichage binaire. La mantisse du nombre est affichée et saisie en mode BIN, chiffres 0 à 1, y compris la partie fractionnaire de la mantisse. L'exposant est toujours un nombre décimal.

Le préfixe **INV** avant l'instruction **BIN** active le mode de débogage auxiliaire, où la première ligne de l'écran affiche la mantisse du nombre affiché au format interne, sous forme de 16 chiffres hexadécimaux. De cette façon, même les décimales cachées de la mantisse peuvent être affichées. Le premier chiffre en partant de la gauche est le chiffre le plus élevé. Le bit de poids fort de la mantisse ayant une valeur de «1» est masqué et remplacé par le bit de signe. Le mode débogage est désactivé en sélectionnant le mode d'affichage sans le préfixe **INV**.

En mode exposant Eng (ingénieur), l'exposant est un multiple de 4.

## 8D Mode octal, OCT

Libellé : **OCT**

Séquence de touches : **2nd** **2nd** **3**

L'instruction **OCT** fait passer le mode d'affichage standard (décimal) en affichage octal. La mantisse du nombre est affichée et saisie en mode octal, chiffres 0 à 7, y compris la partie fractionnaire de la mantisse. L'exposant est toujours un nombre décimal.

La spécification du préfixe **INV** avant l'instruction **OCT** active le mode de débogage auxiliaire, où la première ligne de l'écran affiche la mantisse du nombre affiché au format interne, sous forme de 16 chiffres hexadécimaux. De cette façon, même les décimales cachées de la mantisse peuvent être affichées. Le premier chiffre en partant de la gauche est le chiffre le plus élevé. Le bit de poids fort de la mantisse ayant une valeur de «1» est masqué et remplacé par le bit de signe. Le mode débogage est désactivé

en sélectionnant le mode d'affichage sans le préfixe **INV**.

En mode exposant Eng (ingénieur), l'exposant est un multiple de 3.

## 8E OU entre deux opérands binaires, OR

Libellé : **OR**

Séquence de touches : **2nd** **2nd** **+**

L'instruction **OR** effectue une opération OU (somme de bits) au niveau du bit entre le premier et le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant plusieurs fois sur **=**.

Une somme au niveau du bit signifie que le résultat de l'opération est un bit «1» si au moins un bit d'entrée a la valeur «1».

Exemple:

**BIN** ... passage à l'affichage en mode binaire

**1** **0** **1** **1** **0** **0** **1** **1** ... saisie du premier opérande (10110011 = 179 décimal)

**OR** ... l'opération OU binaire est effectuée

**0** **0** **1** **0** **0** **1** **1** **0** ... saisie du deuxième opérande (00100110 = 38 décimal)

**=** [10110111] ... calcul du résultat (10110111 = 183 décimal)

$$\begin{array}{r} 10110011 \\ \text{OR } 00100110 \\ = 10110111 \end{array}$$

## 91 Démarrer et arrêter le programme, R/S

Libellé : **R/S**

Séquence de touches : **R/S**

Le bouton **R/S** peut être utilisé pour démarrer ou arrêter un programme en cours. Au démarrage, le programme commence à s'exécuter à partir du pointeur de programme courant (l'adresse courante peut être trouvée en passant au mode de programmation **LRN**).

Un programme peut ne pas toujours pouvoir continuer à s'exécuter après son arrêt : une adresse de retour d'un sous-programme peut être perdue ou un autre programme de bibliothèque peut rester commuté.

## 92 Retour de sous-programme, RTN

Libellé : **RTN**

Séquence de touches : **INV SBR**

L'instruction **RTN** est utilisée pour retourner au programme depuis un sous-programme. L'adresse d'origine après l'instruction qui a appelé le sous-programme (l'instruction **SBR** ou **A** à **F**) est extraite de la pile d'adresses et devient la prochaine de pas à exécuter. Si le sous-programme a été lancé à partir du clavier, la calculatrice s'arrête.

L'instruction RTN est saisie depuis le clavier en appuyant sur **INV SBR**.

## 93 Séparateur décimal, .

Libellé : **.**

Séquence de touches : **.**

Le bouton **.** sert à séparer les chiffres de la partie entière et les chiffres de la partie décimale d'un nombre. Si le bouton est **.** enfoncé lors de l'édition

de l'exposant d'un nombre, l'édition reviendra à la saisie de la mantisse du nombre.

S'il est précédé du préfixe **INV** le point **.** tronque le nombre de façon identique à la séquence **EE INV EE**, mais l'avantage est que le mode exposant n'est pas modifié. Cette procédure facilite la suppression de la partie décimale cachée en arrondissant le nombre. Une autre alternative est l'opération **Op 82**.

## 94 Changement de signe, +/-

Libellé : **+/-**

Séquence de touches : **+/-**

Le bouton **+/-** change le signe du nombre sur l'écran. Si vous appuyez dessus lors de la saisie de l'exposant d'un nombre, **+/-** change le signe de l'exposant.

Le fonctionnement Op 10 peut également être utilisé à la place de INV +/-.

## 95 Réalisation du calcul, =

Libellé : **=**

Séquence de touches : **=**

Le bouton **=** permet de terminer les opérations arithmétiques en cours et d'effectuer un calcul.

En appuyant plusieurs fois sur la touche **=**, la dernière opération effectuée au niveau le plus bas peut être répétée. Le premier opérande est le nombre affiché, le deuxième opérande est le nombre saisi lors de l'opération comme deuxième opérande (ou le deuxième résultat du calcul intermédiaire). Les opérandes peuvent être échangés avec le bouton **x<>y**.

Attention : Certains programmes de la TI-58/59 d'origine ne calculent pas

avec répétition des opérations, ils utilisent la touche  $\boxed{=}$  plus d'une fois et cela peut entraîner un calcul incorrect. Lors de l'importation de programmes, il peut être nécessaire de vérifier et de modifier cela.

Exemple:

$\boxed{5} \boxed{\times} \boxed{6} \boxed{=}$  [30] ...  $5 \times 6 = 30$

$\boxed{7} \boxed{=}$  [42] ...  $7 \times 6 = 42$

$\boxed{9} \boxed{:} \boxed{3} \boxed{=}$  [3] ...  $9 / 3 = 3$

$\boxed{1} \boxed{2} \boxed{=}$  [4] ...  $12 / 3 = 4$

$\boxed{2} \boxed{\lt\>}$  [3] ... échange des opérandes, le deuxième opérande sera désormais le nombre 2

$\boxed{1} \boxed{6} \boxed{=}$  [8] ...  $16 / 2 = 8$

## 97 Boucle de programme, Dsz

Libellé :  $\boxed{\text{Dsz}}$

Séquence de touches :  $\boxed{2\text{nd}} \boxed{0}$

L'instruction  $\boxed{\text{Dsz}}$  (Decrement and Skip if Zero) est utilisée pour exécuter de manière répétée un programme à l'aide d'un registre de comptage de boucles. L'instruction  $\boxed{\text{Dsz}}$  est suivie de 2 paramètres. Le premier paramètre est le numéro du registre de données 0 à F (correspond aux registres R00 à R15). Le deuxième paramètre est l'adresse de saut - soit sous forme d'adresse absolue, soit sous forme d'étiquette.

La fonction  $\boxed{\text{Dsz}}$  consiste donc à décrémenter (diminuer de 1) le registre spécifié et s'il a atteint zéro, à sauter à l'adresse donnée comme deuxième paramètre et poursuivre l'exécution. (c'est-à-dire répéter la boucle si le registre n'est pas 0).

Si le préfixe  $\boxed{\text{INV}}$  est placé avant l'instruction  $\boxed{\text{Dsz}}$ , le sens opposé de l'instruction est exécuté : l'adresse de saut est ignorée tant que le résultat de la décrémentatation est nul. (c'est-à-dire sautez vers l'autre adresse lorsque le registre est différent de 0).



L'instruction **Dsz** ne possède pas de code spécial pour l'adressage indirect. L'adressage indirect peut être réalisé en spécifiant le code **Ind** avant le premier paramètre. Dans ce cas, l'index du registre du compteur est extrait du registre spécifié comme paramètre de code **Ind**, et/ou en spécifiant le code **Ind** avant le deuxième paramètre, auquel cas l'adresse de saut est extraite du registre spécifié comme deuxième code **Ind** paramètre. Voir également Adressage indirect.

Une alternative à l'instruction **Dsz** est l'instruction **HIR** avec les codes E0 à EF (DJNZ) et avec les codes F0 à FF (DJZ), qui utilisent des registres HIR au lieu de registres de données. Voir 82 Instructions internes HIR.

L'instruction de boucle **Dsz** fonctionne plus précisément comme suit : si le contenu du registre avant l'opération est supérieur à 0, la valeur du registre est diminuée de 1. S'il est inférieur à 0, la valeur est augmentée de 1. Si le contenu du registre était 0, la valeur est pas changé. Si le résultat de l'opération est zéro ou si l'opération a franchi la limite zéro, l'opération pour zéro est effectuée selon la fonction sélectionnée

Enfin, la valeur du registre est arrondie à un nombre entier, ce qui évite l'accumulation d'erreurs lors de la répétition des opérations. C'est en quoi l'instruction **Dsz** diffère de la fonction de la calculatrice TI-58/59 d'origine. L'arrondi est nécessaire car la calculatrice ET-58 fonctionne avec des nombres binaires. La TI-58/59 originale fonctionne avec les nombres BCD, elle arrondit donc automatiquement aux nombres exprimés en chiffres décimaux. Bien que cet écart de fonctionnalité ne se manifeste généralement pas dans la pratique, il est possible que certains programmes s'attendent à ce que les nombres décimaux de la boucle ne soient pas arrondis.

Exemple - effacer tous les registres:

**RST** **LRN** ... activation du mode programmation

**9** **9** **STO** **00** ... index du dernier registre dans registre R00

**CLR** ... 0 à stocker dans les registres

**Lbl** **CLR** ... étiquette de début de boucle

**STO** **Ind** **00** ... 0 (affichage) est stocké dans le registre d'index R00

**DSZ** **0** **CLR** ... décrémenter R00 et s'il n'est pas nul, aller à CLR

**R/S** ... fin de programme

**LRN** ... sortie du mode programmation

**9 STO 99** ... stocker la valeur 9 pour test dans le registre R99

**5 STO 15** ... stocker la valeur 5 pour test dans le registre R 15

**RST R/S** ... lancer le programme

**RCL 99** [0] ... contrôle registre R99, contient 0

**RCL 15** [0] ... contrôle registre R15, contient 0

**RCL 00** [0] ... contrôle registre R00, contient 0

*Remarque : Bien que la boucle n'écrit pas dans le registre R00, elle contient un compteur qui a une valeur 0 en fin de boucle, ce qui garantit sa remise à zéro.*

## 9A Nombre d'or [constante phi], phi

Libellé : **phi**

Séquence de touches : **2nd 2nd R/S**

Le bouton **phi** permet d'appeler la constante "nombre d'or phi", qui a une valeur de 1.618033988749894848.  $\phi = (1 + \sqrt{5})/2$

## 9B Mode décimal, DEC

Libellé : **DEC**

Séquence de touches : **2nd 2nd 0**

L'instruction **DEC** fait passer le mode d'affichage en affichage décimal (le mode de calculatrice par défaut). La partie entière du nombre est affichée et saisie dans le mode décimal, chiffres 0 à 9, y compris la partie décimale du nombre. L'exposant est toujours un nombre décimal.

Le préfixe **INV** avant l'instruction **DEC** active le mode de débogage auxiliaire, où la première ligne de l'écran affiche la mantisse du nombre

affiché au format interne, sous forme de 16 chiffres hexadécimaux. De cette façon, même les décimales cachées de la mantisse peuvent être affichées. Le premier chiffre en partant de la gauche est le chiffre le plus élevé. Le bit de poids fort de la mantisse ayant une valeur de «1» est masqué et remplacé par le bit de signe. Le mode débogage est désactivé en sélectionnant le mode d'affichage sans le préfixe **INV**.

En mode exposant Eng (ingénieur), l'exposant est un multiple de 3.

## 9C Incrémentation/décrémentation dans un registre, Inc

Libellé : **Inc**

Séquence de touches : **2nd** **2nd** **|**

L'instruction **Inc** incrémente (augmente de 1) le contenu du registre de données R00 à R99 dont le numéro est précisé en paramètre de l'instruction. Si le préfixe **INV** est donné avant l'instruction Inc, l'opération inverse est effectuée : décrémenter le registre (diminuer de 1).

Placer le code **Ind** après le code d'instruction **Inc**, mais avant de spécifier le paramètre, change l'instruction en adressage indirect **Inc Ind** (voir 6B Incrémentation indirecte dans un registre Inc Ind).

Les opérations **Op 20** à **Op 3F** ont une fonction similaire, mais elles ne s'appliquent qu'aux registres R00 à R15.

*Remarque : Contrairement à la calculatrice TI-58/59 d'origine, qui utilisait l'interprétation BCD des nombres, la calculatrice ET-58 utilise un format de nombres binaires. Par conséquent, les résultats des opérations ne sont pas automatiquement arrondis aux chiffres décimaux. Cela peut se manifester de telle manière qu'après un grand nombre d'opérations répétées (par exemple des milliers d'instructions Inc), l'écart par rapport aux nombres entiers s'accumule de telle sorte que l'égalité des nombres entiers n'est pas correctement détectée. Pour cette raison, il est recommandé d'arrondir les résultats intermédiaires aux nombres entiers avec l'instruction round pour un grand nombre d'opérations avec des nombres entiers.*

## 9D Inversion de bits, NOT

Libellé : **NOT**

Séquence de touches : **2nd** **2nd** **+/-**

L'instruction NOT effectue une inversion bit à bit du nombre affiché (registre X). L'inversion de bits modifie les valeurs de bits de 0 à 1 et les valeurs de bits de 1 à 0.

système numérique autre que binaire :

Dans l'inversion au niveau du bit, il n'est pas possible de distinguer la taille de l'opérande avec laquelle l'opération doit être effectuée. Ceci est traité selon le système de numérotation en cours. Tout d'abord, l'opération de changement de signe et de décrémentation du nombre (décrémentation de 1) est effectuée. Si le mode décimal est défini ou si le résultat de l'opération est positif, aucune autre opération n'est effectuée. Dans les autres cas (base autre que 10 et nombre négatif) le résultat de l'opération est masqué pour que les chiffres tiennent sur l'afficheur.

Exemple:

**DEC** ... mode décimal

**1** **2** **3** **NOT** [-124] ... l'inverse de 123 est -124

**HEX** ... mode hexadécimal

**1** **2** **3** **NOT** [FFFFFFFFFEDC] ... l'inverse de 0x123 est 0xEDC

## 9E Pourcentage, %

Libellé : **%**

Séquence de touches : **2nd** **2nd** **=**

L'instruction **%** effectue une opération de pourcentage entre le premier et le deuxième opérande. S'il s'agit du niveau le plus bas de l'expression, le calcul peut être répété pour un autre premier opérande en appuyant

plusieurs fois sur [=].

L'instruction pourcentage [%] a deux utilisations. Il peut être saisi entre le premier et le deuxième opérande, ou utilisé avec les opérateurs + - \* : alors l'instruction [%] s'utilise après les opérandes au lieu du signe égal [=].

Exemples:

1) Avec l'opérateur somme + ajoute le pourcentage donné à la base.

1 2 3 + 4 5 [%] [178.35] ...  $123 + 45\% = 123 + 123 \cdot 45/100 = 178.35$

2) Avec l'opérateur de différence - soustrait le pourcentage donné de la base.

1 2 3 - 4 5 [%] [67.65] ...  $123 - 45\% = 123 - 123 \cdot 45/100 = 67.65$

3) Avec l'opérateur de multiplication \* calcule le pourcentage de la base.

1 2 3 \* 4 5 [%] [55.35] ...  $123 * 45\% = 123 \cdot 45/100 = 55.35$

4) Avec l'opérateur de division / calcule le pourcentage provenant de la base.

4 5 / 1 2 3 [%] [36.585...] ...  $45/123 \% = 45/123 \cdot 100 = 36.585...\%$

5) Pourcentage [%] en position d'opérateur, calcule les pourcentages à partir de la base.

1 2 3 [%] 4 5 = [55.35] ...  $123 * 45\% = 123 \cdot 45/100 = 55.35$

## 15. Opérations spéciales Op

L'instruction **Op** doit être complétée par un paramètre sous la forme d'un nombre à 2 chiffres hexadécimaux.

### Op 00 Efface les registres d'impression 1 à 4

L'opération **Op 00** réinitialise le contenu des registres d'impression 1 à 4. La réinitialisation a la même signification que le remplissage des registres avec des espaces.

Contrairement à la TI-58/59 d'origine, les registres d'impression de l'ET-58 ne sont pas partagés avec la pile opérationnelle. Ce sont des registres indépendants distincts et ne sont pas modifiés par des opérations autres que **Op 00** à **Op 04**.

### Op 01..04 Stocke dans registre d'impression 1 à 4

Les opérations **Op 01** à **Op 04** stockent le contenu de l'écran (registre X) dans les registres d'impression 1 à 4 et sont utilisées pour préparer le texte à imprimer sur l'écran ou dans un fichier d'impression. Chaque registre d'impression peut contenir jusqu'à 8 caractères. Un caractère est saisi sous forme de nombre décimal à 2 chiffres compris entre 00 et 99, 00 étant affiché sous forme d'espace.

L'éditeur de numéros permet de saisir jusqu'à 16 chiffres (correspondant à un maximum de 8 caractères). Lorsqu'il sera affiché plus tard, il n'affichera que le nombre maximum de 14 chiffres, mais ce n'est pas un problème car il conserve toujours les 16 chiffres en interne. Si les 16 chiffres ne sont pas saisis, le texte est rempli à partir de la gauche avec des espaces, jusqu'à un maximum de 8 caractères.

Contrairement à la TI-58/59 d'origine, les registres d'impression de l'ET-58 ne sont pas partagés avec la pile opérationnelle. Ce sont des registres indépendants distincts et ne sont pas modifiés par des opérations autres que **Op 00** à **Op 04**.

Pour plus de détails sur la table des caractères, voir le chapitre Table des

caractères.

Exemple "Hello World!":

**1 0 3 Op 53** ... Lit le texte "Hello" (= nombre 40 69 76 76 79)

**Op 01** ... Enregistre le texte dans le registre d'impression 1

**1 0 4 Op 53** ... Lit le texte "World" (= nombre 55 79 82 76 68)

**0 1** ... Ajoute un caractère '!' à la fin du texte

**Op 02** ... Enregistre le texte dans le registre d'impression 2

**Op 1A** ... envoie les registres 1 et 2 sur le texte de la ligne 1

**Op 1F** ... Définit le mode d'affichage du texte

... et affiche sur la 1ère ligne de l'écran : "Hello World!"

## Op 09 Charge un programme de bibliothèque

L'opération **Op 09** transfère le programme de bibliothèque sélectionné (sélectionné par l'instruction **Pgm**) vers la mémoire principale. Le transfert écrase le programme utilisateur. Si le programme de la bibliothèque est plus long que la capacité de la mémoire principale de 1000 étapes (aucun des programmes de la bibliothèque standard), la longueur du programme sera tronquée.

## Op 0A Affichage registres d'impression 1 et 2 [ligne 1]

L'opération **Op 0A** affiche le texte des registres d'impression 1 et 2 (préparés par les opérations **Op 01** et **Op 02**) sur la 1ère ligne de l'écran.

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

## Op 0B Affichage registre d'impression 1 et registre X

L'opération **Op 0B** affiche le texte du registre d'impression 1 (préparé par l'opération **Op 01**) sur la moitié gauche de la 1ère ligne de l'écran suivi du contenu du registre X (contenu d'affichage) dans la moitié droite sur 8 caractères.

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

*Remarque : Le changement du contenu du registre X ne modifiera pas le contenu de l'affichage de la première ligne.*

Exemple, affichage continu de l'état du programme:

**RST LRN** ... active le mode programmation

**CLR STO 01** ... initialise le registre de données R01

**1 0 0 Op 53** ... Lit le texte "Running" (= nombre 50 85 78 78 73 78 71)

**Op 01** ... Enregistre le texte dans le registre d'impression 1

**Lbl =** ... étiquette de début de boucle

**Inc 01 RCL 01** ... incrémente le registre R01

**Op 0B** ... affiche le texte et registre X sur la 1ère ligne

**GTO =** ... boucle

**LRN** ... sortie du mode programmation

**RST R/S** ... lancer le programme

... [Running 123]

... le programme incrémente X et imprime en continu l'état sur la 1ère ligne



## Op 0C Affichage registre d'impression 1 et registre X

L'opération **Op 0C** affiche le texte de la première moitié du registre d'impression 1 (préparé par l'opération **Op 01**) sur le premier quart de la 1ère ligne de l'afficheur suivi du contenu du registre X (contenu d'affichage).

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

*Remarque : Le changement du contenu du registre X ne modifiera pas le contenu de l'affichage de la première ligne.*

## Op 0D Affichage registres d'impression 3 et 4 [ligne 2]

L'opération Op 0D affiche le texte des registres d'impression 3 et 4 (préparés par les opérations Op 03 et Op 04) sur la 2ème ligne de l'écran.

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

## Op 0E Affichage registre d'impression 3 et registre X

L'opération **Op 0E** affiche le texte du registre d'impression 1 (préparé par l'opération **Op 03**) sur la moitié gauche de la 1ère ligne de l'écran suivi du contenu du registre X (contenu d'affichage) dans la moitié droite sur 8 caractères.

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

*Remarque : Le changement du contenu du registre X ne modifiera pas le contenu de l'affichage de la première ligne.*

## Op 0F Affichage registre d'impression 3 et registre X

L'opération **Op 0F** affiche le texte de la première moitié du registre d'impression 1 (préparé par l'opération **Op 03**) sur le premier quart de la 1ère ligne de l'afficheur suivi du contenu du registre X (contenu d'affichage).

Il s'agit d'un texte affiché uniquement pendant l'exécution du programme. L'arrêt du programme efface le contenu du texte, qui est le contenu du texte par défaut pendant l'exécution du programme.

*Remarque : Le changement du contenu du registre X ne modifiera pas le contenu de l'affichage de la première ligne.*

## Op 10 Test de signe

L'opération **Op 10** effectue un test du signe du contenu du registre X (affichage). Si le contenu du registre est inférieur à 0, le résultat de l'opération sera -1. Si le contenu du registre est supérieur à 0, le résultat sera 1. Si le contenu du registre est 0, il restera 0.

## Op 11 Variance

L'opération **Op 11** calcule la variance des données statistiques saisies à l'aide de l'instruction **Stat**. Pour le calcul, il utilise le contenu des registres de données R01 à R06, qui ont été mis à jour par l'instruction **Stat**.

Dans le registre T (registre auxiliaire affiché en appuyant sur **x<>t**) le calcul stocke la variance de la variable X, selon la formule  $\text{var}X = \text{sum}(x^2)/N - (\text{sum}(x)/N)^2$ .

Dans le registre X (affichage), le calcul stocke la variance de la variable Y selon la formule  $\text{var}Y = \text{sum}(y^2)/N - (\text{sum}(y)/N)^2$ .

Exemple:

**INV CMs** ... efface les registres statistiques R01...R06 et les registres X et

T

**2** **3** **Stat** **4** **5** **Stat** **6** **7** **Stat** ... insère les données pour Y (pour X = 0, 1, 2)

**Op** **11** [322.666...] ... variance Y = 322.666...

**x<>t** [.6666..] ... variance X = 0.6666...

Exemple 2:

... registres statistiques remplis selon les données de l'exemple **Op** **12**

**Op** **11** [242.4722...]

**x<>t** [4.1822...]

## Op 12 Coefficients de régression linéaire

L'opération **Op** **12** calcule les coefficients de la droite de régression linéaire à l'aide de la méthode des moindres carrés, calcul basé sur les paires de valeurs (X, Y), saisies à l'aide de la fonction statistique **Stat**. La droite de régression a la forme  $y = m \cdot x + b$ .

Dans le registre T (registre auxiliaire affiché en appuyant sur **x<>t**) le calcul stocke le coefficient 'm' selon la formule  $m = (\text{sum}(x \cdot y) - \text{sum}(x) \cdot \text{sum}(y) / N) / (\text{somme}(x^2) - \text{somme}(x)^2 / N)$ .

Dans le registre X (affichage), le calcul stocke le coefficient 'b' selon la formule  $b = (\text{somme}(y) - m \cdot \text{somme}(x)) / N$ .

Exemple:

**INV** **CMS** ... efface les registres statistiques R01...R06 et les registres X et T

**1** **0** **1** **|** **3** **x<>t** **6** **0** **9** **Stat** ... résultat 1 (101.3, 609)

**1** **0** **3** **|** **7** **x<>t** **6** **2** **6** **Stat** ... résultat 2 (103.7, 626)

**9** **8** **|** **6** **x<>t** **5** **8** **6** **Stat** ... résultat 3 (98.6, 586)

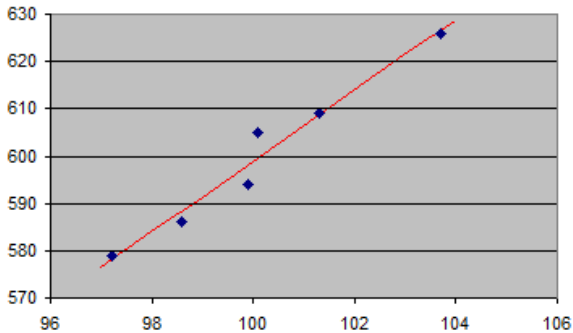
9 9 | 9 x<>t 5 9 4 Stat ... résultat 4 (99.9, 594)

9 7 | 2 x<>t 5 7 9 Stat ... résultat 5 (97.2, 579)

1 0 0 | 1 x<>t 6 0 5 Stat ... résultat 6 (100.1, 605)

Op 12 [-148.506...] ... coefficient b = -148.506...

x<>t [7.4734...] ... coefficient m = 7.4734...



## Op 13 Coefficient de corrélation

Le coefficient de corrélation décrit la dépendance linéaire mutuelle de deux quantités. Il prend des valeurs de -1 à +1. Une valeur de -1 signifie que Y dépend négativement et linéairement de X (l'augmentation de X diminue Y). Une valeur de +1 représente une relation linéaire positive. Une valeur de 0 signifie que les quantités ne dépendent pas les unes des autres.

L'opération Op 13 calcule le coefficient de corrélation à partir des données spécifiées par la fonction statistique Stat.

Le coefficient de corrélation est calculé à partir de la relation  $R = m * \text{devx}/\text{devy}$ .

Le coefficient de pente de la droite de régression  $m = (\text{sum}(x*y) - \text{sum}(x)*\text{sum}(y)/N) / (\text{sum}(x^2) - \text{sum}(x)^2/N)$ .

Écart type pour X  $\text{devx} = \sqrt{(\text{sum}(x^2) - \text{sum}(x)^2/N)/(N-1)}$ .

Écart type pour Y  $\text{devy} = \sqrt{(\text{sum}(y^2) - \text{sum}(y)^2/N)/(N-1)}$ .

Soit calcul de  $R = (\text{sum}(x*y) - \text{sum}(x)*\text{sum}(y)/N) / \sqrt{(\text{sum}(x^2) - \text{sum}(x)^2/N * (\text{somme}(y^2) - \text{somme}(y)^2/N))}$ .

Exemple:

... registres statistiques remplis selon les données de l'exemple **Op 12**

**Op 13** [0.0051370...]

## Op 14 Régression linéaire de Y sur X

L'opération **Op 14** calcule la valeur de Y à partir de la valeur de X en utilisant une droite de régression linéaire, selon la formule  $y = m*x + b$ . Voir Op 12 Coefficients de régression linéaire.

Exemple:

... registres statistiques remplis selon les données de l'exemple **Op 12**

**9 7 Op 14** [576.4166...] ... pour X=97 calcul de Y=576.4166...

**1 0 4 Op 14** [628.7306...] ... pour X=104 calcul de Y=628.7306...

## Op 15 Régression linéaire de X sur Y

L'opération **Op 15** calcule la valeur X à partir de la valeur Y à l'aide d'une droite de régression linéaire, selon la formule  $x = (y - b)/m$ . Voir Op 12 Coefficients de régression linéaire.

Exemple:

... registres statistiques remplis selon les données de l'exemple **Op 12**

**5 8 0 Op 15** [97.4794...] ... pour Y=580 calcul de X=97.4794...

**6 3 0 Op 15** [104.170...] ... pour Y=630 calcul de X=104.170...

## Op 16, Op 17 Gestion de la mémoire [inopérant]

Les opérations **Op 16** et **Op 17** sont utilisées dans la TI-58/59 d'origine pour définir la partition de mémoire RAM entre la mémoire programme et la mémoire des registres de données. Avec l'ET-58, la partition ne peut pas être définie et correspond toujours à un maximum de 1000 étapes de programme (en mémoire EEPROM) et de 100 registres de données ou plus (en mémoire RAM). Les opérations **Op 16** et **Op 17** afficheront uniquement le numéro 999.99.

## Op 18 Lève le drapeau 7 si pas d'erreur

Opération **Op 18**, lève le drapeau 7 dans le cas où aucune erreur E n'est rencontrée. Dans le cas contraire, l'état du drapeau 7 reste inchangé. L'état du drapeau peut être testé avec l'instruction **IfFlg**. Une autre option pour tester la condition d'erreur est le drapeau F, qui est directement lié à l'indication d'erreur E.

## Op 19 Lève le drapeau 7 si erreur

Opération **Op 19**, lève le drapeau 7 dans le cas où une erreur E est rencontrée. Dans le cas contraire, l'état du drapeau 7 reste inchangé. L'état du drapeau peut être testé avec l'instruction **IfFlg**. Une autre option pour tester la condition d'erreur est le drapeau F, qui est directement lié à l'indication d'erreur E.

## Op 1A Affichage registres impression 1 et 2 [à l'arrêt]

L'opération **Op 1A** affiche le texte des registres d'impression 1 et 2 (préparés par les opérations **Op 01** et **Op 02**) sur la 1ère ligne de l'écran.

C'est le texte affiché si le programme n'est pas en cours d'exécution et si le mode texte est actif, activé par l'instruction **Op 1F**. Le mode texte peut être désactivé avec la touche **CLR**.

**Exemple** - voir Op 01...04 Stocke dans registre d'impression 1 à 4

## Op 1B Affichage registre d'impression 1 et registre X

L'opération **Op 1B** imprime le texte du registre d'impression 1 (préparé par l'opération **Op 01**) sur la moitié gauche de la 1ère ligne de l'écran suivi du contenu du registre X (contenu d'affichage) dans la moitié droite sur 8 caractères.

Le texte est affiché si le programme n'est pas en cours d'exécution et si le mode texte est activé par l'instruction **Op 1F**. Ce mode texte peut être désactivé avec la touche **CLR**.

*Remarque : Le changement du contenu du registre X n'affectera pas le contenu de l'affichage de la première ligne.*

## Op 1C Affichage registre d'impression 1 et registre X

L'opération **Op 1C** affiche le texte de la première moitié du registre d'impression 1 (préparé par l'opération **Op 01**) sur le premier quart de la 1ère ligne de l'afficheur suivi du contenu du registre X (contenu d'affichage).

Le texte est affiché si le programme n'est pas en cours d'exécution et si le mode texte est activé par l'instruction **Op 1F**. Ce mode texte peut être désactivé avec la touche **CLR**.

*Remarque : Le changement du contenu du registre X n'affectera pas le contenu de l'affichage de la première ligne.*

## Op 1D Active le mode d'affichage 'Indicateurs' sur 1ère ligne

L'opération **Op 1D** active le mode d'affichage de la 1ère ligne de l'afficheur avec les indicateurs (voir chapitre 8 Indicateurs à l'écran). Il s'agit de l'état par défaut après la mise sous tension de la calculatrice. Si ce mode est sélectionné avant d'activer le mode 'Texte' avec **Op 1F**, il est restauré en appuyant sur le bouton **CLR**.

## Op 1E Active le mode d'affichage 'Registre T' sur 1ère ligne

L'opération **Op 1E** active le mode d'affichage de la 1ère ligne de l'écran avec le registre T. Les deux nombres (T et X) sont affichés sur les deux lignes de l'écran et peuvent être utilisés, par exemple, pour les nombres complexes. Si ce mode est sélectionné avant d'activer le mode 'Texte' avec **Op 1F**, il est restauré en appuyant sur le bouton **CLR**.

## Op 1F Active le mode d'affichage 'Texte' sur 1ère ligne

L'opération **Op 1F** active le mode d'affichage avec texte pour la 1ère ligne de l'afficheur. Le texte peut être affiché sur une ligne en utilisant les opérations **Op 1A** à **Op 1C**. Il s'agit du texte affiché uniquement lorsque le programme est arrêté. Le bouton **CLR** peut être utilisé pour désactiver le mode texte de l'écran - le mode **Op 1D** avec commutateurs ou le mode **Op 1E** avec registre T revient alors, en fonction du dernier mode actif.

La désactivation du mode texte ne modifie pas le contenu de la ligne de texte. Le contenu de la ligne peut être préparé à l'avance et affiché en activant le mode texte de l'afficheur uniquement lorsque cela est nécessaire.

**Exemple** - voir Op 01...04 Stocke dans registre d'impression 1 à 4

## Op 20 à Op 2F Incrémentation du registre R00 à R15

Les opérations **Op 20** à **Op 2F** augmentent le contenu du registre de données R00 à R15 de 1 (incrément). Il s'agit d'une opération compatible avec la TI-58/59 d'origine. Avec l'ET 58, il est plus approprié d'utiliser l'instruction **Inc**, qui s'applique à tous les registres et qui permet l'adressage indirect.

## Op 30 à Op 3F Décrémentation du registre R00 à R15

Les opérations **Op 30** à **Op 3F** diminuent le contenu du registre de données R00 à R15 de 1 (décrément). Il s'agit d'une opération compatible



avec la TI-58/59 d'origine. Avec l'ET 58, il peut être plus approprié d'utiliser l'instruction **INV Inc**, qui s'applique à tous les registres et qui permet l'adressage indirect.

## Op 40 Affiche le code de la touche

L'opération **Op 40** affiche le code de la touche enfoncée dans le registre X (affichage). Chaque touche a une valeur de 0 à 254 et correspond au code du bouton dans le système numérique décimal. S'il n'y a aucun caractère prêt dans le tampon du clavier, le nombre 255 est renvoyé. Les codes des boutons prennent en compte le préfixe **2nd**.

Exemple - teste les touches enfoncées:

**RST LRN** ... activation du mode programmation

**HEX** ... passer au système de numérotation HEX (pour une lisibilité plus facile des codes)

**0F 0F x<>t** ... un code "invalide" est stocké dans le registre T 255=FF

**Lbl** **=** ... étiquette de début de boucle

**Op 40** ... l'opération pour récupérer la touche du clavier

**x=t** **=** ... si le code 255 est renvoyé, rien n'est appuyé, retour en boucle

**Pause** ... code de la touche enfoncée

**GTO** **=** ... continuer en boucle

**LRN** ... sortie du mode programmation

**RST R/S** ... lancer le programme

... Lorsqu'un bouton est enfoncé, son code hexadécimal s'affiche sur l'écran (sauf pour les boutons **2nd**, **R/S**). Arrêt du programme avec **R/S**.

## Op 41 Test d'appui sur une touche

L'opération **Op 41** vous permet de tester si le bouton souhaité est enfoncé. La coordonnée du bouton est saisi comme paramètre d'entrée de

l'opération. Comme coordonnée le premier chiffre (supérieur) représente la ligne du clavier 1 à 9, le deuxième chiffre (inférieur) est la colonne du clavier 1 à 5. Le système de coordonnées n'est pas concerné par la saisie préalable du préfixe. Pour l'exemple, le bouton **1** a pour coordonnée 82, ce qui signifie ligne 8 colonne 2. L'opération renvoie une valeur de 1 si la coordonnée du bouton enfoncé correspond à la valeur de test. Renvoie 0 si aucun appui correspondant à la valeur de test.

Exemple - Test d'appui sur le bouton **EE** (affiche 1 si enfoncé) :

**RST** **LRN** ... activation du mode programmation

**HEX** ... passer au système de numérotation HEX

**Lbl** **=** ... étiquette de début de boucle

**5** **2** **Op** **41** ... Test de pression sur EE, 5ème ligne et 2ème colonne

**Pause** ... affichage d'état 1 (enfoncé) ou 0 (non enfoncé)

**GTO** **=** ... pokračování ve smyčce

**LRN** ... sortie du mode programmation

**RST** **R/S** ... lancer le programme

... Affiche 1 si **EE** est enfoncé, sinon affiche 0.

Arrête du programme avec **R/S**.

## Op 42 Affiche un caractère à l'écran

L'opération **Op** **42** affiche 1 caractère sur l'écran. Le caractère est écrit dans les tampons d'impression et y reste aussi longtemps que le tampon d'impression correspondant est valide. En entrée de l'opération, le registre X (contenu de l'affichage) contient un code de caractère à écrire de 00 à 99 (voir la Table des Caractères). Dans le registre T, la position d'affichage est de 0 à 47.

Les nombres 0 à 15 représentent la position dans la 1ère ligne de l'affichage si le programme est en cours d'exécution. Les nombres 16 à 31 représentent la position +15 dans la 2ème ligne de l'afficheur si le programme est en cours d'exécution. Ces deux lignes sont réinitialisées à

la valeur par défaut lorsque le programme est arrêté.

Les nombres 32 à 47 représentent la position +32 dans la 1ère ligne en cas d'arrêt du programme. Cette ligne n'est visible que si le mode affichage texte est activé avec **Op 1F** (le mode est désactivé en appuyant sur **CLR**).

Exemple:

**RST LRN** ... activation du mode programmation

**Lbl A** ... étiquette début de programme

**1 5 x<>t** ... la position en fin de 1ère ligne est préparée dans le registre T

**1 0 Op 42** ... le caractère \* est affiché à la fin de la ligne

**4 7 x<>t** ... la position en fin de 1ère ligne est préparée dans le registre T

**2 9 Op 42** ... le caractère = est affiché en fin de ligne

**Op 1F** ... le mode texte est activé pour affichage après exécution

**2 0 0 Op 44** ... délai de 2 secondes

**RTN** ... fin de programme (**INV SBR**)

**LRN** ... sortie du mode programmation

**A** ... test du programme : Il fonctionne pendant environ 2 secondes après chaque pression et affiche un astérisque \* à la fin de la 1ère ligne pendant ce laps de temps. Lorsqu'il est terminé, le signe égal = apparaîtra à l'écran. Le mode caractère peut être annulé en appuyant sur **CLR**.

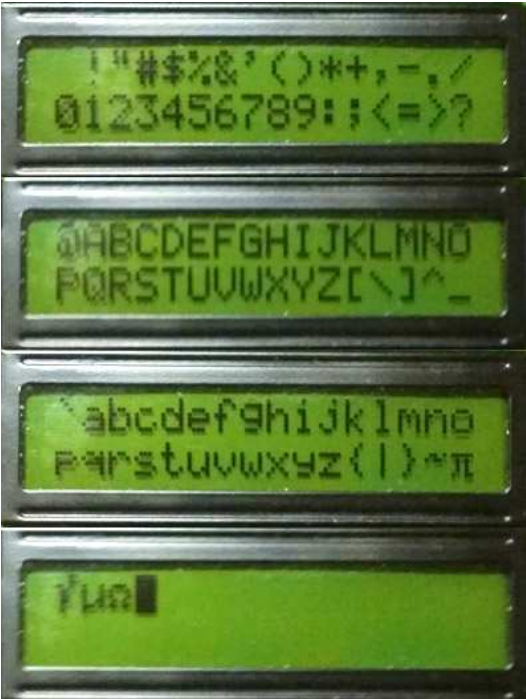
## Op 43 Chargement de la police

La calculatrice permet de redéfinir 8 caractères de l'afficheur LCD, avec le code 92 à 99. L'opération **Op** **43** redéfinit les caractères de la police sélectionnée en fonction du nombre affiché :

- 0 ... police par défaut
- 1 ... colonne de gauche
- 2 ... colonne de droite
- 3 ... lignes et graphiques
- 4 ... pixels

Dans le cas des polices 1 à 4, la barre oblique inverse 60 \ est remplacée par une ligne verticale |. Les polices peuvent être affichées avec le programme **A** de la bibliothèque ML-01.

Police standard par défaut « 0 », caractères 00 à 99



Police pour la colonne de gauche '1', caractères redéfinis 92 à 99



Police de la colonne de droite « 2 », caractères redéfinis 92 à 99



Police pour lignes et graphiques '3', caractères redéfinis 92 à 99



Police pour les pixels '4', caractères redéfinis 92 à 99



Exemple - affichage des tables de polices:

Contrôles : sélection de polices de 0 à 4, défilement des polices SST et BST, fin R/S.

**RST** **LRN** ... activation du mode programmation

**CLR STO 01** ... initialisation registre R01  
**Lbl =** ... début de l'affichage d'une page de caractères  
**3 2 STO 02** ... compteur de caractères affichés dans registre R02  
**CP** ... position initiale du caractère 0 dans le registre T  
**RCL 01** ... premier caractère à afficher  
**Lbl x^2** ... début de la boucle d'affichage des caractères  
**Op 42** ... affiche le caractère X à la position T  
**+ 1 =** ... incrément numéro de caractère  
**x<>t + 1 = x<>t** ... incrément de la position  
**Dsz 2 x^2** ... boucle d'affichage de caractères  
**Lbl Inx** ... début de la boucle d'attente  
**Op 40 x<>t** ... saisie d'un code clavier dans le registre T  
**6 5 INV x=t STO** ... teste si code clavier **SST**  
**RCL 01 + 3 2 =** ... augmente l'index de page  
**AND 1 2 7 = STO 01** ... calcul plage d'index de caractères  
**GTO =** ... voir une nouvelle page  
**Lbl STO** ... traitement si pas de saisie **SST**  
**8 1 INV x=t RCL** ... teste si code clavier **BST**  
**RCL 01 + 9 6 =** ... diminue l'index de page  
**AND 1 2 7 = STO 01** ... calcul plage d'index de caractères  
**GTO =** ... voir une nouvelle page  
**Lbl RCL** ... traitement si pas de saisie **BST**  
**4 INV x>=t Inx** ... test code clavier **0...4**  
**x<>t Op 43** ... chargement police **0...4**  
**GTO =** ... voir nouvelle page

**LRN** ... sortir du mode programmation

**RST** **R/S** ... test du programme, utilisation **SST**, **BST**, **0**...**4**, fin : **R/S**

## Op 44 Pause paramétrée

L'opération **Op 44** crée une pause dans le programme pour une période de 0 à 255 spécifiée dans le registre X (nombre à l'écran) sous la forme d'un multiple de 10 ms (c'est-à-dire max. 2,55 secondes). En raison du retard d'exécution du programme, le temps obtenu peut légèrement différer de 10 à 20 %.

## Op 45 Afficher le pointeur à gauche de la 1ère ligne

L'opération **Op 45** charge la police numéro 1 (colonne de gauche) dans l'écran LCD et affiche le pointeur à partir de la gauche (barre de progression) sur la 1ère ligne pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 80.

Exemple - barre de progression sur la 1ère ligne:

**RST** **LRN** ... active le mode de programmation

**CLR** ... valeur initiale du pointeur = 0

**Lbl** **STO** ... début de la boucle de traitement

**Op 45** ... affiche le pointeur

**+ 1 =** ... incrément du pointeur

**mod 8 0 =** ... ramène à l'intervall 0 à 80

**Op 80** ... court délai de 10 ms

**GTO** **STO** ... boucle

**LRN** ... sortie du mode programmation

**RST** **R/S** ... test du programme, défilement de la barre de progression



## Op 46 Afficher le pointeur à gauche de la 2ème ligne

L'opération **Op 46** charge la police numéro 1 (colonne de gauche) dans l'écran LCD et affiche le pointeur à partir de la gauche (barre de progression) sur la 2ème ligne pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 80.

Exemple - barre de progression sur la 2ème ligne:

**RST** **LRN** ... active le mode de programmation

**CLR** ... valeur initiale du pointeur = 0

**Lbl** **STO** ... début de la boucle de traitement

**Op 46** ... affiche le pointeur

**+ 1 =** ... incrément du pointeur

**mod 8 0 =** ... ramène à l'intervall 0 à 80

**Op 80** ... court délai de 10 ms

**GTO** **STO** ... boucle

**LRN** ... sortie du mode programmation

**RST** **R/S** ... test du programme, défilement de la barre de progression

## Op 47 Afficher le texte avec le pointeur à gauche à l'arrêt

L'opération **Op 47** charge la police numéro 1 et active le texte dans la 1ère moitié de la ligne 1 puis affiche le pointeur en partant de la gauche (barre de progression) sur la 2ème moitié de la ligne 1 pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 40.

A l'arrêt du programme la ligne 1 reste affichée (idem **OP 1F**).

Exemple - texte avec barre de défilement :

**RST** **LRN** ... active le mode de programmation

**1 0 6 Op 53 Op 01** ... charge "Progress" dans le registre d'impression 1

**CLR** ... valeur initiale du pointeur = 0

**Lbl STO** ... début de la boucle de traitement

**Op 47** ... affiche le texte et le pointeur

**+ 1 =** ... incrément du pointeur

**mod 4 0 =** ... ramène à l'intervall 0 à 40

**Pause** ... pause d'environ 250 ms

**GTO STO** ... boucle

**LRN** ... sortie du mode programmation

**RST R/S** ... test du programme, défilement de la barre de progression

le programme peut être interrompu avec **R/S** puis relancé avec **R/S** sans effacement de la ligne 1.

## Op 48 Afficher le pointeur à droite de la 1ère ligne

L'opération **Op 48** charge la police numéro 2 (colonne de droite) dans l'écran LCD et affiche le pointeur à partir de la droite (barre de progression) sur la 1ère ligne pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 80.

Exemple - barre de progression sur la 1ère ligne:

**RST LRN** ... active le mode de programmation

**CLR** ... valeur initiale du pointeur = 0

**Lbl STO** ... début de la boucle de traitement

**Op 48** ... affiche le pointeur

**+ 1 =** ... incrément du pointeur

**mod 8 0 =** ... ramène à l'intervall 0 à 80

**Op 80** ... court délai de 10 ms

**GTO** **STO** ... boucle

**LRN** ... sortie du mode programmation

**RST** **R/S** ... test du programme, défilement de la barre de progression

## Op 49 Afficher le pointeur à droite de la 2ème ligne

L'opération **Op** **49** charge la police numéro 2 (colonne de droite) dans l'écran LCD et affiche le pointeur à partir de la droite (barre de progression) sur la 2ème ligne pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 80.

Exemple - barre de progression sur la 1ère ligne:

**RST** **LRN** ... active le mode de programmation

**CLR** ... valeur initiale du pointeur = 0

**Lbl** **STO** ... début de la boucle de traitement

**Op** **49** ... affiche le pointeur

**+** **1** **=** ... incrément du pointeur

**mod** **80** **=** ... ramène à l'intervall 0 à 80

**Op** **80** ... court délai de 10 ms

**GTO** **STO** ... boucle

**LRN** ... sortie du mode programmation

**RST** **R/S** ... test du programme, défilement de la barre de progression

## Op 4A Afficher le texte avec le pointeur à droite à l'arrêt

L'opération **Op** **4A** charge la police numéro 2 et active le texte dans la 1ère moitié de la ligne 1 puis affiche le pointeur en partant de la droite (barre de progression) sur la 2ème moitié de la ligne 1 pendant l'exécution du programme. L'entrée est la valeur X (contenu affiché) = 0 à 40.

A l'arrêt du programme la ligne 1 reste affichée (idem **OP 1F**).

Exemple - texte avec barre de défilement :

**RST LRN** ... active le mode de programmation

**1 0 6 Op 53 Op 01** ... charge "Progress" dans le registre d'impression 1

**CLR** ... valeur initiale du pointeur = 0

**Lbl STO** ... début de la boucle de traitement

**Op 4A** ... affiche le texte et le pointeur

**+ 1 =** ... incrément du pointeur

**mod 4 0 =** ... ramène à l'intervall 0 à 40

**Pause** ... pause d'environ 250 ms

**GTO STO** ... boucle

**LRN** ... sortie du mode programmation

**RST R/S** ... test du programme, défilement de la barre de progression

le programme peut être interrompu avec **R/S** puis relancé avec **R/S** sans effacement de la ligne 1



## Op 4B Affichage d'un graphique à barres lors de l'exécution

L'opération **Op 4B** charge la police numéro 3 (lignes) dans l'écran LCD et affiche une colonne graphique avec la valeur 0 à 16 selon le registre X (nombre sur l'écran) et la position 0 à 15 selon le registre T, pendant l'exécution du programme.

Exemple - onde sinusoïdale animée

**RST** **LRN** ... active le mode de programmation

**CLR** **STO** **01** ... initialisation registre R01

**Deg** ... unité de calcul de l'angle

**Lbl** **=** ... début de l'affichage du graphique

**CP** ... initialisation registre T

**Lbl** **+** ... début du cycle de l'affichage d'une colonne

**RCL** **01** ... valeur de la sinusoïde

**sin** **+** **1** **=** **\*** **8** **=** **round** ... valeur de l'onde comprise entre 0 et 16

**Op** **4B** ... affichage d'une colonne du graphique

**RCL** **01** **+** **2** **2** **.** **5** **=** **STO** **01** ... progression de l'onde sinusoïdale

**X/T** **+** **1** **=** **X/T** ... incrémenter la position sur l'écran

**1** **6** **INV** **x=t** **+** ... test de position pour continuer avec la colonne suivante

**RCL** **01** **+** **2** **2** **.** **5** **=** **STO** **01** ... progression de l'onde sinusoïdale

**GTO** **=** ... boucle

**LRN** ... quitter le mode programmation

**RST** **R/S** ... démarrage du programme, l'onde sinusoïdale se déplace de droite à gauche



## Op 4C Affichage d'un graphique à barres avec texte

L'opération **Op** **4C** charge la police numéro 3 et active le texte dans la 1ère moitié de la ligne 1 puis affiche une colonne graphique de valeur 0 à 8 en partant de la droite sur la 2ème moitié de la ligne 1 pendant l'exécution du programme.

A l'arrêt du programme la ligne 1 reste affichée (idem **OP 1F**).

### Exemple - sinusoïde

**RST LRN** ... active le mode de programmation

**CLR STO 01** ... initialisation registre R01

**1 0 7 Op 53 0 0 Op 01** ... charge "Graph" dans le registre d'impression  
1

**Deg** ... unité de calcul de l'angle

**CP** ... initialisation registre T

**Lbl +** ... début du cycle de l'affichage d'une colonne

**RCL 01** ... valeur de la sinusoïde

**sin + 1 = 4 = round** ... valeur de l'onde comprise entre 0 et 8

**Op 4C** ... affichage d'une colonne du graphique avec le texte

**RCL 01 + 4 5 = STO 01** ... progression de la sinusoïde

**X/T + 1 = X/T** ... incrémenter la position sur l'écran

**8 INV x=t +** ... test de position pour continuer avec la colonne suivante

**R/S** ... fin du programme

**LRN** ... quitter le mode programmation

**RST R/S** ... démarrage du programme puis fin, l'onde sinusoïdale s'affiche.



## Op 4D Paramètres de pixels

L'opération **Op 4D** charge la police numéro 4 (pixels) dans l'écran LCD et définit les coordonnées des pixels d'affichage de 0 à 15 horizontalement selon le registre X (sur l'écran) et de 0 à 5 verticalement selon le registre T.

## Op 4E Effacer un pixel

L'opération **Op 4E** charge la police numéro 4 (pixels) dans l'écran LCD et supprime un pixel sur l'écran au moment de l'exécution avec les coordonnées 0 à 15 horizontalement selon le registre X (sur l'écran) et 0 à 5 verticalement selon le registre T.

## Op 4F Commutation de pixels

L'opération **Op 4F** charge la police numéro 4 (pixels) dans l'écran LCD et allume (= allume ou éteint) en cours d'exécution un pixel sur l'écran avec les coordonnées 0 à 15 horizontalement selon le registre X (sur l'écran) et 0 à 5 verticalement selon le registre T.

Exemple - pixels aléatoires:

**RST LRN** ... activation du mode programmation

**6 INV rand Int X/T** ... coordonnée verticale aléatoire 0...5

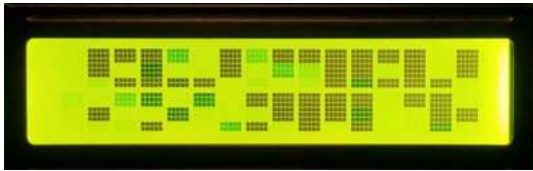
**1 6 INV rand Int** ... coordonnée horizontale aléatoire 0...15

**Op 4F** ... commutation de pixels

**RST** ... répétition

**LRN** ... sortir du mode programmation

**RST R/S** ... démarrage du programme, les pixels clignotent de manière aléatoire sur l'écran



## Op 50 Trouver le plus grand dénominateur commun

L'opération **Op 50** trouve le plus grand commun diviseur de deux entiers non nuls X (affichage) et T, selon l'algorithme euclidien. Stocke le résultat dans le registre X (affichage). Cette opération est utilisée pour simplifier des fractions.

Exemple, simplifier la fraction 260/340

**2 6 0 X/T** ... stocker le premier nombre dans le registre T

**3 4 0** ... stocker le deuxième nombre dans le registre X

**Op 50** [20] ... trouver le plus grand diviseur commun = 20

**2 6 0 / 2 0 =** [13] ... le numérateur de la fraction est 13

**3 4 0 / 2 0 =** [17] ... le dénominateur de la fraction est 17

... la fraction simplifiée est 13/17.

## Op 51 Rappel du registre du générateur aléatoire

L'opération **Op 51** affiche (registre X) la valeur du registre générateur aléatoire interne (*seed*). Le registre est un entier de taille DWORD et avec une plage de valeurs de 0 à 4294967295.

## Op 52 Chargement du registre du générateur aléatoire

L'opération **Op 52** charge le registre générateur aléatoire interne (*seed*) avec la valeur du registre d'affichage X. Le registre est un entier DWORD



avec une plage de valeurs de 0 à 4294967295. Un caractère aléatoire reproductible peut être obtenu en définissant le registre.

Remarque : Le registre du générateur aléatoire est stocké dans la mémoire EEPROM à chaque réinitialisation de la calculatrice et ainsi le caractère aléatoire des valeurs générées est assuré presque sans répétition même avec des démarrages répétés de la calculatrice. Il n'est pas recommandé de définir le registre du générateur d'aléatoire, car cela perdrait le bénéfice de l'aléatoire.

## Op 53 Charger le texte prédéfini

L'opération **Op 53** charge la valeur du texte prédéfini (dans la table interne de la ROM) dans le registre X (affichage). La longueur du texte est max. 8 caractères, soit 16 chiffres au format interne (voir Tableau des caractères). Avant l'opération, il est nécessaire de saisir le numéro du texte dans le registre X (affichage) selon le tableau suivant. Il est possible d'ajouter un point décimal et un chiffre décimal après le numéro du texte - dans ce cas, le texte chargé sera décalé vers la gauche du nombre de positions (espaces) donné par le chiffre décimal. Le texte est renvoyé dans un état modifiable - il est possible d'y ajouter des caractères supplémentaires.

(Voir Op 01..04 Stocke dans registre d'impression 1 à 4.)

0 OK	28 Divide	56 ArcLen	84 Your
1 ERROR	29 Power	57 Chord	85 You
2 Diagnose	30 Root	58 S-Area	86 My
3 Result	31 Square	59 Low	87 Lost
4 Input	32 Key	60 High	88 Count
5 Enter	33 Button	61 Too Low	89 Counter
6 Index	34 (1=YES)	62 Correct	90 Calc
7 Row	35 Vector	63 Too High	91 Calcul
8 Column	36 Library	64 Game	92 Calculer
9 Continue	37 First	65 Time	93 Won
10 STOP	38 Second	66 Working	94 Wait
11 Matrix	39 Third	67 ...	95 Press
12 Complex	40 Fourth	68 Win	96 a Key
13 Number	41 Param.	69 Loss	97 Reaction
14 Element	42 Entry	70 Bankroll	98 Response
15 Item	43 Output	71 Success	99 Alien

16 Print	44 Rows	72 Crash	100 Running
17 from	45 Columns	73 Fail	101 Test
18 to	46 Size	74 Speed	102 Help
19 Display	47 Ready	75 Check	103 Hello
20 Program	48 Lambda	76 Landing	104 World
21 Load	49 Polynom	77 Mission	105 ET-58
22 Save	50 Angle	78 Complete	106 Progress
23 YES	51 Side	79 Failure	107 Graph
24 NO	52 Triangle	80 Smooth	108 Black
25 Add	53 Area	81 Turn	109 White
26 Subtract	54 Perimet	82 Turns	110 Height
27 Multiply	55 Radius	83 Computer	111 Width

*Remarque : L'opération **Op 53** laisse le nombre dans l'état d'entrée (partie entière) où des caractères de texte supplémentaires peuvent être ajoutés. Cependant, si vous effectuez d'autres opérations numériques avec le nombre (par exemple, sauvegarde dans le registre et rechargement), le nombre peut apparaître sous une forme différente, avec des décimales ou avec un exposant - dans ce cas, il ne sera pas possible de continuer à ajouter des caractères à la partie entière.*

## Op 54 Ajouter un nombre au texte

L'opération **Op 54** peut être utilisée pour ajouter un entier signé contenu dans le registre T à un texte à afficher. Cette opération est utilisée pour préparer un texte suivi d'un nombre qui peut être variable. L'édition du nombre reste modifiable, afin que des caractères supplémentaires puissent être ajoutés. (voir exemple ci-dessous)

*Remarque : L'opération **Op 54** désactive le mode exposant EE et l'exposant technique Eng, car l'exposant est incompatible avec cette opération. Le nombre ajouté au texte sera affiché sans décimales avec un éventuel arrondi.*

Exemple invite « Entrez a[i] : »

**RST** **LRN** ... activation du mode programmation

**Lbl** **A** ... étiquette du programme pour afficher l'invite

**X/T** ... stocke l'index dans le registre T

**5 Op 53 Op 01** ... charge "Enter" dans le registre d'impression 1

**6 5 5 9** ... ajoute le texte "a["

**Op 54** ... ajoute l'index à partir du registre T

**6 1 2 6 0 0 Op 02** ... ajoute le texte "]: " et enregistre dans le registre d'impression 2

**Op 1A** ... prépare les registres d'impression pour la 1ère ligne

**0** ... 0 est affiché à l'écran

**Op 1F** ... active le mode d'affichage de texte

**RTN** ... fin du sous-programme (INV SBR)

**LRN** ... sortie du mode programmation

**2 3 A** ... test : la 1ère ligne affiche l'invite "Enter a[23] :"

## Op 55 Initialiser la pile de nombres complexes ou de fractions

Les nombres complexes et les fractions sont des paires de nombres. Pendant le calcul, les opérands sont stockés dans la pile dans les registres de données et traités en mode RPN « Reverse Polish Notation ». Cela signifie que les opérands sont d'abord stockés dans la pile de nombres, puis que l'opération correspondante est effectuée entre eux.

L'opération **Op 55** définit d'abord la pile dans les registres de données avant d'utiliser pour la première fois des nombres complexes ou des fractions. Le registre T contient l'index du premier registre de données de la pile, le registre X (sur l'écran) contient le nombre de paires de nombres dans la pile. Les nombres étant stockés par paires, donc le nombre total de registres sera le double du nombre spécifié. Implicitement (lorsque **Op 55** n'est pas utilisé), la pile est définie pour 10 paires de nombres à partir du registre R10, donc jusqu'au registre R29 (c'est-à-dire comme si la séquence **1 0 X/T 1 0 Op 55** avait été saisie).

Pour les calculs avec des nombres complexes ou des fractions, il est recommandé d'activer le mode d'affichage combiné à l'aide de **Op 1E**. Dans ce cas, le contenu du registre T est affiché sur la 1ère ligne de

l'afficheur et le contenu du registre X sur la 2ème ligne.

Dans le cas de calculs avec des nombres complexes, le registre T contient la partie réelle du nombre et le registre X contient la partie imaginaire du nombre. Après conversion en coordonnées polaires, le registre T contient le module (valeur numérique absolue, rayon) et le registre X contient la phase (argument, angle). Les calculs de nombres complexes sont toujours effectués en coordonnées cartésiennes. L'instruction **P->R** peut être utilisée pour convertir un nombre complexe entre coordonnées cartésiennes et polaires.

Dans le cas de calculs avec fractions (a/b), le registre T contient le numérateur 'a' et le registre X contient le dénominateur 'b'. Pour simplifier les fractions, l'opération **Op 50** peut être utilisée pour trouver le plus grand commun diviseur de la fraction.

### **Op 56 Affiche le nombre de nombres complexes dans la pile**

L'opération **Op 56** affiche (registre X) le nombre de nombres complexes ou de fractions présents dans la pile. Juste après l'initialisation avec **Op 55**, la valeur 0 est renvoyée.

### **Op 57 Insérer un nombre dans la pile de nombres complexes**

L'opération **Op 57** ajoute un nouveau nombre dans la pile de nombres complexes ou de fractions. En entrée, le registre T contient la partie réelle d'un nombre complexe (ou le numérateur d'une fraction), le registre X contient la partie imaginaire d'un nombre complexe (ou le dénominateur d'une fraction).

### **Op 58 Rappel un nombre de la pile de nombres complexes**

L'opération **Op 58** rappelle un nombre en haut de la pile de nombres complexes ou de fractions. Le numéro chargé n'est pas effacé de la pile, la pile reste inchangée. La partie réelle du nombre complexe (ou le numérateur de la fraction) est renvoyée dans le registre T, la partie imaginaire du nombre complexe (ou le dénominateur de la fraction) est renvoyée dans le registre X (affichage).

## Op 59 Annule un nombre dans pile de nombres complexes

L'opération **Op 59** supprime le dernier nombre du haut de la pile de nombres complexes ou de fractions. Le nombre total de paires de nombres dans la pile est diminué de 1.

## Op 5A Échange 2 nombres dans pile de nombres complexes

Avec l'opération **Op 5A**, dans la pile de nombres complexes ou de fractions, le nombre en haut de la pile (le dernier nombre) est échangé avec l'avant-dernier nombre.

## Op 5B Duplique un nombre dans pile de nombres complexes

Avec l'opération **Op 5B**, le dernier nombre en haut de la pile est dupliqué dans la pile de nombres complexes ou de fractions. Le nombre total de nombres dans la pile est incrémenté de 1.

## Op 5C Addition de deux nombres complexes X+Y

L'opération **Op 5C** ajoute le dernier nombre complexe Y en haut de la pile à l'avant-dernier nombre X. Annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat de X+Y et deviendra donc le dernier nombre en haut de la pile.

## Op 5D Différence de deux nombres complexes X-Y

L'opération **Op 5D** soustrait le dernier nombre complexe Y en haut de la pile de l'avant-dernier nombre X. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat X-Y et deviendra donc le dernier nombre en haut de la pile.

## Op 5E Produit de deux nombres complexes $X*Y$

L'opération **Op 5E** multiplie l'avant-dernier nombre complexe  $X$  par le dernier nombre  $Y$  en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X*Y$  et deviendra donc le dernier nombre en haut de la pile.

## Op 5F Quotient de deux nombres complexes $X/Y$

L'opération **Op 5F** divise l'avant-dernier nombre complexe  $X$  par le dernier nombre  $Y$  en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X/Y$  et deviendra donc le dernier nombre en haut de la pile.

## Op 60 Exponentiation de deux nombres complexes $X^Y$

L'opération **Op 60** élève l'avant-dernier nombre complexe  $X$  à la puissance  $Y$  (dernier nombre  $Y$  en haut de la pile). Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X^Y$  et deviendra donc le dernier nombre en haut de la pile.

## Op 61 Racine de deux nombres complexes $X^{(1/Y)}$

Opération **Op 61** extrait la racine de l'avant-dernier nombre complexe  $X$  par le dernier nombre  $Y$  en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X^{(1/Y)}$  et deviendra donc le dernier nombre en haut de la pile.

## Op 62 Logarithme de deux nombres complexes $\log Y(X)$

L'opération **Op 62** calcule le logarithme de l'avant-dernier nombre complexe  $X$  avec la base du dernier nombre  $Y$  en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $\log Y(X)$  et deviendra donc le dernier nombre en haut de la pile.

### Op 63 Le carré du nombre complexe $X^2$

L'opération **Op 63** calcule le carré du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $X^2$  à la position du nombre d'origine  $X$ .

### Op 64 La racine carrée du nombre complexe $\sqrt{X}$

L'opération **Op 64** calcule la racine carrée du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $\sqrt{X}$  à la position du nombre d'origine  $X$ .

### Op 65 L'inverse du nombre complexe $1/X$

L'opération **Op 65** calcule l'inverse du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $1/X$  à la position du nombre d'origine  $X$ .

### Op 66 L'exposant naturel du nombre complexe $e^X$

L'opération **Op 66** calcule l'exposant naturel du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $e^X$  à la position du nombre d'origine  $X$ .

### Op 67 Le logarithme naturel du nombre complexe $\ln(X)$

L'opération **Op 67** calcule le logarithme népérien du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $\ln(X)$  à la position du nombre d'origine  $X$ .

### Op 68 Sinus du nombre complexe $\sin(X)$

L'opération **Op 68** calcule le sinus du nombre complexe  $X$  en haut de la pile. Elle laisse le résultat  $\sin(X)$  à la position du nombre d'origine  $X$ .

### Op 69 Cosinus d'un nombre complexe $\cos(X)$

L'opération **Op 69** calcule le cosinus du nombre complexe  $X$  en haut de la

pile. Elle laisse le résultat  $\cos(X)$  à la position du nombre d'origine X.

### Op 6A Tangente du nombre complexe $\tan(X)$

L'opération **Op 6A** calcule la tangente du nombre complexe X en haut de la pile. Elle laisse le résultat  $\tan(X)$  à la position du nombre d'origine X.

### Op 6B Arcsinus du nombre complexe $\text{asin}(X)$

L'opération **Op 6B** calcule l'arc sinus du nombre complexe X en haut de la pile. Elle laisse le résultat  $\text{asin}(X)$  à la position du nombre d'origine X.

### Op 6C Arccosinus du nombre complexe $\text{acos}(X)$

L'opération **Op 6C** calcule l'arc cosinus du nombre complexe X en haut de la pile. Elle laisse le résultat  $\text{acos}(X)$  à la position du nombre d'origine X.

### Op 6D Arctangente du nombre complexe $\text{atan}(X)$

L'opération **Op 6D** calcule l'arctangente du nombre complexe X en haut de la pile. Elle laisse le résultat  $\text{atan}(X)$  à la position du nombre d'origine X.

### Op 6E Convertir un nombre complexe en un nombre polaire

L'opération **Op 6E** convertit le nombre complexe X en haut de la pile de coordonnées cartésiennes en coordonnées polaires. Elle laisse le résultat à la position du nombre d'origine X. Les opérations arithmétiques ne peuvent pas être effectuées sur la pile avec un nombre sous forme polaire.

### Op 6F Convertir nombres polaires en nombres complexes

L'opération **Op 6F** convertit le nombre X en haut de la pile de coordonnées polaires en coordonnées cartésiennes. Elle laisse le résultat à la position du nombre d'origine X. Les opérations arithmétiques ne



peuvent pas être effectuées sur la pile avec un nombre sous forme polaire.

## Op 70 Trouver le passage par zéro de A'

L'opération **Op 70** recherche numériquement les passages par zéro de la fonction utilisateur **A'**. Avant utilisation, une fonction est créée dans le programme principal, étiquetée **Lbl A'**, qui calcule la valeur de sortie y pour la valeur d'entrée x. La fonction **A'** ne doit pas utiliser le signe égal **=** ou la fonction **CLR**.

A l'entrée de l'opération **Op 70**, la valeur finale x est stockée dans le registre T, à laquelle s'arrête la recherche du zéro. Le registre X sera la valeur d'entrée par défaut. La fonction **Op 70** divise l'intervalle entre T et X en 100 segments. Dans chaque section, il teste (en appelant à plusieurs reprises la fonction utilisateur **A'**) si la fonction passe par zéro, c'est-à-dire si le signe change entre le début et la fin du segment. S'il trouve une section avec un passage à zéro, il recherche l'endroit exact du passage à zéro, en utilisant la méthode consistant à diviser par deux les intervalles jusqu'à ce que l'écart d'imprécision soit en dehors de la zone affichable.

Si la fonction trouve un passage par zéro, elle renvoie la valeur trouvée de x dans le registre X. La valeur trouvée de x peut devenir la nouvelle valeur de départ pour une nouvelle recherche car la fonction ne recherche pas de passage par zéro dans la valeur de départ. Un nouvel appel à la fonction **Op 70** poursuivra la recherche du prochain passage à zéro à partir de la dernière valeur x trouvée (qui doit être conservée dans le registre X à cet effet).

Lors de la recherche, il est nécessaire de considérer attentivement le début et la fin de l'intervalle recherché. Lorsque l'intervalle est trop grand, la segmentation peut être trop grossière et le passage par zéro peut être manqué. Au contraire, si la section est trop courte, le passage par zéro recherché peut se situer en dehors de la zone testée.

## Op 71 Intégrale de Simpson de la fonction A'

L'opération **Op 71** calcule numériquement l'intégrale de la fonction utilisateur **A'**, par la méthode de Simpson. Avant utilisation, une fonction est créée dans le programme principal, étiquetée **Lbl A'**, qui calcule la

valeur de sortie y pour la valeur d'entrée x. La fonction **A'** ne doit pas utiliser le signe égal **=** ou la fonction **CLR**.

Avant d'appeler la fonction, la limite inférieure de calcul de l'intégrale  $x_0$  est stockée dans le registre HIR H1, la limite supérieure  $x_n$  est stockée dans H2 et le nombre d'étapes n (un nombre pair) est stocké dans H3. La fonction renvoie la valeur de l'intégrale dans le registre X (à l'écran).

## **Op 72 Convertir un angle de l'unité d'angle en cours en radians**

L'opération **Op 72** convertit l'angle de l'unité d'angle actuelle (sélectionnée par l'utilisateur) en radians. Cela permet d'effectuer des calculs trigonométriques avec des angles en radians sans avoir à modifier les paramètres utilisateur de l'unité d'angle.

## **Op 73 Convertir un angle en radians à l'unité d'angle en cours**

L'opération **Op 73** convertit l'angle des radians en l'unité d'angle actuelle (sélectionnée par l'utilisateur). Cela permet d'effectuer des calculs trigonométriques avec des angles en radians sans avoir à modifier les paramètres utilisateur de l'unité d'angle.

## **Op 74 Distribution de probabilité normale Z(x)**

L'opération **Op 74** calcule la distribution de probabilité normale Z(x) pour la valeur « x » comprise entre -150 et +150.

## **Op 75 Distribution gaussienne complémentaire Q(x)**

L'opération **Op 75** calcule la distribution gaussienne complémentaire Q(x) pour la valeur de « x » comprise entre -8 et +8.

## **Op 76 Distribution normale cumulative de P(x)**

L'opération **Op 76** calcule la distribution normale cumulative P(x) pour la

valeur de « x » comprise entre -8 et +8.

### Op 77 Maximum

L'opération **Op 77** compare les registres X et T, et renvoie la plus grande des valeurs dans le registre X.

### Op 78 Minimum

L'opération **Op 78** compare les registres X et T, et renvoie la plus petite des valeurs dans le registre X.

### Op 79 Réinitialiser tout les registres HIR

L'opération **Op 79** réinitialise le contenu de tous les registres HIR de H0 à H15.

### Op 7A Conversion décimale en fraction

L'opération **Op 7A** convertit le nombre décimal « x » en fraction  $a/b$ , renvoyant le numérateur « a » dans le registre T et le dénominateur « b » de la fraction dans le registre T.

### Op 7B Conversion d'une fraction en un nombre décimal

L'opération **Op 7B** divise le numérateur de la fraction « a » présent dans le registre T par le dénominateur « b » présent dans le registre X et renvoie le résultat dans le registre X sous forme de nombre décimal.

### Op 7C La somme de la fraction X+Y

L'opération **Op 7C** ajoute le dernier nombre fractionnaire Y en haut de la pile à l'avant-dernier nombre X. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat X+Y et deviendra donc

le dernier nombre en haut de la pile.

### Op 7D Différence de fraction X-Y

L'opération **Op 7D** soustrait le dernier nombre fractionnaire Y en haut de la pile de l'avant-dernier nombre X.. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat X-Y et deviendra donc le dernier nombre en haut de la pile.

### Op 7E Produit de deux fractions de X\*Y

L'opération **Op 7E** multiplie l'avant-dernier nombre fractionnaire X par le dernier nombre Y en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat X\*Y et deviendra donc le dernier nombre en haut de la pile.

### Op 7F Quotient de deux fractions X/Y

L'opération **Op 7F** divise l'avant-dernier nombre fractionnaire X par le dernier nombre Y en haut de la pile. Elle annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat X/Y et deviendra donc le dernier nombre en haut de la pile.

### Op 80 Temps de pause de 10 ms

L'opération **Op 80** attendra dans le programme pendant 10 ms (0,01 seconde). En raison du retard potentiel dans l'exécution du programme, le temps résultant peut être un peu plus long que le temps prévu (environ 10 à 20 %).

### Op 81 Temps de pause de 100 ms

L'opération **Op 81** attendra dans le programme pendant 100 ms (0,1 seconde). En raison du retard potentiel dans l'exécution du programme, le temps résultant peut être un peu plus long que le temps prévu (environ 10 à 20 %).

## Op 82 Suppression des chiffres masqués

L'opération **Op 82** supprime les chiffres cachés d'un nombre affiché sur l'écran (registre X). C'est une fonction similaire à la séquence **EE INV EE**, mais l'avantage est que le mode de l'exposant n'est pas modifié. Cette procédure permet de supprimer facilement les chiffres cachés et d'arrondir le nombre. Une autre alternative est la fonction **INV .**, qui laisse le nombre sous une forme modifiable.

## Op 83 Début du chronométrage

La calculatrice utilise un temporisateur interne de 16 bits avec une résolution de 10 ms (0,01 seconde) et une durée de période de 10,9 minutes. L'opération **Op 83** enregistre l'état actuel de la minuterie sous forme d'horodatage pour les mesures d'intervalle de temps ultérieures.

## Op 84 Détermination du temps écoulé

L'opération **Op 84** soustrait de la valeur actuelle du compteur de temps l'horodatage enregistré par l'opération **Op 83**. Elle renvoie la différence dans le registre X (affichage) sous forme de temps écoulé en secondes. La résolution des données obtenues est de 10 ms (0,01 seconde) et la période maximale mesurable est de 10,9 minutes. Passé ce délai, le décalage horaire déborde et se remet à zéro.

La méthode de mesure du temps utilisant un compteur de temps interne est plus précise que la fonction de retard (**Op 44**, **Op 80**, **Op 81**), car elle ne dépend pas de la charge du processeur pendant l'exécution du programme.

## Op 85 Détermination du nombre de registres de données

L'opération **Op 85** renvoie le nombre de registres de données disponibles dans la calculatrice. Cela sert à garantir la portabilité du programme entre différentes variantes de la calculatrice. Alternativement, il peut également être utilisé pour détecter une variante du calculateur.

## Op 86 Détermination de l'état des drapeaux utilisateur

L'opération **Op 86** renvoie un entier de 16 bits (allant de 0 à 65 535) qui représente l'état des 16 drapeaux utilisateur. La fonction est utilisée pour un accès rapide aux drapeaux.

## Op 87 Calcul de la somme de contrôle de la mémoire ROM

L'opération **Op 87** calcule la somme de contrôle de la mémoire ROM de la calculatrice (en utilisant la méthode CRC-XModem) et la renvoie dans le registre X sous la forme d'un nombre entier de 16 bits (plage de 0 à 65535). En même temps, il renvoie la valeur attendue de la somme de contrôle stockée dans la ROM dans le registre T. Si les données ne correspondent pas, la calculatrice est endommagée.

## Op 88 Réglage du délai d'extinction de la calculatrice

L'opération **Op 88** peut être utilisée pour définir la période d'inactivité après laquelle la calculatrice s'éteint. La saisie de l'opération est le temps en secondes dans le registre X (affichage), minimum 5 secondes et maximum 650 secondes (soit près de 11 minutes). La saisie d'une valeur de 0 désactive l'arrêt automatique de la calculatrice.

## Op 89 Détermination du délai d'extinction de la calculatrice

L'opération **Op 89** peut être utilisée pour déterminer le temps en secondes d'arrêt automatique de la calculatrice pendant l'inactivité. 0 signifie arrêt automatique désactivé.

## Op 8A Affichage de la version du firmware de la calculatrice

L'opération **Op 8A** affiche la version du firmware de la calculatrice sur l'écran, comme après la réinitialisation. Le nom de la variante de la calculatrice apparaîtra à l'écran pendant 2 secondes, accompagné d'un code à 6 chiffres représentant la date de la version du firmware de la calculatrice. Par exemple "ET-58 200908" désigne la variante de calculatrice "Basique" et la date du firmware (build) est le 09/08/2020.

## Op 8B Réinitialiser la calculatrice

L'opération **Op** **8B** réinitialise la calculatrice de la même manière que le retrait de la batterie.

## 16. Table des caractères

La table de caractères utilisée pour l'impression sur un fichier ou sur l'écran utilise un codage de caractères basé sur le code ASCII (réduit du décalage 32) et diffère de la table de caractères TI-58/59 d'origine. Les caractères sont saisis sous forme de 2 chiffres d'un nombre décimal compris entre 00 et 99. Un registre de données peut contenir jusqu'à 8 caractères, soit 16 chiffres. A cet effet, l'éditeur de nombres vous permet de saisir jusqu'à 16 chiffres décimaux, mais n'affiche que le nombre maximum. 14 chiffres.

00 spc	16 0	32 @	48 P	64 '	80 p	96 racine
01 !	17 1	33 A	49 Q	65 a	81 q	97 micro
02 "	18 2	34 B	50 R	66 b	82 r	98 oméga
03 #	19 3	35 C	51 S	67 c	83 s	99 rectangle
04 \$	20 4	36 D	52 T	68 d	84 t	
05 %	21 5	37 E	53 U	69 e	85 u	
06 &	22 6	38 F	54 V	70 f	86 v	
07 '	23 7	39 G	55 W	71 g	87 w	
08 (	24 8	40 H	56 X	72 h	88 x	
09 )	25 9	41 I	57 Y	73 i	89 y	
10 *	26 :	42 J	58 Z	74 j	90 z	
11 +	27 ;	43 K	59 [	75 k	91 {	
12 ,	28 <	44 L	60 \	76 l	92	
13 -	29 =	45 M	61 ]	77 m	93 }	
14 .	30 >	46 N	62 ^	78 n	94 ~	
15 /	31 ?	47 O	63 _	79 o	95 pi	

Les huit derniers caractères de code 92 à 99 peuvent être régénérés en chargeant les polices à l'aide de l'opération **Op** **43**. En cas de régénération de la police, l'antislash 60 \ est remplacé par un trait vertical |.



Exemple, texte "Calculate filter"

C a l c u l a t e \_ f i l t e r

35 65 76 67 85 76 65 84 **Op** **01** 69 00 70 73 76 84 69 82 **Op** **02**

## 17. Programmes de la bibliothèque

La calculatrice ET-58 est équipée d'une bibliothèque avec de riches programmes de bibliothèque. Les programmes de bibliothèque sont activés par l'instruction **Pgm** suivie du programme de bibliothèque numéro 1 à 50 (ou plus pour la bibliothèque utilisateur). Le chiffre 0 représente le programme utilisateur principal. Les programmes de la bibliothèque peuvent être visualisés à l'aide de la touche **LRN** ou transférés vers la mémoire principale avec l'instruction **Op 09**.

La plupart des programmes de bibliothèque utilisent les registres HIR (H0 à H15) comme registres de travail. Si la bibliothèque nécessite des données utilisateur, elle utilise préférentiellement les registres de données du registre R10 et supérieur, les registres R00 à R09 sont réservés par ex. pour la fonction statistique **Stat**. Ce n'est que dans certains cas qu'il utilise tous les registres de données.

Pour les fonctions qui impriment une invite sur la 1ère ligne de l'écran, le texte de l'invite peut être effacé en appuyant sur **CLR**. La fonction **Pgm 01 SBR CE** peut être utilisée pour réinitialiser les paramètres de la calculatrice à l'état par défaut.

### ML-01 Diagnostic

Version		Flags	Fill	Reset
Polices	Touches	Stat	Clear	Diag

Le programme de bibliothèque ML-01 est utilisé à des fins de diagnostic et d'assistance.

#### **Lbl A** Polices

Consultation des polices. L'écran affiche une page de 32 caractères sur l'écran LCD. Les boutons SST et BST peuvent être utilisés pour faire défiler les pages de la police. La police contient 4 pages avec les caractères 0 à 99 (la dernière page est incomplète). Les boutons 0 à 4 changent la police utilisée (voir Op 43 Chargement de police, avec des exemples de polices) :

0 par défaut, 1 colonne de gauche, 2 colonnes de droite, 3 lignes et graphiques, 4 pixels. La fonction se termine avec la touche **R/S**.

### **Lbl A** Version

Affichage de la version de la calculatrice. Le nom de la version de la calculatrice s'affiche pendant 2 secondes ainsi qu'un code à 6 chiffres représentant la date de la version du firmware de la calculatrice. Par ex. « ET-58 201005 » signifie date de firmware (build) 5/10/2020.

### **Lbl B** Touches

Affiche les code des touches. La fonction affiche les codes des boutons enfoncés en code HEX. Les codes prennent en compte l'utilisation éventuelle des préfixes **2nd** ou **3rd**. Tous les boutons sauf les boutons **2nd** et **R/S** peuvent être testés. La fonction se termine avec la touche **R/S**. La calculatrice est alors en mode HEX, il faudra utiliser **DEC** pour repasser en mode décimal.

### **Lbl C**, **Lbl CLR** Réinitialisation des statistiques

Réinitialiser les registres statistiques R01 à R06 (pour la fonction **Stat**), réinitialiser les registres X et T.

### **Lbl C** Flags / Drapeaux

Affichage de l'état de tous les drapeaux utilisateur 0 à 15 en partant de la droite. La 1ère ligne de l'écran affiche l'état de tous les commutateurs sous forme de groupe de 16 chiffres 0 et 1. L'affichage peut être terminé en appuyant sur **CLR**.

### **Lbl D**, **Lbl CE** Réinitialisation valeurs

La fonction réinitialise la calculatrice aux paramètres par défaut, réinitialise les registres statistiques R01 à R06, réinitialise les registres X et T.

### **Lbl D** Remplir les registres

Remplissage de tous les registres de données avec le numéro affiché à l'écran.

### **Lbl** **E**, **Lbl** **=** Diagnostic

La fonction teste la fonctionnalité de la calculatrice et vérifie également la somme de contrôle CRC de la mémoire ROM avec le micrologiciel de la calculatrice. En fonction du résultat du test, l'écran affichera le texte "Diagnose OK" ou "Diagnose ERROR".

### **Lbl** **E'** Réinitialisation calculatrice

La fonction réinitialise la calculatrice de la même manière que si vous retiriez la batterie.

## **ML-02 Matrices: Inversion, déterminant, équations**

i->x	->1/A	j->1/a		-> A .1/A
n	j:a	-> A	i:b	->x

Le programme ML-02 permet de calculer le déterminant d'une matrice et son inverse. Il prend en charge les matrices carrées 2x2 à 9x9.

### **Lbl** **A** Dimension de la matrice

Saisir de la dimension 'n' de la matrice carrée (n = 1 à 9).

### **Lbl** **B** Saisie des données de la matrice

Saisie de l'index de la colonne de départ de la matrice 'j' et saisie des données de la matrice. Les données sont saisies à partir de la colonne spécifiée par ligne de haut en bas. Après avoir écrit chaque élément, appuyez sur **R/S** pour continuer. En cas d'erreur, vous pouvez ressaisir le numéro de colonne, appuyer sur **B** et continuer la saisie à partir de la

première ligne de la colonne saisie.

### **Lbl C** Déterminant

Calcule le déterminant de la matrice.

### **Lbl D** Saisir un vecteur

Spécifier l'index de l'entrée initiale 'i' du vecteur et saisir les données du vecteur. Après avoir écrit chaque élément, appuyez sur **R/S** pour continuer. En cas d'erreur, le numéro de l'élément de départ peut être ressaisi, appuyez sur **D** et continuez la saisie à partir de l'élément saisi.

### **Lbl E** Résoudre des équations

Résoud le système d'équations linéaires spécifié par le vecteur et la matrice.

### **Lbl A'** Lecture du vecteur

Spécifier l'index de l'entrée initiale 'i' du vecteur et lire les entrées. Passez à l'élément suivant en appuyant sur **R/S**. La lecture peut être répétée en entrant un nouvel index d'élément et en appuyant sur **A'**.

### **Lbl B'** Inversion de matrice

D'abord, le déterminant de la matrice doit être calculé avec **C**. Ce déterminant ne doit pas être 0.

### **Lbl C'** Lecture de la matrice inverse

Spécifier l'index de la colonne initiale de la matrice 'j' et lire les données de la matrice inverse (après le calcul précédent avec **B'**). Passez à l'élément suivant en appuyant sur **R/S**. Les données sont lues de la colonne spécifiée de haut en bas.

## Lbl E' Déterminant et inversion

Calcul du déterminant de la matrice et inversion de la matrice. Cette fonction inclut les appels aux fonctions C (déterminant de la matrice) et B (inversion de la matrice).

Exemple:

Création d'une matrice carrée 3x3 avec son contenu

$$A = \begin{pmatrix} 4 & 8 & 0 \\ 8 & 8 & 8 \\ 2 & 0 & 1 \end{pmatrix}$$

Pgm 02 ... activation du programme de bibliothèque ML-02

3 A ... saisie de la dimension de la matrice 3x3

1 B ... la saisie des données commence à la 1ère colonne

4 R/S 8 R/S 2 R/S ... saisie des données de la 1ère colonne

8 R/S 8 R/S 0 R/S ... saisie des données de la 2ème colonne

0 R/S 8 R/S 1 R/S ... saisie des données de la 3ème colonne

C [96] ... calcul du déterminant, résultat = 96

recherche vecteur x dont la multiplication par  $A \cdot x$  donne un vecteur b :

$$b = \begin{pmatrix} 4 \\ 4 \\ 6 \end{pmatrix}$$

1 D ... saisie des données du vecteur pour l'élément 1

4 R/S 4 R/S 6 R/S ... saisie des données vectorielles b

E ... résolution du système d'équations linéaires

1 A' ... lecture des données du vecteur x à partir de l'élément 1

**R/S** [4] **R/S** [-1.5] **R/S** [-2] ... résolution du système d'équations x

La solution du système d'équations donné est

$$x = \begin{matrix} 4 \\ -1.5 \\ -2 \end{matrix}$$

Nous voulons maintenant calculer la matrice inverse  $A^{-1}$ . Le déterminant a déjà été calculé 96.

**B'** ... calcul de la matrice inverse

**1 C'** ... lecture des données de la matrice inverse à partir de la colonne 1

**R/S** [.0833...] **R/S** [.0833...] **R/S** [-.1666...] ... lecture colonne 1

**R/S** [-.0833...] **R/S** [.0416...] **R/S** [.1666...] ... lecture colonne 2

**R/S** [.6666...] **R/S** [-.3333...] **R/S** [-.3333...] ... lecture colonne 2

La matrice inverse est :

$$A^{-1} = \begin{matrix} & 0.0833 & -0.0833 & 0.6667 \\ 0.0833 & 0.04167 & -0.3333 & \\ -0.1667 & 0.1667 & -0.3333 & \end{matrix}$$

Sinon, nous pouvons la convertir en fractions en utilisant ML-26:

$$A^{-1} = \begin{matrix} & 1/12 & -1/12 & 2/3 \\ 1/12 & 1/24 & -1/3 & \\ -1/6 & 1/6 & -1/3 & \end{matrix}$$

## ML-03 Addition et multiplication de matrices

j:c	i:xi	->Ax	i->y	
m,n	j:a	j:b	la1,la2	A+B

Le programme ML-03 permet l'addition de matrices (avec multiplication par une valeur scalaire) et la multiplication de matrices. Lors de l'ajout, les deux matrices sont entrées dans leur intégralité en mémoire. Lors de la multiplication, seule la colonne de la deuxième matrice est saisie et également une seule colonne est lue, l'opération est répétée séquentiellement pour toutes les colonnes.

### Lbl A Dimensions de la matrice

Saisie des dimensions de la matrice : saisir d'abord le nombre de lignes 'm' et appuyer sur **A**. Saisir ensuite le nombre de colonnes de la matrice 'n' et appuyer à nouveau sur **A**.

### Lbl B Saisie des données de la 1ère matrice

Saisir l'index de la colonne de la 1ère matrice à partir de laquelle les données ( $j = 1 \dots n$ ) doivent être saisies et appuyez sur **B**. Saisir ensuite les données de la première matrice (A) par colonnes dans l'ordre, de haut en bas. Appuyer sur **R/S** pour valider chaque valeur.

### Lbl C Saisie des données de la 2ème matrice

Saisir l'index de la colonne de la 2ème matrice à partir de laquelle les données ( $j = 1 \dots n$ ) doivent être saisies et appuyez sur **C**. Saisir ensuite les données de la deuxième matrice (B) par colonnes dans l'ordre, de haut en bas. Appuyer sur **R/S** pour valider chaque valeur.

### Lbl D Saisie d'un multiple scalaire

Saisir le multiple scalaire lambda1 de la première matrice A et appuyer sur **D**. Saisir ensuite le multiple scalaire lambda2 de la deuxième matrice B et



appuyer à nouveau sur **D**.

### **Lbl E** Somme des matrices

Effectuer la somme des matrices  $C = \text{lambda1} * A + \text{lambda2} * B$

### **Lbl A'** Lecture de la somme des matrices

Entrer l'index de colonne de la matrice résultante à partir de laquelle lire les données ( $j = 1 \dots n$ ) et appuyer sur **A'**. Lire les données par éléments de colonne dans l'ordre de haut en bas. Appuyez sur **R/S** pour continuer.

### **Lbl B'** Choix de la colonne de la matrice à multiplier

Saisir le numéro d'article de la colonne de la 2ème matrice de multiplication, à partir de laquelle seront saisies les données ( $i = 1 \dots m$ ). Entrer les données de la colonne de la deuxième matrice, continuer avec **R/S**. La multiplication se fait colonne par colonne. Tout d'abord, choisir la colonne de la 2ème matrice pour la multiplication, effectuer la multiplication, lire la colonne résultante, puis répéter l'opération séquentiellement pour toutes les colonnes de la deuxième matrice.

### **Lbl C'** Multiplier une colonne

La fonction **C'** multiplie une colonne de la deuxième matrice par la matrice A. Après lecture du résultat, continuer en saisissant la colonne suivante de la deuxième matrice.

### **Lbl D'** Lire la colonne de résultat de la multiplication

Saisir le numéro de l'entrée de colonne de la matrice résultante à partir de laquelle seront lues les données ( $i = 1 \dots n$ ). Lire les données et enchaîner avec **R/S**. Après lecture de la colonne, la colonne suivante de la deuxième matrice peut être saisie pour continuer en multipliant la colonne suivante.

## Exemple

Somme des matrices  $C = A - 2*B$

$$A = \begin{pmatrix} 2 & 3 & 0 \\ 1 & 0 & 5 \end{pmatrix}$$
$$B = \begin{pmatrix} 4 & 0 & -1 \\ 3 & 2 & 6 \end{pmatrix}$$

**Pgm 03** ... activation du programme de bibliothèque ML-03

**2 A 3 A** ... dimension de la matrice : 2 lignes (m) et 3 colonnes (n)

**1 B** ... saisir les données de la première matrice à partir de la colonne 1

**2 R/S 1 R/S** ... saisir les données de la 1ère colonne

**3 R/S 0 R/S** ... saisir les données de la 2ème colonne

**0 R/S 5 R/S** ... saisir les données de la 3ème colonne

**1 C** ... saisir les données de la deuxième matrice à partir de la colonne 1

**4 R/S 3 R/S** ... saisir les données de la 1ère colonne

**0 R/S 2 R/S** ... saisir les données de la 2ème colonne

**1 +/- R/S 6 R/S** ... saisir les données de la 3ème colonne

**1 D 2 +/- D** ... saisir lambda1 (1) et lambda2 (-2)

**E** ... calcul de la somme des matrices  $C = A - 2*B$

**1 A'** ... zahájení čtení výsledku součtu počínaje sloupcem 1

**[-6] R/S [-5]** ... lire les données de la 1ère colonne

**R/S [3] R/S [-4]** ... lire les données de la 2ème colonne

**R/S [2] R/S [-7]** ... lire les données de la 3ème colonne

Matrice résultante de la somme :

$$C = \begin{pmatrix} -6 & 3 & 2 \\ -5 & -4 & -7 \end{pmatrix}$$

Multiplication de la matrice résultante C par la matrice D ( $E = C \cdot D$ )

$$D = \begin{array}{cc} 3 & 1 \\ 0 & 2 \\ 4 & 3 \end{array}$$

**1** **B'** ... la colonne de la 2ème matrice sera renseignée à partir du 1er élément

**3** **R/S** **0** **R/S** **4** **R/S** ... saisie des données 1ère colonne 2ème matrice

**C'** ... multiplier la 1ère matrice par une colonne de la 2ème matrice

**1** **D'** ... la colonne résultat sera lue à partir du 1er élément

**[-10]** **R/S** **[-43]** ... lecture de la 1ère colonne du résultat

**1** **B'** ... la colonne de la 2ème matrice sera renseignée à partir du 1er élément

**1** **R/S** **2** **R/S** **3** **R/S** ... saisir les données de la 2ème colonne de la 2ème matrice

**C'** ... multiplier la 1ère matrice par une colonne de la 2ème matrice

**1** **D'** ... la colonne résultat sera lue à partir du 1er élément

**[6]** **R/S** **[-34]** ... lecture de la 2ème colonne du résultat

Matrice résultante de la multiplication :

$$E = \begin{array}{cc} -10 & 6 \\ -43 & -34 \end{array}$$

## ML-04 Arithmétique complexe

INIT	X-Y	X:Y	logyX	X<>Y
ENTER	X+Y	XxY	X^Y	X^1/Y

Le programme ML-04 est utilisé pour les calculs arithmétiques de base avec des nombres complexes. Les nombres complexes contiennent la partie réelle du nombre dans le registre T et la partie imaginaire du nombre dans le registre X (contenu d'affichage). Pour les opérations sur les nombres complexes, une pile de 10 nombres complexes est utilisée, située dans les registres de données R10 à R29. Lors des opérations, les opérandes nécessaires sont d'abord stockés dans la pile puis l'opération correspondante est effectuée.

### **Lbl** **A** Ajouter un nombre à la pile

Entrer la partie réelle du nombre, appuyer sur **x<>t**, entrer la partie imaginaire du nombre et appuyer sur **A**, le nombre saisi est ajouté en haut de la pile.

### **Lbl** **A'** Initialisation de la pile

L'opération **A'** initialise la pile de nombres dans les registres R10 à R29. L'initialisation peut être réutilisée pour un nouveau calcul. Lors de l'initialisation, l'écran passe en mode deux lignes. La ligne supérieure affiche la partie réelle du nombre complexe (registre T) et la ligne inférieure affiche la partie imaginaire du nombre complexe (registre X). Le mode d'affichage sur deux lignes peut être terminé par l'opération **Op** **1D** ou en appelant le sous-programme **Pgm** **01** **SBR** **CE**.

### **Lbl** **B** Somme des nombres X+Y

L'opération **B** ajoute le nombre Y en haut de la pile à l'avant-dernier nombre X. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne

inférieure (registre X).

### **Lbl B'** Différence des nombres $X-Y$

L'opération **B'** soustrait le nombre Y en haut de la pile de l'avant-dernier nombre X. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C** Multiplication des nombres $X*Y$

L'opération **C** multiplie le nombre Y en haut de la pile par l'avant-dernier nombre X. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C'** Division des nombres $X:Y$

L'opération **C'** divise l'avant-dernier nombre X par le nombre Y du haut de la pile. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl D** élévation à la puissance du nombre $X^Y$

L'opération **D** élève l'avant-dernier nombre X à la puissance du nombre Y en haut de la pile. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl** **D'** Logarithme $\log Y(X)$

L'opération **D'** calcule le logarithme de l'avant-dernier nombre X par la base du nombre Y en haut de la pile. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl** **E** Racine de $X^{(1/Y)}$

L'opération **E** calcule la racine nième de l'avant-dernier nombre X avec le nombre Y en haut de la pile. Le nombre Y en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl** **E'** Permuter les opérands

L'instruction **E'** permute les nombres Y (en haut de la pile) et le nombre X (avant-dernier de la pile). Le nouveau nombre Y est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

Exemple, calcul de  $((2 + 3i) * (1 - 1i))^{(1 + 1i)}$ :

**Pgm** **04** ... activation du programme de bibliothèque ML-04

**A'** ... initialisation de la pile de nombres complexes

**2** **x<>t** **3** **A** ... mettre le nombre  $(2 + 3i)$  sur la pile

**1** **x<>t** **1** **+/-** **A** ... mettre le nombre  $(1 - 1i)$  sur la pile

**C** ... multiplication des nombres  $X * Y$

**1** **x<>t** **1** **A** ... mettre le nombre  $(1 + 1i)$  sur la pile

**D** ... élévation à la puissance  $X ^ Y$ , résultat :  $(-1.0584... + 4.0495...i)$

## ML-05 Fonctions complexes

INIT	$e^X$	$x^2$	P->R	DEL
ENTER	$\ln X$	Sqrt	R->P	1/X

Le programme ML-05 est utilisé pour calculer des fonctions avec des nombres complexes. Les nombres complexes contiennent la partie réelle du nombre dans le registre T et la partie imaginaire du nombre dans le registre X (contenu d'affichage). Pour les opérations sur les nombres complexes, une pile de 10 nombres complexes est utilisée, située dans les registres de données R10 à R29. Lors des opérations, les opérandes nécessaires sont d'abord stockés dans la pile puis l'opération correspondante est effectuée. Les angles sont en radians.

**Lbl A** Ajouter un nombre à la pile

Entrer la partie réelle du nombre, appuyer sur **x<>t**, entrer la partie imaginaire du nombre et appuyer sur **A**, le nombre saisi est ajouté en haut de la pile.

**Lbl A'** Initialisation de la pile

L'opération **A** initialise la pile de nombres dans les registres R10 à R29. L'initialisation peut être réutilisée pour un nouveau calcul. Lors de l'initialisation, l'écran passe en mode deux lignes. La ligne supérieure affiche la partie réelle du nombre complexe (registre T) et la ligne inférieure affiche la partie imaginaire du nombre complexe (registre X). Le mode d'affichage sur deux lignes peut être terminé par l'opération **Op 1D** ou en appelant le sous-programme **Pgm 01 SBR CE**.

**Lbl B** Logarithme népérien de  $\ln X$

L'opération **B** calcule le logarithme népérien du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne

inférieure (registre X).

### **Lbl B'** Exposant naturel $e^X$

L'opération **B'** calcule l'exposant naturel du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C** Racine carrée de X

L'opération **C** calcule la racine carrée du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C'** Elévation au carré $X^2$

L'opération **C'** élève au carré du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl D** Conversion en coordonnées polaires $R \rightarrow P$

L'opération **D** convertit le nombre X en haut de la pile de coordonnées cartésiennes en coordonnées polaires. Le résultat de l'opération est alors affiché sur l'écran avec le module du nombre (valeur absolue) sur la ligne supérieure (registre T) et l'argument du nombre en radians (phase, angle) sur la ligne inférieure (registre X). Les autres opérations avec un nombre en coordonnées polaires ne sont pas prises en charge.



## **Lbl** **D'** Conversion en coordonnées cartésiennes P -> R

L'opération **D'** convertit le nombre X en haut de la pile de coordonnées polaires (en radians) en coordonnées cartésiennes. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X). Les opérations avec un nombre en coordonnées polaires ne sont pas prises en charge.

## **Lbl** **E** Inverse 1/X

L'opération **E** calcule l'inverse du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl** **E'** Supprimer un nombre

L'opération **E'** supprime le nombre X en haut de la pile.

Exemple, calcul de la racine carrée de  $(2 + 3i)$  :

**Pgm** **05** ... activation du programme de bibliothèque ML-05

**A'** ... initialisation de la pile de nombres complexes

**2** **x<>t** **3** **A** ... mettre le nombre  $(2 + 3i)$  sur la pile

**C** ... calcul de racine carrée (Sqrt)

Résultat du calcul  $(1.6714... + 0.89597...i)$

**C'** ... calcul inverse pour contrôle ( $x^2$ ), résultat :  $(2 + 3i)$

## ML-06 Fonctions trigonométriques complexes

INIT	asin X	acos X	atan X	ABS
ENTER	sin X	cos X	tan X	NEG

Le programme ML-06 est utilisé pour calculer des fonctions trigonométriques avec des nombres complexes. Les nombres complexes contiennent la partie réelle du nombre dans le registre T et la partie imaginaire du nombre dans le registre X (contenu d'affichage). Pour les opérations sur les nombres complexes, une pile de 10 nombres complexes est utilisée, située dans les registres de données R10 à R29. Lors des opérations, les opérandes nécessaires sont d'abord stockés dans la pile puis l'opération correspondante est effectuée. Les angles sont en radians.

**Lbl** **A** Ajouter un nombre à la pile

Entrer la partie réelle du nombre, appuyer sur **x<>t**, entrer la partie imaginaire du nombre et appuyer sur **A**, le nombre saisi est ajouté en haut de la pile.

**Lbl** **A'** Initialisation de la pile

L'opération **A'** initialise la pile de nombres dans les registres R10 à R29. L'initialisation peut être réutilisée pour un nouveau calcul. Lors de l'initialisation, l'écran passe en mode deux lignes. La ligne supérieure affiche la partie réelle du nombre complexe (registre T) et la ligne inférieure affiche la partie imaginaire du nombre complexe (registre X). Le mode d'affichage sur deux lignes peut être terminé par l'opération **Op** **1D** ou en appelant le sous-programme **Pgm** **01** **SBR** **CE**.

**Lbl** **B** Sinus de X

L'opération **B** calcule le sinus du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre

X).

### **Lbl B'** Arcsinus de X

L'opération **B'** calcule l'arcsinus du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C** Cosinus de X

L'opération **C** calcule le cosinus du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl C'** Arccosinus de X

L'opération **C'** calcule l'arccosinus du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

### **Lbl D** Tangente de X

L'opération **D** calcule la tangente du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl D'** Arctangente de X

L'opération **D'** calcule l'arctangente du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl E** Négation de X (+/-)

L'opération **E** calcule la négation du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

## **Lbl E'** Valeur absolue de X

L'opération **E'** calcule la valeur absolue du nombre X en haut de la pile. Le nombre X en haut de la pile est remplacé par le résultat de l'opération qui devient le nouveau nombre en haut de la pile. Le résultat de l'opération est alors affiché sur l'écran avec la partie réelle du nombre sur la ligne supérieure (registre T) et la partie imaginaire du nombre sur la ligne inférieure (registre X).

Exemple, calcul du sinus de  $(1 + 3i)$  :

**Pgm 06** ... activation du programme de bibliothèque ML-06

**A'** ... initialisation de la pile de nombres complexes

**1** **x<>t** **3** **A** ... mettre le nombre  $(1 + 3i)$  sur la pile

**B** ... calcul du sinus

Résultat du calcul  $(8.4716.. + 5.4126...i)$

**B'** ... calcul inverse pour contrôle (arcsinus), résultat :  $(1 + 3i)$

## ML-07 Calcul d'un polynôme

n	i;ai	x->P(x)		

Le programme ML-07 calcule la valeur du polynôme  $P(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$ , spécifié par les valeurs des coefficients.

**Lbl A** Ordre du polynôme

L'opération **A** définit l'ordre du polynôme 'n'.

**Lbl B** Saisie des coefficients

L'opération **B** permet de saisir les valeurs des coefficients du polynôme. L'entrée de la fonction est l'indice du coefficient  $i=0..n$ , à partir duquel les valeurs seront saisies. Saisir ensuite les valeurs, continuer avec la touche **R/S**.

**Lbl C** Calcul de la valeur

L'opération **C** calcule la valeur du polynôme pour la valeur donnée 'x'.

Exemple, polynome  $P(x) = 2 - 3x + x^2$ :

**Pgm 07** ... activation du programme de bibliothèque ML-07

**2 A** ... définir l'ordre du polynôme  $n=2$

**0 B** ... saisir les coefficients à partir de l'index  $i=0$

**2 R/S 3 +/- R/S 1 R/S** ... saisie des coefficients  $a_0, a_1, a_2$

**2 C [0]** ... calculer la valeur du polynôme  $P(2) = 0$

**1 +/- C [6]** ... calculer la valeur du polynôme  $P(-1) = 6$

## ML-08 Zéros d'une fonction

ZERO			f(x)	GRAPH

Le programme ML-08 recherche numériquement les passages par zéro de la fonction.

La première étape consiste à créer une fonction intitulée **Lbl A** dans le programme principal de l'utilisateur. L'entrée de la fonction est la valeur  $x$ , la sortie est la valeur  $y$ . Les fonctions ne doivent pas utiliser l'instruction **=** ou l'instruction **CLR**.

### **Lbl A** Recherche de zéro

Entrez la valeur finale 'xmax' jusqu'à laquelle la recherche du zéro doit avoir lieu et appuyez sur **X<>T**. Entrez la valeur de départ 'xmin' à partir de laquelle rechercher et appuyez sur **A**. Après quelques secondes, le programme affichera la valeur 'x' de passage de la fonction par zéro, ou il clignotera 9,999+9999 pour indiquer que le passage par zéro n'a pas été trouvé. Si une valeur a été trouvée, vous pouvez appuyer à nouveau sur **A**, le programme recherchera le prochain passage par zéro à partir de l'emplacement spécifié, en laissant donc la dernière valeur 'x' trouvée à l'écran. La limite supérieure de l'intervalle reste toujours dans le registre T.

La recherche du zéro commence à partir d'une valeur de 'x' légèrement supérieure à l'origine spécifiée de 'xmin', afin que la dernière valeur trouvée de 'x' ne soit pas répétée à plusieurs reprises. Si le passage par zéro se situe dans la valeur initiale 'xmin', choisissez une valeur par défaut inférieure 'xmin'.

Lors de la recherche de zéro, le programme divise d'abord l'intervalle spécifié 'xmin' à 'xmax' en 100 segments. Dans chaque section, il teste s'il passe par zéro, c'est-à-dire si le signe de la valeur de la fonction change. S'il trouve une section de passage à zéro, il recherche la valeur exacte de l'emplacement 'x' de passage à zéro, en utilisant la méthode de bisection d'intervalle. La recherche s'effectue jusqu'à la résolution maximale, lorsque la largeur de la section testée dépasse les données affichables.

La fonction ne trouvera pas de passage par zéro si la courbe ne touche que l'axe 'x' à son minimum ou à son maximum, c'est-à-dire il n'y aura aucun changement de signe au point de contact.

Le début et la fin de l'intervalle testé doivent être choisis judicieusement. Si l'intervalle est choisi trop grand, certaines passes risquent d'être manquées, car il peut y avoir plusieurs passes dans une section testée au 1/100 (la section ne montre pas de changement de signe). En revanche, si l'intervalle est choisi trop petit, certains passages peuvent rester en dehors de la zone testée.

### **Lbl** **D** Valeur de la fonction

L'opération **D** calcule la valeur de la fonction **A** dans le programme principal pour une valeur donnée de x.

### **Lbl** **E** Graphique

L'opération **E** trace le graphique de la fonction sur l'écran. Entrez la valeur 'xmax' de la fin de l'intervalle à tracer, appuyez sur **x<>t**, entrez la valeur 'xmin' du début de l'intervalle à tracer et appuyez sur **E**. L'affichage du graphique peut être terminé en appuyant sur n'importe quel bouton (sauf **2nd**). Le graphique dessiné est normalisé - la valeur maximale et minimale de la fonction f(x) est recherchée et le graphique est étiré jusqu'à la plage maximale de valeurs.

Exemple, passages par zéro de la fonction  $f(x) = 4 \cdot \sin(x) + 1 - x$ :

**RST** **LRN** ... activation du mode programmation

**Lbl** **A** ... étiquette du début de la fonction **A**

**(** **STO** **10** ... stocke la valeur de 'x' dans le registre R10

**sin** **x** **4** ...  $4 \cdot \sin(x)$

**+ 1** **RCL** **10** **)** ...  $+ 1 - x$

**RTN** ... fin du sous-programme (**INV** **SBR**)

**LRN** ... quitter le mode programmation

**Pgm 08** ... activation du programme de bibliothèque ML-08

**Rad** ... l'axe 'x' représentera l'angle spécifié en radians

**3** **x<>t** **3** **+/-** ... recherche dans l'intervalle de -3 à +3

**A** [-2.2100...] ... premier passage par zéro = -2.2100...

**A** [-0.3421...] ... deuxième passage par zéro = -0.3421...

**A** [2.7020...] ... troisième passage par zéro = 2.7020...

**A** [9.999+9999] ... pas d'autre passage par zéro trouvé

**CLR** ... effacer l'indication d'erreur

**3** **x<>t** **3** **+/-** **E** ... graphe dans l'intervalle de -3 à +3



## ML-09 Approximation de Simpson (continue)

f(x)				
x0	xn	n	->I	GRAPH

Le programme ML-09 calcule la valeur approximative de l'intégrale d'une fonction entre deux bornes 'x0' et 'xn' en utilisant la méthode de Simpson.

La première étape consiste à créer une fonction intitulée **Lbl A'** dans le programme principal de l'utilisateur. L'entrée de la fonction est la valeur x, la sortie est la valeur y. Les fonctions ne doivent pas utiliser l'instruction **=** ou l'instruction **CLR**.



## **Lbl A** Limite inférieure de $x_0$

La touche **A** permet de saisir la limite inférieure ' $x_0$ ' de l'intégrale calculée.

## **Lbl B** Limite supérieure de $x_n$

La touche **B** permet de saisir la limite supérieure ' $x_n$ ' de l'intégrale calculée.

## **Lbl C** Nombre de sous-intervalles $n$

La touche **C** permet de saisir le nombre de sous-intervalles ' $n$ ' en lesquels l'intervalle spécifié sera divisé. Le nombre de sous-intervalles doit être un nombre pair, sinon il sera automatiquement corrigé en un nombre pair.

## **Lbl D** Calcul de l'intégrale

L'opération **D** calcule l'intégrale de la fonction utilisateur **A** dans l'intervalle donné ' $x_0$ ' à ' $x_n$ ' avec le nombre d'étapes ' $n$ ' donné, en utilisant l'approximation de Simpson.

## **Lbl E** Graphe

L'opération **E** trace le graphique de la fonction dans l'intervalle spécifié ' $x_0$ ' à ' $x_n$ '. L'affichage du graphique peut être terminé en appuyant sur n'importe quel bouton (sauf **2nd**). Le graphique dessiné est normalisé - la valeur maximale et minimale de la fonction  $f(x)$  est recherchée et le graphique est étiré jusqu'à la plage maximale de valeurs.

## **Lbl A'** Valeur de la fonction

L'opération **A'** calcule la valeur de la fonction **A'** dans le programme principal pour la valeur donnée de  $x$ .

Exemple, intégrale de la fonction  $1/(\cos(x) + 2)$  comprise entre 0 et  $\pi/2$  :

**RST** **LRN** ... activation du mode programmation

**Lbl** **A'** ... étiquette du début de la fonction **A'**

**Rad** ... l'axe 'x' représentera l'angle spécifié en radians

**(** **cos** **+** **2** **)** **1/x** ... fonction  $1/(\cos(x) + 2)$

**RTN** ... fin du sous-programme (**INV** **SBR**)

**LRN** ... quitter le mode programmation

**Pgm** **09** ... activation du programme de bibliothèque ML-09

**0** **A** ... début de l'intervalle 'x0'

**pi** **:** **2** **=** **B** ... fin d'intervalle 'xn'

**2** **0** **C** ... nombre de sous-intervalles 'n'

**D** [0.6046...] ... calcul de l'intégrale

**pi** **B** **E** ... tracer le graphique dans l'intervalle 0 à pi



**2** **\*** **pi** **=** **B** **+/-** **A** **E** ... tracer le graphique dans l'intervalle  $-2*\pi$  à  $+2*\pi$



## ML-10 Approximation de Simpson (discrète)

n	dx	i,fi	->I	

Le programme ML-10 calcule une intégrale numérique à partir de valeurs discrètes en utilisant l'approximation de Simpson.

**Lbl A** Nombre de sous-intervalles 'n'

La touche **A** permet de saisir le nombre de sous-intervalles 'n' en lesquels l'intervalle calculé est divisé. Le nombre de sous-intervalles doit être un nombre pair. Le nombre d'échantillons d'entrée est « n+1 » (y<sub>0</sub> à y<sub>n</sub>).

**Lbl B** Incrément 'dx'

La touche **B** permet de saisir l'incrément 'dx' de la coordonnée x.

**Lbl C** Saisie des valeurs

La touche **C**, permet de saisir l'index (0...n) à partir duquel les valeurs 'y' sont saisies, ensuite chaque valeur saisie sera validée avec la touche **R/S**.

**Lbl D** Calcul de l'intégrale

L'opération **D** calcule la valeur de l'intégrale pour les valeurs données y<sub>0</sub>...y<sub>n</sub>.

Exemple:

**Pgm 10** ... activation du programme de bibliothèque ML-10

**4 A** ... nombre de sous-intervalles n = 4

**1 B** ... incrément dx = 1

**0 C** ... index de début de saisie des valeurs, i = 0

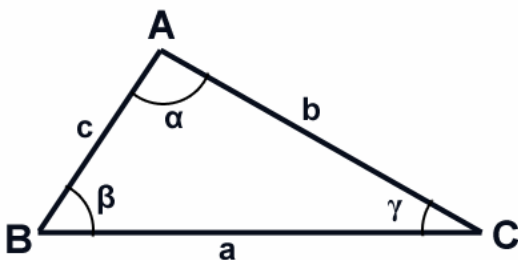
1 R/S 8 R/S 2 7 R/S 6 4 R/S 1 2 5 R/S ... saisie des valeurs

D [156] ... calcul de l'intégrale, I = 156

## ML-11 Résolution d'un triangle (1)

				PERIM
SSS	SSA	ASS	SAS	AREA

Le programme ML-11 résout des triangles spécifiés principalement par des côtés (3 côtés ou 2 côtés et 1 angle).



Les sommets du triangle sont nommés 'A', 'B', 'C'. Les longueurs des côtés, situées du côté opposé au sommet, sont marquées 'a', 'b', 'c'. Les angles sous-tendus par les côtés adjacents au sommet sont étiquetés «  $\alpha$  » (alpha), «  $\beta$  » (bêta) et «  $\gamma$  » (gamma). Dans le programme, les angles sont marqués par les lettres « A », « B » et « C » (la calculatrice ne dispose pas des lettres nécessaires de l'alphabet grec). Les angles sont calculés dans la mesure d'angle sélectionnée.

Pour les opérations A à D, les paramètres d'entrée connus du triangle sont saisis. Après chaque valeur saisie, appuyez sur R/S pour valider. Le programme calcule les paramètres manquants ainsi que l'aire du triangle et son périmètre. Continuez à afficher les résultats en appuyant à nouveau sur R/S. Il n'est pas nécessaire de poursuivre toutes les opérations avec R/S pour commencer un nouveau calcul.

## **Lbl** **A** Méthode SSS

L'opération **A**, permet de saisir les longueurs des côtés « a », « b » et « c » en validant chaque valeur avec **R/S**. Le programme calcule les angles 'A', 'B', 'C', l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl** **B** Méthode SSA

L'opération **B**, permet de saisir les longueurs des côtés « a », « b » et la valeur de l'angle « A » en validant chaque valeur avec **R/S**. Le programme calcule la longueur du côté « c », les valeurs des angles « B », « C », l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl** **C** Méthode ASS

L'opération **C**, permet de saisir la valeur de l'angle « B », les longueurs des côtés « a » et « b » en validant chaque valeur avec **R/S**. Le programme calcule la longueur du côté 'c', les valeurs des angles 'A', 'C', l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl** **D** Méthode SAS

L'opération **D**, permet de saisir la longueur du côté « a », la valeur de l'angle « C » et la longueur du côté « b » en validant chaque valeur avec **R/S**. Le programme calcule la longueur du côté 'c', les valeurs des angles 'A', 'B', l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl** **E** Aire du triangle

L'opération **E** affiche l'aire du triangle calculée à partir des valeurs saisies avec l'une des opérations A à D.

## **Lbl** **E'** Périmètre du triangle

L'opération **E** affiche le périmètre du triangle calculé à partir des valeurs saisies avec l'une des opérations A à D.

Exemple:

**Pgm** **11** ... activation du programme de bibliothèque ML-11

**Deg** ... choix de l'unité de mesure d'angle

**A** ... utilisation méthode SSS

**2** **5** **R/S** **4** **0** **R/S** **5** **8** **R/S** ... saisie longueurs des 3 côtés 25, 40 et 58

[20.7509...] ... angle calculé A = 20.7509...

**R/S** [34.5336...] ... angle calculé B = 34.5336...

**R/S** [124.715...] ... angle calculé C = 124.715...

**R/S** [410.99...] ... aire 410.99...

**R/S** [123] ... périmètre 123

**B** ... utilisation méthode SSA

**3** **.** **4** **6** **R/S** **5** **.** **6** **R/S** **1** **5** **.** **5** **R/S** ... saisie côtés 3.46 et 5.6, angle 15.5

[8.5159...] ... côté calculé c = 8.5159...

**R/S** [25.627...] ... angle calculé B = 25.627...

**R/S** [138.87...] ... angle calculé C = 138.87...

**D** ... utilisation méthode SAS

**3** **8** **0** **R/S** **3** **8** **R/S** **3** **2** **0** **R/S** ... saisie côté 380, angle 38 et côté 320

[234.85...] ... côté calculé  $c = 234.85...$

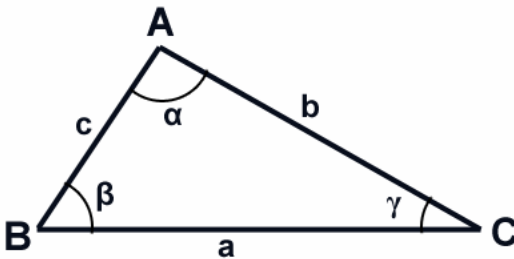
**R/S** [84.978...] ... angle calculé  $A = 84.978...$

**R/S** [57.021...] ... angle calculé  $B = 57.021...$

## ML-12 Résolution d'un triangle (2)

SAA	AAS	ASA	AREA	PERIM

Le programme ML-12 résout des triangles définis principalement par des angles (1 côté et 2 angles).



Les sommets du triangle sont nommés 'A', 'B', 'C'. Les longueurs des côtés, situées du côté opposé au sommet, sont marquées 'a', 'b', 'c'. Les angles sous-tendus par les côtés adjacents au sommet sont étiquetés «  $\alpha$  » (alpha), «  $\beta$  » (bêta) et «  $\gamma$  » (gamma). Dans le programme, les angles sont marqués par les lettres « A », « B » et « C » (la calculatrice ne dispose pas des lettres nécessaires de l'alphabet grec). Les angles sont calculés dans la mesure d'angle sélectionnée.

Pour les opérations **A** à **C**, les paramètres d'entrée connus du triangle sont saisis. Après chaque valeur saisie, appuyez sur **R/S** pour valider. Le programme calcule les paramètres manquants ainsi que l'aire du triangle et son périmètre. Continuez à afficher les résultats en appuyant à nouveau sur **R/S**. Il n'est pas nécessaire de poursuivre toutes les opérations avec **R/S** pour commencer un nouveau calcul.

## **Lbl A** Méthode SAA

L'opération **A**, permet de saisir la longueur du côté « a » et les valeurs des angles « A », « B » en validant chaque valeur avec **R/S**. Le programme calcule la longueur des côtés « b », « c » et la valeur de l'angle « C », l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl B** Méthode AAS

L'opération **B**, permet de saisir les valeurs des angles « A », « C » et la longueur du côté « a » et les valeurs des angles « A », « B » en validant chaque valeur avec **R/S**. Le programme calcule la longueur des côtés « b », « c » et la valeur de l'angle « B », l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl C** Méthode ASA

L'opération **C**, permet de saisir la valeur de l'angle « B », la longueur du côté « a » et la valeur de l'angle « C » en validant chaque valeur avec **R/S**. Le programme calcule la longueur des côtés « b », « c » et la valeur de l'angle « A », l'aire du triangle et son périmètre. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

## **Lbl D** Aire du triangle

L'opération **D** affiche l'aire du triangle calculée à partir des valeurs saisies avec l'une des opérations A à C.

## **Lbl E** Périmètre du triangle

L'opération **E** affiche le périmètre du triangle calculé à partir des valeurs saisies avec l'une des opérations A à C.

Exemple:

**Pgm 12** ... activation du programme de bibliothèque ML-12



**Deg** ... choix de l'unité de mesure d'angle

**C** ... utilisation méthode ASA

**1 1 0 R/S 1 8 R/S 5 2 . 2 R/S** ... saisie angle 110, côté 18, angle 52.2

[55.331...] ... côté calculé  $b = 55.331...$

**R/S** [46.526...] ... côté calculé  $c = 46.526...$

**R/S** [17.8] ... angle calculé  $A = 17.8$

**B** ... utilisation méthode AAS

**1 7 . 8 R/S 8 5 . 4 R/S 2 . 2 5 R/S** ... angles 17.8 et 85.4, côté 2.25

[7.1658...] ... côté calculé  $b = 7.1658...$

**R/S** [7.3365...] ... côté calculé  $c = 7.3365...$

**R/S** [76.8] ... angle calculé  $B = 76.8$

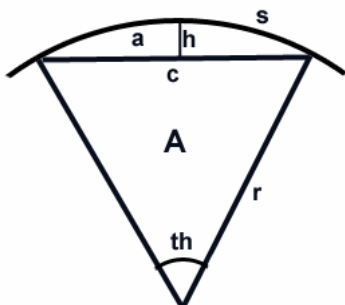
**R/S** [8.0355...] ... aire 8.0355...

**R/S** [16.7523...] ... périmètre 16.7523...

## ML-13 Calcul d'arcs de cercles

th,r	th,s	th,c	r,s	r,c

Le programme ML-13 résout l'arc circulaire, le segment circulaire et le segment de ligne circulaire.



th = angle au centre  $\theta$   
r = rayon  
s = longueur de l'arc  
c = longueur de la corde  
A = surface du secteur  
a = surface du segment  
h = hauteur du segment

**Lbl** **A** Saisie angle th et rayon r

La touche **A** permet la saisie de l'angle central 'th' et du rayon 'r' en validant chaque valeur avec **R/S**. Le programme calcule la longueur de l'arc 's', la longueur de la corde 'c', l'aire du segment 'A', l'aire du segment 'a' et la hauteur du segment 'h'. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

**Lbl** **B** Saisie angle th et longueur de l'arc s

La touche **B** permet la saisie de l'angle central 'th' et de la longueur de l'arc 's'. Le programme calcule le rayon 'r', la longueur de la corde 'c', l'aire du segment 'A', l'aire du segment 'a' et la hauteur du segment 'h'. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

**Lbl** **C** Saisie angle th et longueur de la corde c

La touche **C** permet la saisie de l'angle central 'th' et de la longueur de la corde 'c'. Le programme calcule le rayon 'r', la longueur de l'arc 's', l'aire du

segment 'A', l'aire du segment 'a' et la hauteur du segment 'h'. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

**Lbl** **D** Saisie rayon r et longueur de l'arc s

La touche **D** permet la saisie du rayon 'r' et de la longueur de l'arc 's'. Le programme calcule l'angle au centre 'th', la longueur de la corde 'c', l'aire du segment 'A', l'aire du segment 'a' et la hauteur du segment 'h'. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

**Lbl** **E** Saisie rayon r et longueur de la corde c

La touche **E** permet la saisie du rayon 'r' et de la longueur de corde 'c'. Le programme calcule l'angle au centre 'th', la longueur de l'arc 's', l'aire du segment 'A', l'aire du segment 'a' et la hauteur du segment 'h'. Appuyez sur **R/S** pour continuer après chaque affichage de résultat.

Exemple:

**Pgm** **13** ... activation du programme de bibliothèque ML-13

**Deg** ... choix de l'unité de mesure d'angle

**A** **6** **2** **R/S** **1** **5** **R/S** ... saisie angle  $th = 62^\circ$  et rayon  $r = 15$

[16.2315...] ... longueur de l'arc  $s = 16.2315...$

**R/S** [15.4511...] ... longueur de la corde  $c = 15.4511...$

**R/S** [121.736...] ... surface du secteur  $A = 121.736...$

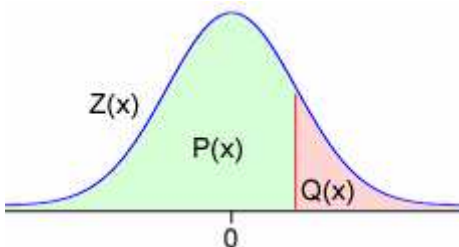
**R/S** [22.405...] ... surface du segment  $a = 22.405...$

**R/S** [2.1424...] ... hauteur du segment  $h = 2.1424...$

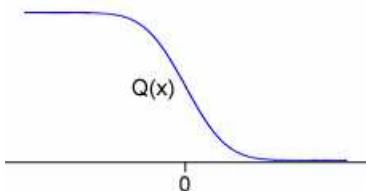
## ML-14 Distribution normale

GRAPH	GRAPH	GRAPH		
Z(x)	Q(x)	P(x)		

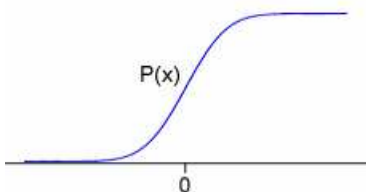
Le programme ML-14 calcule la distribution de probabilité normale standard (ou gaussienne).



La courbe de distribution normale est décrite par l'expression  $Z(x) = 1/\sqrt{2\pi} \cdot \exp(-x^2/2)$ .



L'intégrale (aire) de la courbe à partir du point limite spécifié Q(x) est appelée distribution gaussienne complémentaire, ou également fonction Q.



L'intégrale (aire) de la courbe jusqu'au point de coupure spécifié P(x) est appelée distribution normale cumulative.

**Lbl** **A** Distribution standard Z(x)

L'opération **A** calcule la distribution de probabilité normale standard Z(x) pour « x » dans la plage -150 à +150. Il y a des points d'inflexion aux positions  $x = -1$  et  $+1$ .

### **Lbl** **A'** Graphe de la distribution standard

L'opération **A'** affiche un tracé de la distribution normale standard  $Z(x)$ . Terminer l'affichage du graphique en appuyant sur n'importe quelle touche (sauf la **2nd**).

### **Lbl** **B** Distribution complémentaire $Q(x)$

L'opération **B** calcule la distribution de probabilité complémentaire  $Q(x)$  pour 'x' compris entre -8 et +8.

### **Lbl** **B'** Graphe distribution complémentaire

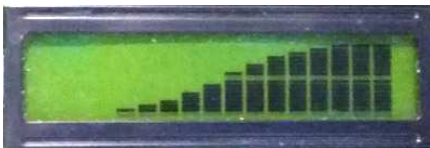
L'opération **B'** affiche le graphique de la distribution complémentaire de  $Q(x)$ . Terminer l'affichage du graphique en appuyant sur n'importe quelle touche (sauf la **2nd**).

### **Lbl** **C** Distribution normale cumulative $P(x)$

L'opération **C** calcule la distribution de probabilité cumulée  $P(x)$  pour « x » dans la plage -8 à +8.

### **Lbl** **C'** Graphe de la distribution cumulée

L'opération **C'** affiche le graphique de la distribution cumulée de  $P(x)$ . Terminer l'affichage du graphique en appuyant sur n'importe quelle touche (sauf **2nd**).



## ML-15 Génération de nombres aléatoires

A,x	B,s	No(A,B)	Int(A,B)	No(x,s)

Le programme ML-15 utilise le générateur pseudo-aléatoire interne de la calculatrice Rand, basé sur la méthode du générateur congruentiel linéaire (LCG), pour générer des nombres aléatoires.

**Lbl A** Limite inférieure « A » ou centre 'x'

L'opération **A** est utilisée pour spécifier une limite inférieure pour générer un nombre aléatoire uniforme 'A' ou pour spécifier un point médian 'x' pour générer un nombre aléatoire normal.

**Lbl B** Limite supérieure de 'B' ou écart 's'

L'opération **B** est utilisée pour spécifier une limite supérieure afin de générer un nombre aléatoire uniforme 'B' ou pour spécifier une variance 's' afin de générer un nombre aléatoire normal.

**Lbl C** Caractère aléatoire uniforme

L'opération **C** génère un nombre décimal aléatoire avec une distribution uniforme dans l'intervalle de 'A' (inclus) à 'B' (exclus).

**Lbl D** Caractère aléatoire entier uniforme

L'opération **D** génère un nombre entier aléatoire avec une distribution uniforme dans l'intervalle de 'A' (inclus) à 'B' (exclus).

**Lbl E** Hasard normal

L'opération **E** génère un nombre décimal aléatoire avec une distribution normale standard (gaussienne) de centre 'x' et de variance 's' selon la formule  $y = s * \sqrt{-2 * \ln(\text{rnd1})} * \cos(2 * \pi * \text{rnd2}) + x$ .

## ML-16 Combinaisons, permutations, factorielles

$n!$	$\ln n!$	V rep	P rep	C rep
n	r	V	P	C

Le programme ML-16 est utilisé pour calculer le nombre de variations, de permutations, de combinaisons et de factorielles. La variation signifie une sélection ordonnée d'éléments « r » à partir d'un ensemble « n » (l'ordre des éléments compte). Une permutation est un cas particulier de variation lorsque tous les éléments ( $r = n$ ) sont sélectionnés. La combinaison signifie une sélection non ordonnée d'éléments « r » dans un ensemble « n » (l'ordre des éléments n'a pas d'importance). Les sélections peuvent se faire soit avec répétition éventuelle des éléments, soit sans répétition.

**Lbl A** Nombre total d'éléments 'n'

L'opération **A** est utilisée pour spécifier le nombre total d'éléments 'n'.

**Lbl B** Nombre d'éléments de sélection 'r'

L'opération **B** permet de préciser le nombre d'éléments de la sélection 'r'. Ce n'est pas nécessaire dans le calcul de permutation.

**Lbl C** Variation sans répétition

L'opération **C** calcule le nombre de variations (l'ordre dépend) de la sélection des éléments « r » dans l'ensemble « n », sans répéter les éléments.

**Lbl D** Permutation sans répétition

L'opération **D** calcule le nombre de permutations (l'ordre dépend) d'un ensemble de 'n' éléments, sans éléments répétitifs.

**Lbl E** Combinaison sans répétition

L'opération **E** calcule le nombre de combinaisons (l'ordre n'a pas

d'importance) de sélection d'éléments « r » dans l'ensemble « n », sans répéter les éléments.

### **Lbl A'** Factorielle n!

L'opération **A'** calcule la factorielle du nombre 'n'.

### **Lbl B'** Le logarithme de la factorielle ln n!

L'opération **B'** calcule le logarithme népérien de la factorielle de 'n'.

### **Lbl C'** Variation avec répétition

L'opération **C'** calcule le nombre de variations (l'ordre dépend) de la sélection des éléments 'r' de l'ensemble 'n', les éléments se répétant.

### **Lbl D'** Permutation avec répétition

L'opération **D'** calcule le nombre de permutations (l'ordre dépend) d'un ensemble de 'n' éléments, les éléments se répétant.

### **Lbl E'** Combinaison avec répétition

L'opération **E'** calcule le nombre de combinaisons (l'ordre n'a pas d'importance) de sélection d'éléments 'r' dans l'ensemble 'n', en répétant les éléments.

Exemples: Sélectionner **Pgm 16** avant chaque exemple.

Exemple 1:

Il y a 4 livres sur l'étagère. Combien de possibilités pour ranger les livres ?

**4 A** ... saisir le nombre total d'éléments  $n = 4$

**D [24]** ... nombre de permutations sans répétition = 24



Exemple 2:

25 étudiants postulent pour une bourse. Les 3 premiers auront un montant de bourse (différent selon le rang). Combien de résultats possibles ?

... saisir le nombre total d'éléments  $n = 25$

... saisir le nombre d'éléments de sélection  $r = 3$

[13800] ... nombre de variations sans répétition = 13800

Exemple 3:

Il y a 52 cartes dans le paquet. Un joueur choisit 4 cartes. Combien y a-t-il de possibilités ?

... saisir le nombre total d'éléments  $n = 52$

... saisir le nombre d'éléments de sélection  $r = 4$

[270725] ... nombre de combinaisons sans répétition = 270725

Exemple 4:

Il y a 26 lettres dans l'alphabet. Combien d'entre eux pouvons-nous transformer en groupes de 3 lettres ?

... saisir le nombre total d'éléments  $n = 26$

... saisir le nombre d'éléments de sélection  $r = 3$

[17576] ... nombre de variations avec répétition = 17576

Exemple 5:

Combien de manières de répartir 20 billets entre 10 personnes ? Chacun pouvant obtenir de 0 à 20 billets. Nombre de combinaisons de 20 éléments dans un ensemble répétitif de 10 éléments ?

... saisir le nombre total d'éléments  $n = 10$

... saisir le nombre d'éléments de sélection  $r = 20$

[10015005] ... nombre de combinaisons avec répétition = 10015005

## ML-17 Moyennes mobiles

n	m-AVG			

Le programme ML-17 calcule une moyenne mobile, c'est-à-dire la moyenne des dernières valeurs saisies.

**Lbl A** Nombre d'échantillons

La fonction **A** permet de saisir le nombre d'échantillons dans l'ensemble des dernières valeurs saisies.

**Lbl B** Ajout d'un échantillon

La fonction **B** ajoute un échantillon et calcule la moyenne mobile.

Exemple:

**Pgm 17** ... activation du programme de bibliothèque ML-17

**3 A** ... définition de la taille de la fenêtre d'échantillonnage = 3

**4 5 B** [45] ... ajout de l'échantillon 45, moyenne = 45

**5 0 B** [47.5] ... ajout de l'échantillon 50, moyenne = 47.5

**5 7 B** [50.6666...] ... ajout de l'échantillon 57, moyenne = 50.6666...

**6 5 B** [57.3333...] ... ajout de l'échantillon 65, moyenne = 57.3333...

**7 3 B** [65] ... ajout de l'échantillon 73, moyenne = 65

**8 1 B** [73] ... ajout de l'échantillon 81, moyenne = 73

**8 4 B** [79.3333...] ... ajout de l'échantillon 84, moyenne = 79.3333...

**8 4 B** [83] ... ajout de l'échantillon 84, moyenne = 83

**7 8 B** [82] ... ajout de l'échantillon 78, moyenne = 82

## ML-18 Intérêts composés (Méthode US)

S	$(1+i)S$	a	$(1+i)a$	INIT
N	%I	PV	FV	

Le programme ML-18 calcule les intérêts composés.

**Lbl A** Nombre de périodes N

L'opération **A** permet de saisir le nombre de périodes N. Si 0, le nombre de périodes est calculé.

**Lbl B** Taux d'intérêt %I

L'opération **B** permet de saisir le taux d'intérêt en % par période. Si 0, le taux d'intérêt est calculé.

**Lbl C** Valeur actuelle PV

L'opération **C** permet de saisir la valeur actuelle PV (Present Value). Si 0, la valeur actuelle est calculée.

**Lbl D** Valeur future FV

L'opération **D** permet de saisir la valeur future FV (Future Value). Si 0, la valeur future est calculée.

**Lbl A'** Fonds d'amortissement Sni

L'opération **A'** calcule les annuités du fonds d'amortissement Sni.

**Lbl B'** Annuité due FV  $(1+i)Sni$

L'opération **B'** calcule l'annuité due pour la valeur future FV  $(1+i)Sni$ .

## **Lbl C'** Annuités ordinaires ani

L'opération **C'** calcule les annuités ordinaires pour la valeur actuelle PV 'ani'.

## **Lbl D'** Annuité due PV (1+i)ani

L'opération **D'** calcule l'annuité due pour la valeur actuelle PV (1+i)ani.

## **Lbl E'** Initialisation

L'opération **E'** initialise le programme.

### Exemple 1:

**Pgm 18** ... activation du programme de bibliothèque ML-18

**E'** ... initialisation

**2 4 A** ... nombre de périodes = 24 mois

**5 | 7 5** ... taux d'intérêt pour 1 an = 5.75%

**: 1 2 = B** [0.48] ... taux d'intérêt pour 1 mois = 0.48%

**5 0 0 C** ... valeur actuelle PV = 500

**0 D** [560.78] ... calcul de la valeur PV future dans 12 mois = 560.78

### Exemple 2:

**Pgm 18** ... activation du programme de bibliothèque ML-18

**E'** ... initialisation

**3 6 5 A** ... nombre de périodes = 365 jours

**5 | 7 5** ... taux d'intérêt pour 1 an = 5.75%

**: 3 6 5 = B** [0.02] ... taux d'intérêt pour 1 jour = 0.02%

**1 0 0 0 C** ... valeur actuelle PV = 1000

0 D [1059.18] ... calcul de la valeur PV future dans 365 jours = 1059.18

4 A ... nouveau nombre de périodes = 4 trimestres

6 ... taux d'intérêt différent pendant 1 an = 6%

: 4 = B [1.50] ... taux d'intérêt pour 1 trimestre = 1.50%

0 D [1061.36] ... valeur PV future différée sur 4 trimestres = 1061.36

#### Exemple 3:

Pgm 18 ... activation du programme de bibliothèque ML-18

E ... initialisation

1 2 A ... nombre de périodes = 12 mois

1 C ... valeur actuelle PV = 1

1 1 0 5 7 5 D ... valeur future FV = 1.0575 (augmentée de 5.75%)

0 B [0.47] ... calcul du taux d'intérêt requis pour 1 mois = 0.47%

2 4 A ... nombre de périodes = 24 mois

5 0 0 C ... valeur actuelle PV = 500

0 D [559.15] ... calcul de la valeur PV future dans 24 mois = 559.15

#### Exemple 4:

Pgm 18 ... activation du programme de bibliothèque ML-18

E ... initialisation

1 3 A ... nombre de périodes = 13 mois

1 2 3 4 C ... valeur actuelle PV = 1234

1 3 0 0 D ... valeur future FV = 1300

0 B [0.40] ... calcul du taux d'intérêt requis pour 1 mois = 0.40%

1 2 A ... nombre de périodes = 12 mois

1 C ... valeur actuelle PV = 1

0 D [1.05] ... calcul de la valeur future FV = 1.05

: 1 = x 1 0 0 = [4.93] ... l'augmentation est d'environ 4.93%

## ML-19 Annuités

sink	dueFV	ordPV	duePV	INIT
N	%I	PMT	PV/FV	B.PMT

Le programme ML-19 traite des remboursements des annuités.

Une annuité consiste en une série de paiements égaux effectués à des intervalles de temps réguliers. L'annuité est un cas d'intérêts composés avec paiements périodiques. Si les paiements sont effectués à la fin de la période, l'annuité est appelée annuité ordinaire. Si les paiements sont effectués au début de chaque période, on dit qu'il s'agit d'annuités à échoir.

Il y a maintes situations financières qui impliquent non seulement des séries de paiements mais également un dernier paiement pouvant être inférieur ou supérieur aux paiements réguliers. On appelle ceci des paiements libératoires pouvant s'appliquer au remboursement anticipé d'un prêt ou encore à la vente de biens ayant rapporté un flot régulier de loyers, provoquant ainsi un important apport de revenus à la fin de l'investissement. Un paiement libératoire est égal au paiement résiduel principal à cet instant.

Ce programme traite quatre catégories d'annuités :

- Fonds d'amortissement
- Annuité à échoir /FV
- Annuité ordinaire /PV
- Annuité à échoir /PV

**Lbl** **A** Nombre de périodes N

L'opération **A** permet de saisir le nombre de périodes N. Si 0, le nombre de périodes est calculé.

**Lbl** **B** Taux d'intérêt %I

L'opération **B** permet de saisir le taux d'intérêt en % par période. Si 0, le taux d'intérêt est calculé.

## **Lbl C** Paiement de la période PMT

L'opération **C** permet de saisir le paiement de la période PMT. Si 0, le paiement par période est calculé.

## **Lbl D** Valeur actuelle ou future PV/FV

L'opération **D** permet de saisir la valeur actuelle PV (Present Value) ou la valeur future FV (Future Value). Si 0, la valeur est calculée.

## **Lbl E** Paiement libératoire BAL

l'opération **E** permet de saisir le paiement libératoire (versement résiduel en fin de période). Si 0, le paiement libératoire est calculée.

## **Lbl A'** Fonds d'amortissement

L'opération **A'** concerne les fonds d'amortissement (épargne pour régler les dettes, fonds d'amortissement).

## **Lbl B'** Annuités à échoir FV

L'opération **B'** concerne les échéances de la future rente FV.

## **Lbl C'** Annuités ordinaires PV

L'opération **C'** concerne les échéances de la rente ordinaire.

## **Lbl D'** Annuités échues PV

L'opération **D'** concerne les échéances de la rente d'épargne.

## **Lbl E'** Initialisation

L'opération **E'** initialise le programme.

### Exemple 1, fonds d'amortissement :

**Pgm 19** ... activation du programme de bibliothèque ML-19

**E'** ... initialisation

**A'** ... fonds d'amortissement

**4** | **5** **x** ... nombre d'années = 4.5

**1** **2** **=** **A** [54] ... nombre de mois total = 54

**5** | **2** **5** ... taux d'intérêt pour 1 an = 5.75%

**:** **1** **2** **=** **B** [0.4375] ... taux d'intérêt pour 1 mois = 0.4375%

**2** **5** **C** ... paiement par période (mois) = 25

**0** **D** [1519.08] ... calcul de la valeur PV future dans 4,5 ans = 1519.08

**1** **0** **x** **1** **2** **=** **A** [120] ... nouvelle période de 10 ans = 120 mois

**0** **D** [3934.42] ... calcul de la valeur PV future dans 10 ans = 3934.42

### Exemple 2, annuités à échoir FV :

**Pgm 19** ... activation du programme de bibliothèque ML-19

**E'** ... initialisation

**B'** ... annuités à échoir FV

**1** **0** **0** **0** **0** **D** ... valeur actuelle = 10000

**5** **0** **C** ... paiement par période (mois) = 50

**1** **0** **x** **1** **2** **=** **A** [120] ... période de 10 ans = 120 mois

**0** **B** [0.7869] ... taux d'intérêt calculé = 78.69%

### Exemple 3, annuités ordinaires PV :

**Pgm 19** ... activation du programme de bibliothèque ML-19

**E'** ... initialisation

**C'** ... annuités ordinaires PV



8 | . | 7 | 5 | = ... taux d'intérêt annuel = 8.75%

| : | 1 | 2 | = | B | [0.7292] ... taux d'intérêt pour 1 mois = 0.7292%

3 | 2 | 0 | 0 | 0 | D | ... valeur future FV = 32000

3 | 0 | x | 1 | 2 | = | A | [360] ... période de 30 ans = 360 mois

0 | C | [251.74] ... calcul PMT pour 1 mois = 251.74

2 | 0 | x | 1 | 2 | = | A | [240] ... période de 20 ans = 240 mois

0 | C | [282.79] ... calcul PMT pour 1 mois = 282.79

#### Exemple 4, annuités échues PV :

Pgm | 19 | ... activation du programme de bibliothèque ML-19

E | ... initialisation

D | ... annuités échues PV

4 | 5 | 0 | 0 | 0 | D | ... valeur future FV = 45000

2 | 0 | 0 | 0 | C | ... paiement par période (mois) = 2000

1 | 0 | 0 | 0 | 0 | E | ... paiement libératoire BAL = 10000

2 | x | 1 | 2 | = | A | [24] ... nombre de périodes = 24 mois

0 | B | [1.9638] ... calcul du taux d'intérêt = 1,9638% par mois

x | 1 | 2 | = | [23.5651] ... taux d'intérêt pour 1 an = 23.5651%

# ML-20 Nombre de jour entre deux dates, jour de la semaine

(MMDD.YYYY)				AbsDay
Date1	Date2	NoDays	DayOfWeek	Julian

Le programme ML-20 calcule le jour de la semaine et le nombre de jours entre deux dates. La plus petite date prise en charge est le 01/01/1583.

## Lbl A Saisie date 1

L'opération A permet de saisir la première date, sous la forme MMJJ.AAAA (mois, jour et année).

## Lbl B Saisie date 2

L'opération B permet de saisir la deuxième date, sous la forme MMJJ.AAAA (mois, jour et année).

## Lbl C Nombre de jours entre les dates

L'opération C calcule le nombre de jours entre les dates spécifiées par les opérations A et B.

## Lbl D Jour de la semaine

L'opération D permet de saisir une date au format MMJJ.AAAA (mois, jour et année) et calcule le jour de la semaine. L'opération renvoie le jour de la semaine sous la forme numérique 0 à 6 :

- |              |              |
|--------------|--------------|
| 0 = samedi   | 4 = mercredi |
| 1 = dimanche | 5 = jeudi    |
| 2 = lundi    | 6 = vendredi |
| 3 = mardi    |              |

## **Lbl E** jour julien

L'opération **E** convertit une date au format MMJJ.AAAA (mois, jour et année) en jour julien, utilisé en astronomie. La date la plus basse prise en charge, le 01/01/1583, a un jour julien de 2299238. Le jour julien change à midi, le jour julien donné est pour le matin (l'après-midi est plus haut de 1). Le jour julien est le nombre de jours écoulés depuis une date conventionnelle fixée au 1<sup>er</sup> janvier de l'an 4713 av. J.-C

## **Lbl E'** Journée absolue

L'opération **E'** convertit une date sous la forme MMJJ.AAAA (mois, jour et année) en jour absolu, c'est-à-dire nombre de jours depuis le début de notre ère. La date la plus basse prise en charge, le 01/01/1583, a un jour absolu de 578179.

### Exemple 1:

**Pgm 20** ... activation du programme de bibliothèque ML-20

**6 0 1 | 1 9 6 0 A** ... saisie première date 01/06/1960

**1 0 3 1 | 1 9 7 6 B** ... saisie deuxième date 31/10/1976

**C [5996]** ... nombre de jours entre les dates = 5996

**1 0 0 1 | 1 9 7 6 A** ... autre première date 01/10/1976

**C [30]** ... nombre de jours entre les dates = 30

### Exemple 2:

**Pgm 20** ... activation du programme de bibliothèque ML-20

**1 2 0 7 | 1 9 4 1 STO 01** ... date 07/12/1941

**D [1]** ... le 12/07/1941 était un dimanche

**RCL 01 E** [2430335] ... date julienne = 2430335

## ML-21 Jeu du nombre mystérieux HILO

M INIT	M LO	M HI	M CORR	
START	GUESS	SCORE		

Le programme ML-21 est un jeu de devinettes (HI-LO game).

**Lbl A** Nouveau jeu

L'opération **A** permet de démarrer une nouvelle partie.

**Lbl B** Saisie du nombre

L'opération **B** permet de saisir le nombre estimé de 1 à 1023. La calculatrice affichera -1 si le nombre à deviner est supérieur, +1 si le nombre à deviner est inférieur et 0 clignotera si la réponse est correcte.

**Lbl C** Score

L'opération **C** permet d'afficher le nombre de coups joués.

**Lbl A'** Nouveau jeu pour la calculatrice

L'opération **A'** démarre un nouveau jeu dans lequel le joueur pense au nombre et la calculatrice le devine. Saisir le nombre que la calculatrice doit deviner puis appuyer sur **A'**. La calculatrice affichera le nombre qu'elle propose.

**Lbl B'** Estimation proposée inférieure

Appuyer sur **B'** si l'estimation de la calculatrice est inférieure au nombre souhaité.

**Lbl C'** Estimation proposée supérieure

Appuyer sur **C'** si l'estimation de la calculatrice est supérieure au nombre souhaité.

**Lbl** **D'** Estimation correcte

Appuyer sur **D'** si l'estimation de la calculatrice correspond au nombre à deviner. Le nombre de coups joués par la calculatrice s'affiche.

## ML-22 Vérification de relevés bancaires

Checking	Savings	I%/Yr	Periods/Yr	
Balance	Deposit	Withdr	No.Period	New Bal.

Le programme ML-22 est utilisé pour suivre un compte courant et un compte d'épargne. Le type de compte est sélectionné par les opérations A' ou B'. Après avoir sélectionné un type de compte, les autres opérations fonctionneront pour ce type de compte.

**Lbl** **A** Balance

L'opération **A** affiche le solde du compte sélectionné avec les touches **A'** ou **B'**.

**Lbl** **B** Dépôt

L'opération **B** permet d'ajouter un dépôt (augmente le solde du compte) au compte sélectionné avec les touches **A'** ou **B'**.

**Lbl** **C** Retrait

L'opération **C** permet d'effectuer un retrait (diminue le solde du compte) du compte sélectionné avec les touches **A'** ou **B'**.

**Lbl** **D** Nombre de période N

L'opération **D** permet de saisir le nombre de périodes déjà écoulées N avant la nouvelle transactions. Le solde d'épargne est mis à jour.

## **Lbl E** Nouvelle balance

L'opération **E** établit le nouveau solde du compte sélectionné avec les touches **A'** ou **B'**.

## **Lbl A'** Sélection compte courant

L'opération **A'** continuera à fonctionner avec le compte courant.

## **Lbl B'** Sélection compte épargne

L'opération **B'** continuera à fonctionner avec le compte épargne.

## **Lbl C'** Taux d'intérêt 1%

L'opération **C'** permet de saisir le taux d'intérêt **I** en pourcentage annuel.

## **Lbl D'** Nombre de périodes par an

L'opération **D'** permet de saisir le nombre de périodes par an.

### Exemple:

**Pgm 22** ... activation du programme de bibliothèque ML-22

**A'** ... Sélection compte courant

**2 3 1 | 7 0 E** ... saisie du solde du compte courant

**2 3 1 | 6 0 B** [463.30] ... dépôt sur le compte, nouveau solde 463.30

**5 0 B** [513.30] ... dépôt sur le compte, nouveau solde = 513.30

**4 3 | 1 0 C** [470.20] ... retrait sur le compte, nouveau solde = 470.20

**1 8 | 7 3 C** [451.47] ... retrait sur le compte, nouveau solde = 451.47

**1 0 3 | 7 9 C** [347.68] ... retrait sur le compte, nouveau solde = 347.68

**1 0 | 3 6 C** [337.32] ... retrait sur le compte, nouveau solde = 337.32

**B'** ... Sélection compte épargne

**1 7 3 2 . 8 4 E** ... saisie du solde du compte épargne

**5 C'** ... le taux d'intérêt est de 5% par an

**3 6 5 D'** ... nombre de périodes par an = 365

**1 0 D** [1735.22] ... décalage de 10 jours, nouveau solde = 1735.22

**3 0 4 B** [2039.22] ... dépôt sur le compte, nouveau solde = 2039.22

**4 D** [2040.22] ... décalage de 4 jours, nouveau solde = 2040.22

**4 2 8 B** [2468.33] ... dépôt sur le compte, nouveau solde = 2468.22

**6 D** [2470.36] ... décalage de 4 jours, nouveau solde = 2470.36

**1 0 0 0 C** [1470.36] retrait sur le compte, nouveau solde = 1470.36

**1 0 D** [1472.38] ... décalage de 10 jours, nouveau solde = 1472.38

## ML-23 Opérations sexagésimales

(dd.mmss)				
n	+ - p	* a	/ a	

Le programme ML-23 est utilisé pour les calculs avec des unités de temps et d'angle. Les données sont saisies sous la forme dd.mmss, où le nombre d'heures ou de degrés est avant le point décimal, les 2 premiers chiffres après le point décimal sont des minutes et les 2 chiffres suivants après le point décimal sont des secondes. Si les secondes contiennent des dixièmes, ils sont ajoutés sous forme de chiffres supplémentaires après les secondes.

**Lbl A** Saisie du nombre initial

L'opération **A** permet de saisir le premier nombre (temps ou angle). Ce nombre est saisi au format dd.mmss.

## **Lbl B** Ajouter ou soustraire un autre nombre

L'opération **B** permet de saisir le nombre (temps ou angle) qui doit être ajouté ou soustrait (selon le signe) au nombre initial. Si le résultat de l'opération doit être utilisé pour d'autres opérations, il doit être enregistré comme nombre initial avec la touche **A**.

## **Lbl C** Multiplier par une constante

L'opération **C** permet de multiplier le nombre initial par une constante scalaire. Si le résultat de l'opération doit être utilisé pour d'autres opérations, il doit être enregistré comme nombre initial avec la touche **A**.

## **Lbl D** Diviser par une constante

L'opération **D** permet de diviser le nombre initial par une constante scalaire. Si le résultat de l'opération doit être utilisé pour d'autres opérations, il doit être enregistré comme nombre initial avec la touche **A**.

Exemple:

**Pgm 23** ... activation du programme de bibliothèque ML-23

**8 A** ... nombre initial = 8:00:00 (8 heures)

**3 . 2 B** [11.2000] ... ajout du temps 3:20:00, résultat= 11:20:00

**4 7 . 0 0 3 1 A** ... nombre initial = 47:00:31

**2 4 . 4 3 3 5 +/- B** [22.1656] ... soustraction 24:43:35, résultat= 22:16:56

**2 0 . 3 0 4 5 A** ... nombre initial = 20:30:45

**2 C** [41.0130] ... multiplier par 2, résultat= 41.0130

**1 6 0 . 8 9 7 7 A** ... nombre initial= 160:89:77 (= 161:30:17)

**2 D** [80.4509] ... diviser par 2, résultat= 80:45:09



## ML-24 Conversions (I)

cm->in	m->ft	m->yd	km->mi	n mi->mi
in->cm	ft->m	yd->m	mi->km	mi->n mi

Le programme ML-24 est utilisé pour convertir les unités.

**Lbl A** Conversion des pouces en centimètres

**Lbl A'** Conversion des centimètres en pouces

**Lbl B** Conversion les pieds en mètres

**Lbl B'** Conversion des mètres en pieds

**Lbl C** Conversion des yards en mètres

**Lbl C'** Conversion des mètres en yards

**Lbl D** Conversion des miles britanniques en kilomètres

**Lbl D'** Conversion des kilomètres en miles britanniques

**Lbl E** Conversion des miles britanniques en nautiques

**Lbl E'** Conversion des nautiques en miles britanniques

## ML-25 Conversions (2)

°C->°F	lit->oz	lit->gal	grm->oz	kg->lb
°F->°C	oz->lit	gal->lit	oz->grm	lb->kg

Le programme ML-25 est utilisé pour convertir les unités.

**Lbl A** Conversion °F en °C

**Lbl A'** Conversion °C en °F

**Lbl B** Conversion des onces liquides en litres

**Lbl B'** Conversion de litres en onces liquides

**Lbl C** Convertir des gallons (US) en litres

**Lbl C'** Conversion des litres en gallons (US)

**Lbl D** Conversion des onces en grammes

**Lbl D'** Conversion de grammes en onces

**Lbl E** Convertir des livres en kilogrammes

**Lbl E'** Convertir des kilogrammes en livres

## ML-26 Arithmétique des fractions

INIT	$a/b \rightarrow d$	$X - Y$	$X : Y$	DEL
ENTER	$d \rightarrow a/b$	$X + Y$	$X \times Y$	$X \leftrightarrow Y$

Le programme ML-26 est utilisé pour les calculs avec des fractions. Les fractions contiennent le numérateur dans le registre T et le dénominateur de la fraction dans le registre X (contenu affiché). Une pile de 10 nombres fractionnaires est utilisée pour les opérations fractionnaires, située dans les registres de données R10 à R29. Lors des opérations, les opérandes nécessaires sont d'abord stockés dans la pile puis l'opération correspondante est effectuée.

Avant les calculs avec des fractions, il est nécessaire d'initialiser la pile de nombres à l'aide de l'opération **A'**. L'opération fait également passer l'affichage en mode deux lignes, où le numérateur de la fraction (registre T) et le dénominateur de la fraction (registre X) sont affichés sur la ligne supérieure de l'écran. Pour repasser l'affichage en mode standard, il est possible d'utiliser la fonction **B'** ou l'opération **Op 1D** ou le programme

## Pgm 01 SBR CE.

### Lbl A Ajouter une fraction dans la pile

Pour ajouter une fraction avec l'opération **A** il faut entrer le numérateur de la fraction, appuyer sur **x<>t**, entrer le dénominateur de la fraction et enfin appuyer sur **A**. La fraction saisie est stockée sur le dessus de la pile.

### Lbl A' Initialisation de la pile

La fonction **A'** permet d'initialiser la pile des fractions dans les registres R10 à R29. Vous pouvez également utiliser l'initialisation à plusieurs reprises si vous souhaitez démarrer un nouveau calcul. Lors de l'initialisation, l'écran passe en mode deux lignes. La ligne supérieure affiche le numérateur de la fraction (registre T) et la ligne inférieure affiche le dénominateur de la fraction (registre X). Le mode d'affichage sur deux lignes peut être terminé par l'opération **Op 1D** ou en appelant le sous-programme **Pgm 01 SBR CE**.

### Lbl B Conversion de décimales en fractions

L'opération **B** convertit le nombre décimal dans le registre X (affichage) en fraction a/b. L'opération B fait passer automatiquement l'affichage en mode deux lignes (registres T et X affichés simultanément). L'opération s'applique sur le nombre courant affiché, pas sur le nombre présent en haut de la pile. Il faut utiliser la fonction **A** pour ajouter le résultat de la conversion à la pile.

### Lbl B' Convertir une fraction en un nombre décimal

L'opération **B'** divise le numérateur de la fraction (dans le registre T) par le dénominateur (dans le registre X) et renvoie le résultat dans X sous forme de nombre décimal. L'opération B' fait passer automatiquement l'affichage en mode monoligne (le registre X et les drapeaux sont affichés). L'opération fonctionne avec les données affichées, pas avec les données de la pile.

### **Lbl** **C** Somme de fractions $X+Y$

L'opération **C** ajoute le dernier nombre fractionnaire  $Y$  présent sur le dessus de la pile à l'avant-dernier nombre  $X$ . L'opération **C** annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X+Y$  et deviendra donc le dernier nombre en haut de la pile.

### **Lbl** **C'** Différence de fractions $X-Y$

L'opération **C'** soustrait le dernier nombre fractionnaire  $Y$  en haut de la pile de l'avant-dernier nombre  $X$ . L'opération **C'** annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X-Y$  et deviendra donc le dernier nombre en haut de la pile.

### **Lbl** **D** Multiplication de fractions $X*Y$

L'opération **D** multiplie l'avant-dernier nombre fractionnaire  $X$  par le dernier nombre  $Y$  en haut de la pile. L'opération **D** annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X*Y$  et deviendra donc le dernier nombre en haut de la pile.

### **Lbl** **D'** Division de fractions $X:Y$

L'opération **D'** divise l'avant-dernier nombre fractionnaire  $X$  par le dernier nombre  $Y$  en haut de la pile. L'opération **D'** annule le dernier nombre en haut de la pile. L'avant-dernier nombre contiendra le résultat  $X:Y$  et deviendra donc le dernier nombre en haut de la pile.

### **Lbl** **E** Permutation des fractions

L'opération **E** échange le nombre  $Y$  en haut de la pile avec l'avant-dernier nombre  $X$ .

### **Lbl** **E'** Supprimer une fraction

L'opération **E'** annule le nombre  $X$  en haut de la pile.

## **Lbl** **+/-** Négation X

L'opération **SBR** **+/-** calcule la négation du nombre X en haut de la pile. Le résultat de l'opération remplacera la valeur originale du nombre X. En même temps, le résultat de l'opération est affiché à l'écran - le numérateur de la fraction sera sur la ligne supérieure (registre T), le dénominateur de la fraction sera sur la ligne du bas (registre X).

## **Lbl** **1/x** Inverse 1/X

L'opération **SBR** **1/x** calcule l'inverse du nombre X en haut de la pile. Le résultat de l'opération remplacera la valeur originale du nombre X. En même temps, le résultat de l'opération est affiché à l'écran - le numérateur de la fraction sera sur la ligne supérieure (registre T), le dénominateur de la fraction sera sur la ligne du bas (registre X).

Exemple,  $23/6 + 1/3$  :

**Pgm** **26** ... activation du programme de bibliothèque ML-26

**A'** ... initialisation de la pile

**2** **3** **x<>t** **6** **A** ... saisie première fraction,  $23/6$

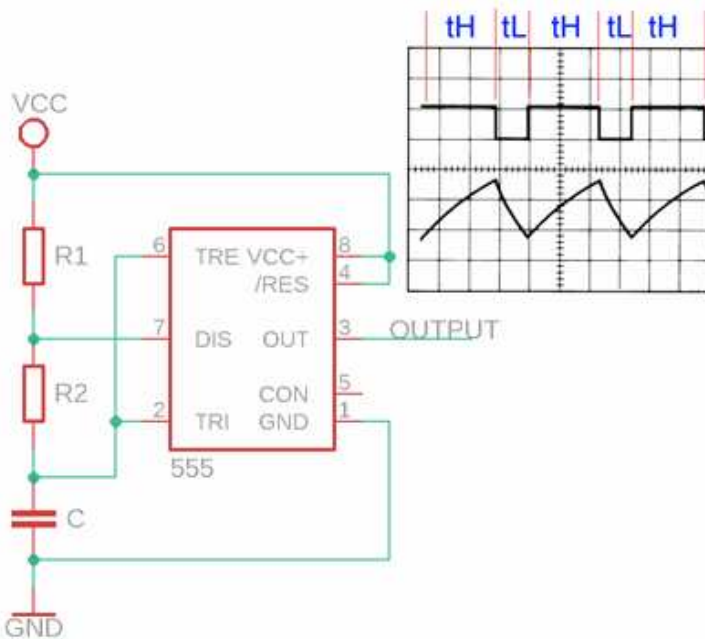
**1** **x<>t** **3** **A** ... saisie deuxième fraction,  $1/3$

**C** ... somme des fractions, résultat =  $25/6$

## ML-27 Générateur astable avec circuit 555

>C	>R1	>R2	>f	>tL
C uF	R1 kO	R2 kO	f Hz	>tH ms

Le programme ML-27 est utilisé pour calculer les données pour un circuit 555 connecté en générateur astable.



temps de charge (sortie à l'état HIGH)  $t_H = \ln(2) * (R1 + R2) * C$

temps de décharge (sortie à l'état LOW)  $t_L = \ln(2) * R2 * C$

période  $T = t_H + t_L = \ln(2) * (R1 + 2*R2) * C$

fréquence  $f = 1/T$

alternance de signaux  $D = t_H / T = (R1 + R2) / (R1 + 2*R2) = t_H * f$

Le programme calcule avec 4 données principales : le condensateur C (en uF), les résistances R1 et R2 (en kohms) et la fréquence f (en Hz). Lors des calculs, vous devez connaître 3 données à partir desquelles le programme dérive la 4ème donnée.

### **Lbl A** Saisie de la capacité C

L'opération **A** permet de saisir la capacité du condensateur C en uF.

### **Lbl A'** Calcul de la capacité C

L'opération **A'** calcule la capacité du condensateur C en uF à partir des paramètres R1, R2 et f.

### **Lbl B** Saisie de la résistance R1

L'opération **B** permet de saisir la valeur de la résistance R1 en kohms.

### **Lbl B'** Calcul de la résistance R1

L'opération **B'** calcule la valeur de la résistance R1 en ohms à partir des paramètres C, R2 et f.

### **Lbl C** Saisie de la résistance R2

L'opération **C** permet de saisir la valeur de la résistance R2 en kohms.

### **Lbl C'** Calcul de la résistance R2

L'opération **C'** calcule la valeur de la résistance R2 en ohms à partir des paramètres C, R1 et f.

### **Lbl D** Saisie de la fréquence f

L'opération **D** permet de saisir la fréquence du signal f en Hz.

## **Lbl D'** Calcul de la fréquence f

L'opération **D'** calcule la fréquence du signal f en Hz.

## **Lbl E** Calcul du temps tH

L'opération **E** calcule le temps de charge tH (le signal est à l'état HIGH) en ms. Les valeurs de C, R1 et R2 doivent être connues avant le calcul.

## **Lbl E'** Calcul du temps tL

L'opération **E'** calcule le temps de décharge tL (le signal est à l'état LOW) en ms. Les valeurs de C et R2 doivent être connues avant le calcul.

Exemple:

**Pgm 27** ... activation du programme de bibliothèque ML-27

**1 0 A** ... capacité du condensateur C = 10 uF

**1 B** ... valeur résistance R1 = 1 kohm

**2 C** ... valeur résistance R2 = 2 kohm

**D'** [28.85...] ... calcul fréquence f = 28.85... Hz

**1/x** [0.03465...] ... calcul période T = 34.65... ms

**E** [20.79...] ... calcul temps tH = 20.79... ms

**E'** [13.86...] ... calcul temps tL = 13.86... ms

**E + E' =** [34.65...] ... pour vérification, tH + tL = T

**D' \* E / 100 =** [60] ... alternance de signaux D = tH \* f / 1000 \* 100 = 60%



## ML-28 (EE-07) Conversion de ratios

->P2/P1	->V2/V1	->Np	->dB	
P2/P1	V2/V1	Np	dB	

Le programme ML-28 convertit les unités pour les ratios, les atténuations et les gains. En saisissant les données dans une unité, les données converties dans n'importe quelle autre unité peuvent être lues.

**Lbl A** Saisie du ratio de puissance P2/P1

**Lbl A'** Calcul du ratio de puissance P2/P1

**Lbl B** Saisie du voltage V2/V1

**Lbl B'** Calcul du voltage V2/V1

**Lbl C** Saisie de la valeur en népers Np

**Lbl C'** Calcul de la valeur en népers Np

**Lbl D** Saisie des décibels dB

**Lbl D'** Calcul des décibels dB

Exemple:

**Pgm 28** ... activation du programme de bibliothèque ML-28

**2 0 A** ... saisie du ratio de puissance P2/P1 = 20

**B'** [4.472...] ... calcul du voltage V2/V1 = 4.472...

**C'** [1.4978...] ... calcul de la valeur en népers Np = 1.4978...

**D'** [13.010...] ... calcul des décibels dB = 13.010...

## ML-29 (EE-11) Diagramme de réactance

->f	->L	->C	XL->L	XC->C
f	L	C	->XL	->XC

Le programme ML-29 est utilisé pour calculer les réactances et résonances des circuits LC.

**Lbl** **A** Saisie de la fréquence f en Hz

**Lbl** **A'** Calcul de la fréquence de résonance f (L et C doivent être connus)

**Lbl** **B** Saisie inductance L en Henrys

**Lbl** **B'** Calcul inductance L en Henrys (f et C doivent être connus)

**Lbl** **C** Saisie capacité C en farads

**Lbl** **C'** Calcul capacité C en farads (f et L doivent être connus)

**Lbl** **D** Calcul Réactance inductive XL en ohms (f et L doivent être connus)

**Lbl** **D'** Conversion de la réactance inductive XL en inductance L (f doit être connu)

**Lbl** **E** Calcul réactance capacitive XC en ohms (f et C doivent être connus)

**Lbl** **E'** Conversion de la réactance capacitive XC en capacité C (f doit être connu)

Exemple:

**Pgm** **29** ... activation du programme de bibliothèque ML-29

**Eng** ... notation ingénieur

**1** **5** **EE** **1** **2** **+/-** **C** ... Saisie capacité 15 pF

**2 7 EE 6 A** ... Saisie fréquence 27 MHz

**E** [392.97...] ... Calcul réactance capacitive  $X_C = 392.97...$  ohms

**1 0 EE 6 +/- B** ... Saisie inductance 10 uH

**D** [1.6964...+3] ... Calcul réactance inductive  $X_L = 1.6964...$  Kohms

**C'** [3.4746...-12] ... pour  $f=27$  MHz et  $L=10$  uH, capacité  $C = 3.4746...$  pF

**E** [1.6964...+3] ... contrôle calcul réactance capacitive

$X_C = 1.6964...$  Kohms. La condition de résonance est  $X_C = X_L$

## ML-30 (EE-12) Conversion d'impédance série / parallèle

->Rs	->Xs	->Rp	->Xp	
Rs	Xs	Rp	Xp	

Le programme ML-30 est utilisé pour calculer les impédances série et parallèle. Après avoir saisi la résistance et la réactance en série ( $R_s$  et  $X_s$  connectés en série), la résistance et la réactance parallèles de substitution ( $R_p$  et  $X_p$  connectés en parallèle) peuvent être calculées. Il en va de même pour la conversion inversée. Toutes les données sont en ohms.

**Lbl A** Saisie résistance en série  $R_s$ .

**Lbl B** Saisie réactance série  $X_s$

**Lbl C** Saisie résistance parallèle  $R_p$

**Lbl D** Saisie réactance parallèle  $X_p$

**Lbl A'** Calcul résistance en série  $R_s$

**Lbl B'** Calcul réactance série  $X_s$

**Lbl C'** Calcul résistance parallèle  $R_p$

**Lbl D'** Calcul réactance parallèle  $X_p$

Exemple:

L'impédancemètre RX a mesuré une résistance parallèle de 75 ohms et une capacité de 25 pF à 125 MHz. Nous devons calculer la connexion en série alternative.

**Pgm 29** ... activation du programme de bibliothèque ML-29

**Eng** ... notation ingénieur

**1 2 5 EE 6 A** ... Saisie fréquence  $f = 125$  MHz

**2 5 EE 1 2 +/- C** ... Saisie capacité  $C = 25$  pF

**E** [50.929...] ... Calcul réactance  $X_C = 50.929...$  ohm $\hat{u}$

**Pgm 30** ... activation du programme de bibliothèque ML-30

[50.929...] **D** ... Saisie réactance parallèle  $X_p$  ( $X_C$  calculé)

**7 5 C** ... Saisie résistance parallèle  $R_p = 75$  ohm $\hat{u}$

**A'** [23.856...] ... Calcul résistance en série  $R_s = 23.856...$  ohm $\hat{u}$

**B'** [34.856...] ... Calcul réactance série  $X_s = 34.856...$  ohm $\hat{u}$

**Pgm 29** ... activation du programme de bibliothèque ML-29

**x<>t** ... résultat  $X_s$  mis en réserve (registre T)

**1 2 5 EE 6 A** ... Saisie fréquence  $f = 125$  MHz

**x<>t E'** [36.528...-12] ... conversion de  $X_C$  en capacité  $C = 36.528...$  pF

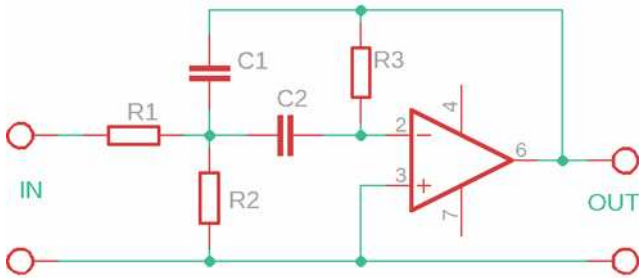
La connexion série de remplacement pour 125 MHz a une résistance de 23,856... ohms et une capacité de 36,528... pF.

## ML-31 (EE-13) Filtres actifs

C1	C2	->BP	->LP	->HP
alpha	A	F	B	

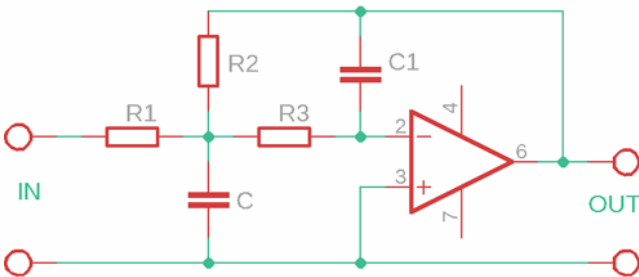
Le programme ML-31 est utilisé pour calculer les valeurs des composants nécessaires à utiliser dans la conception des filtres passe-bas (LP), passe-haut (HP) et passe-bande (BP) actifs.

Passe-bande actif (Bandpass BP)



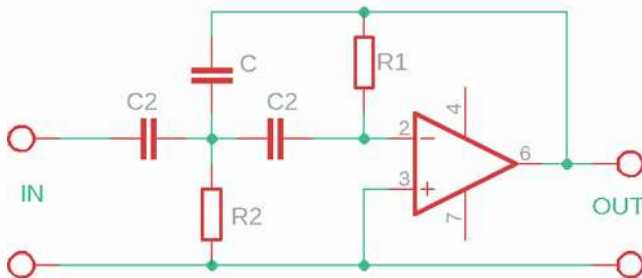
Données connues : bande passante de 3 dB de la bande "B" en Hz, gain moyenne bande "A" en dB, fréquence moyenne bande "F" en Hz, capacités C1 et C2 en farads. Le programme calcule les valeurs de R1, R2 et R3.

Passe-bas actif (Lowpass LP)



Données connues : facteur de crête 'alpha', gain 'A' en dB, fréquence de coupure 'F' en Hz, C1 en farads. Le programme calcule C, R1, R2 et R3.

## Passé-haut actif (Highpass HP)



Données connues : facteur de crête « alpha », gain « A » en dB, fréquence de coupure « F » en Hz, C2 en farads. Le programme calcule C, R1 et R2.

**Lbl A** Saisie facteur de crête « alpha » (LP, HP)

Le facteur de crête indique la rondeur du bord de rupture du filtre. Le facteur  $\alpha = 1$  représente la courbe standard. Pour  $\alpha > 1$ , le bord de cassure est arrondi. Pour  $\alpha < 1$ , un « pic » net se forme au bord de la cassure, les fréquences au point de cassure de la courbe seront accentuées.

**Lbl B** Saisie gain 'A' en dB (BP, LP, HP)

**Lbl C** Saisie fréquence 'F' v Hz (BP, LP, HP)

**Lbl D** Saisie bande passante de 3 dB 'B' v Hz (BP)

**Lbl A'** Saisie capacité C1 en farads (BP, LP)

**Lbl B'** Saisie capacité C2 en farads (BP, HP)

**Lbl C'** Calcul filtres passe-bande

Saisir 'B', 'A', 'F', 'C1' et 'C2' avant de calculer. Après avoir appuyé sur **C'**, le programme affiche les valeurs 'R1', 'R2' et 'R3'. Appuyer sur **R/S** pour afficher la valeur suivante.

## **Lbl** **D'** Calcul passe-bas

Saisir 'alpha', 'A', 'F' et 'C1' avant de calculer. Après avoir appuyé sur **D'**, le programme affiche les valeurs de 'C', 'R1', 'R2' et 'R3'. Appuyer sur **R/S** pour afficher la valeur suivante.

## **Lbl** **E'** Calcul passe-haut

Saisir 'alpha', 'A', 'F' et 'C2' avant de calculer. Après avoir appuyé sur **E'**, le programme affiche C, R1 et R2. Appuyer sur **R/S** pour afficher la valeur suivante.

### Exemple BP:

**Pgm** **31** ... activation du programme de bibliothèque ML-31

**Eng** ... notation ingénieur

**1** **6** **D'** ... Saisie bande passante de 3 dB  $B = 16$  Hz

**3** **0** **B** ... Saisie amplification  $A = 30$  dB

**1** **5** **0** **C** ... Saisie fréquence centrale  $F = 150$  Hz

**1** **0** **0** **EE** **9** **+/-** **A'** **B'** ... Saisie  $C1$  et  $C2 = 100$  nF

**C'** [3.145...+3] ... Calcul  $R1 = 3.145...$  kohm

**R/S** [690.0...] ... Calcul  $R2 = 690.0...$  ohm

**R/S** [198.9...+3] ... Calcul  $R3 = 198.9...$  kohm

### Exemple LP:

**Pgm** **31** ... activation du programme de bibliothèque ML-31

**Eng** ... notation ingénieur

**1** **|** **4** **1** **4** **2** **A** ... Saisie facteur de crête  $\alpha = 1.4142$  ( $=\sqrt{2}$ )

**2** **0** **B** ... Saisie amplification  $A = 20$  dB

**1** **EE** **3** **C** ... Saisie fréquence de coupure  $F = 1$  kHz

**2 0 EE 9 +/- A'** ... Saisie C1 = 20 nF

**D'** [440-9] ... Calcul C = 440 nF

**R/S** [562.69...] ... Calcul R1 = 562.69... ohm

**R/S** [5.626...+3] ... Calcul R2 = 5.626... kohm

**R/S** [511.5...] ... Calcul R3 = 511.4... ohm

Exemple HP:

**Pgm 31** ... activation du programme de bibliothèque ML-31

**Eng** ... notation ingénieur

**. 5 A** ... Saisie facteur de crête alpha = 0.5

**6 B** ... Saisie amplification A = 6 dB

**4 0 0 C** ... Saisie fréquence de coupure F = 400 Hz

**4 7 EE 9 +/- B'** ... Saisie C2 = 47 nF

**E'** [23.55-9] ... Calcul C = 23.55... nF

**R/S** [84.496...+3] ... Calcul R1 = 84.496... kohm

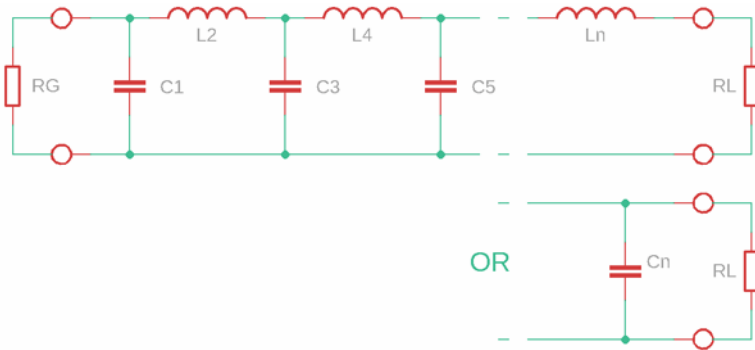
**R/S** [1.692...+3] ... Calcul R2 = 1.692... kohm



# ML-32 (EE-14) Filtres passifs

n	eps	R	fc	->calc

Le programme ML-32 calcule les valeurs des composants pour les filtres passe-bas passifs Butterworth et Tchebycheff.



Le filtre passe-bas passif est constitué de condensateurs alternatifs connectés en parallèle et de bobines connectées en série, numérotées de 1 à n. Pour un « n » pair, le filtre se termine par une bobine, pour un « n » impair, il se termine par un condensateur. La résistance  $R_G$  représente la résistance interne du générateur de signal. La résistance  $R_L$  est la résistance d'entrée de la charge. Le programme suppose que les valeurs des deux résistances correspondent,  $R_G = R_L = R$ .

**Lbl A** Saisie ordre du filtre 'n'

**Lbl B** Saisie ondulation eps en dB

L'ondulation eps est l'ondulation autorisée de la forme d'onde du filtre dans les bandes inférieures (passantes), valable pour le filtre Chebyshev. Dans le cas d'un filtre Butterworth, saisir 0.

**Lbl C** Saisie résistance de terminaison R en ohms

**Lbl D** Saisie fréquence de coupure fc en Hz

## **Lbl** **E** Calcul paramètres de filtre

Le programme affiche alternativement les valeurs Ck et Lk. Appuyer sur **R/S** pour afficher la valeur suivante.

### Exemple 1, Filtre Butterworth d'ordre 9 :

**Pgm** **32** ... activation du programme de bibliothèque ML-32

**Eng** ... notation ingénieur

**9** **A** ... Saisie ordre du filtre  $n = 9$

**0** **B** ... Saisie ondulation  $\epsilon = 0$  (filtre Butterworth)

**2** **EE** **3** **C** ... Saisie résistance de terminaison  $R = 2 \text{ kohm}$

**1** **0** **EE** **3** **D** ... Saisie fréquence de coupure  $f_c = 10 \text{ kHz}$

**E** [2.763...-9] ... Calcul  $C_1 = 2.763... \text{ nF}$

**R/S** [31.83...-3] ... Calcul  $L_2 = 31.83... \text{ mH}$

**R/S** [12.19...-9] ... Calcul  $C_3 = 12.19... \text{ nF}$

**R/S** [59.82...-3] ... Calcul  $L_4 = 59.82... \text{ mH}$

**R/S** [15.91...-9] ... Calcul  $C_5 = 15.91... \text{ nF}$

**R/S** [59.82...-3] ... Calcul  $L_6 = 59.82... \text{ mH}$

**R/S** [12.19...-9] ... Calcul  $C_7 = 12.19... \text{ nF}$

**R/S** [31.83...-3] ... Calcul  $L_8 = 31.83... \text{ mH}$

**R/S** [2.763...-9] ... Calcul  $C_9 = 2.763... \text{ nF}$

### Exemple 2, Chebyshevův filtr 7. řádu:

**Pgm** **32** ... activation du programme de bibliothèque ML-32

**Eng** ... notation ingénieur

**7** **A** ... Saisie ordre du filtre  $n = 7$

**5** **B** ... Saisie ondulation  $\epsilon = 0.5 \text{ dB}$  (filtre Chebycheff)

**1** **EE** **3** **C** ... Saisie résistance de terminaison  $R = 1 \text{ kohm}$

**3** **|** **5** **EE** **3** **D** ... Saisie fréquence de coupure  $f_c = 3.5 \text{ kHz}$

**E** [78.99...-9] ... Calcul  $C1 = 78.99... \text{ nF}$

**R/S** [57.21...-3] ... Calcul  $L2 = 57.21... \text{ mH}$

**R/S** [119.9...-9] ... Calcul  $C3 = 119.9... \text{ nF}$

**R/S** [61.13...-3] ... Calcul  $L4 = 61.13... \text{ mH}$

**R/S** [119.9...-9] ... Calcul  $C5 = 119.9... \text{ nF}$

**R/S** [57.21...-3] ... Calcul  $L6 = 57.21... \text{ mH}$

**R/S** [78.99...-9] ... Calcul  $C7 = 78.99... \text{ nF}$

## ML-33 (EE-15) Convolution du signal

n0	dt	->y(t)		

La convolution est la réponse d'un système linéaire à un signal d'entrée. Dans le programme utilisateur principal, il faut d'abord créer une fonction notée Lbl A', qui renvoie le déroulement du signal d'entrée en fonction du temps  $x(t)$ . De plus, il est nécessaire de créer une fonction notée Lbl B', qui renvoie l'évolution du signal de sortie du système en fonction du temps  $h(t)$ , en réaction au signal impulsionnel d'entrée. Les deux fonctions ne doivent pas utiliser la touche **=** ni la touche **CLR**. Le programme calcule le signal de sortie  $y(t)$ .

**Lbl** **A** Saisie du nombre de sections  $n0$  dans chaque incrément de temps  $dt$

**Lbl** **B** Saisie incrément de temps  $dt$

**Lbl** **C** Calcul valeurs  $y(t)$

L'opération affiche l'heure actuelle 't' sur la ligne supérieure de l'écran et la valeur du signal de sortie  $y(t)$  sur la ligne inférieure. Le calcul s'effectue à partir de l'instant  $0+dt$ , à l'instant  $0$   $y(0) = 0$ . Valeur suivante avec **R/S**.

Retour au mode d'affichage normal avec **Op 1D**.

Exemple:

Le signal d'entrée a pour cours  $x(t) = 2^*t$  pour  $t \leq 0,3$  sinon 0. La réponse à l'impulsion a la forme  $h(t) = 10^*exp(-5^*t)$ .

**RST LRN** ... activation du mode programmation

**Lbl A'** ... l'étiquette du début de la fonction du signal d'entrée  $x(t)$

**( [ CE x 2 ] x<>t** ... valeur  $2^*t$  dans registre T

**[ 6 x>=t Nop** ... si  $2^*t < 0,6$ , aller à l'étiquette Nop

**CP** ... pour  $t > 0,3$  réinitialiser le registre T

**Lbl Nop** ... étiquettes pour le cas  $t \leq 0,3$

**x<>t** ... návrat registru T na displej

**RTN** ... fin de fonction **A'** (**INV SBR**)

**Lbl B'** ... étiquette de début de fonction de réponse  $h(t)$

**( [ +/- x 5 ]** ... valeur  $-5^*t$

**INV Inx x 1 0 ]** ... valeur  $10^*exp(-5^*t)$

**RTN** ... fin de fonction **B'** (= **INV SBR**)

**LRN** ... sortie du mode programmation

**Pgm 33** ... activation du programme de bibliothèque ML-33

**4 A** ... Saisie du nombre de sections pour chaque temps  $dt$ ,  $n_0 = 4$

**[ 1 B** ... Saisie incrément de temps  $dt = 0.1$

**C** [0.0861...] ... Calcul  $y(0.1) = 0.0861...$

**R/S** [0.2960...] ... Calcul  $y(0.2) = 0.2960...$

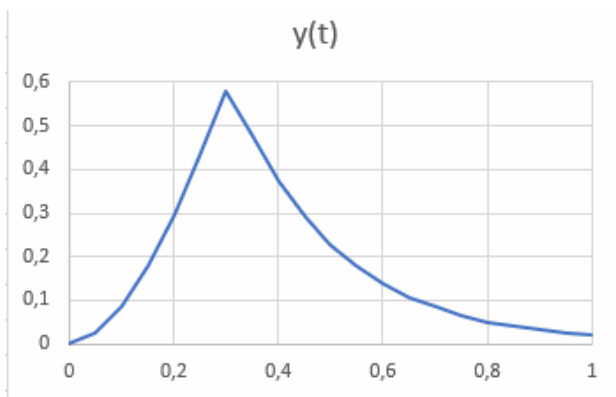
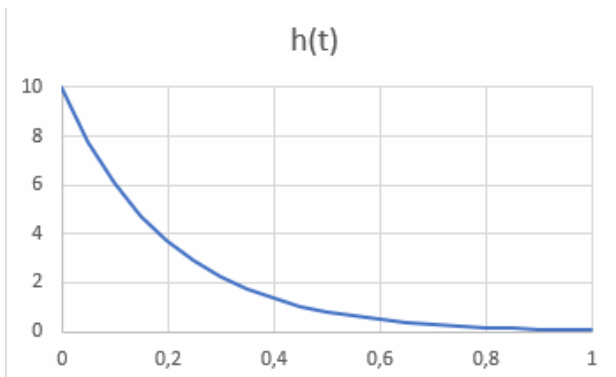
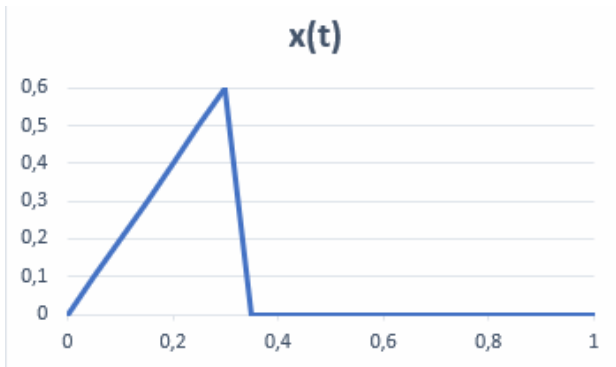
**R/S** [0.5808...] ... Calcul  $y(0.3) = 0.5808...$

**R/S** [0.3978...] ... Calcul  $y(0.4) = 0.3978...$

**R/S** [0.2412...] ... Calcul  $y(0.5) = 0.2412...$

**R/S** [0.1463...] ... Calcul  $y(0.6) = 0.1463...$

**R/S** [0.0887...] ... Calcul  $y(0.7) = 0.0887...$



# ML-34 (EE-17) Transformation de Fourier discrète

N	n,f(n)	->DFT	n,F(n)	->IDFT

Le programme ML-34 calcule la transformation de Fourier discrète. Il convertit les échantillons de signaux en temps réel en un spectre de fréquences (DFT, Discret Fourier Transform) ainsi que la conversion inverse du spectre en évolution temporelle (IDFT, Inverse Discret Fourier Transform). Le programme prend en charge jusqu'à 100 échantillons lors de la conversion DFT et 50 échantillons lors de la conversion IDFT..

## Lbl A Saisie nombre d'échantillons N

Pour DFT maximum 100 échantillons, pour IDFT maximum 50 échantillons.

## Lbl B Saisie échantillons de temps f(n)

Saisir l'index de la première entrée n (= 0...N-1), appuyer sur **B** puis saisir les données temporelles f(n) séquentiellement en validant chaque saisie avec **R/S** pour passer à la suivante. Les échantillons de temps f(n) peuvent aussi être directement chargés dans les registres R00 à R99 : le numéro de registre correspondant à l'index d'entrée : n=0 f(n)= **STO 00**, n=1 f(n)= **STO 01**, ... , n=99 f(n) = **STO 99**.

## Lbl C Calcul DFT

En appuyant sur **C**, les échantillons de temps sont convertis en échantillons de fréquence. Les échantillons de fréquence sont affichés par paires de valeurs : l'amplitude amp(n) est affichée, valeur suivante avec R/S, la phase(n) est affichée, valeur suivante avec **R/S**.

## Lbl D Saisie échantillons de fréquence F(n)

Saisir l'index du premier élément n (= 0...N-1), appuyer sur **D** et saisir les paires d'échantillons - saisir l'amplitude amp(n), valider la saisie avec **R/S** pour passer à la saisie de la phase (n), valider avec **R/S** pour passer à la

saisie de la paire suivante. Les échantillons de fréquence sont stockés en mémoire convertis en partie réelle et imaginaire du nombre. Les registres R00 à R49 stockent les parties réelles des nombres, les registres R50 à R99 stockent les parties imaginaires des nombres. Les valeurs peuvent donc aussi être directement chargées dans les registres R00 à R49 et dans les registres R50 à R99.

## **Lbl** **E** Calcul IDFT

Appuyer sur **E** convertit les échantillons de fréquence en échantillons de temps. Afficher chaque valeur suivante avec **R/S**.

### Exemple:

Le spectre de fréquence contient 32 échantillons (= N). L'échantillon  $n = 2$  a une valeur  $(0 - 16i)$ , l'échantillon  $n = 30$  a une valeur  $(0 + 16i)$ , les autres échantillons ont une valeur 0. Les échantillons seront écrits directement dans la mémoire au format de nombres complexes. Par transformation IDFT inverse, nous devrions obtenir le signal original, qui avait la forme  $S = \sin(n \cdot \pi/8)$ , où  $n = 0 \dots N-1$ . Le signal est une onde sinusoïdale avec une fréquence de  $1/8$ .

**Pgm** **34** ... activation du programme de bibliothèque ML-34

**CMS** ... initialiser tous les registres de données

**3** **2** **A** ... Saisie nombre d'échantillons  $N = 32$

**1** **6** **STO** **80** ... insertion de la composante imaginaire de l'échantillon  $(0 + 16i)$  d'indice 30

**+/-** **STO** **52** ... insertion de la composante imaginaire de l'échantillon  $(0 - 16i)$  d'indice 2

**Fix** **4** ... affichage jusqu'à 4 décimales

**E** ... effectuer la conversion en échantillons de temps

[0] ...  $f(0) = 0$

**R/S** [0.3827] ...  $f(1) = 0.3827$ , attendu :  $\sin(1 \cdot \pi/8) = 0.3827$

**R/S** [0.7071] ...  $f(2) = 0.7071$ , attendu :  $\sin(2 \cdot \pi/8) = 0.7071$

**R/S** [0.9239] ...  $f(3) = 0.9239$ , attendu :  $\sin(3 \cdot \pi/8) = 0.9239$

**R/S** [1.0000] ...  $f(4) = 1$ , attendu :  $\sin(4 \cdot \pi/8) = 1$

**R/S** [0.9239] ...  $f(5) = 0.9239$ , attendu :  $\sin(5 \cdot \pi/8) = 0.9239$

**R/S** [0.7071] ...  $f(6) = 0.7071$ , attendu :  $\sin(6 \cdot \pi/8) = 0.7071$

**R/S** [0.3827] ...  $f(7) = 0.3827$ , attendu :  $\sin(7 \cdot \pi/8) = 0.3827$

**R/S** [0.0000] ...  $f(8) = 0$ , attendu :  $\sin(8 \cdot \pi/8) = 0$

**R/S** [-0.3827] ...  $f(9) = -0.3827$ , attendu :  $\sin(9 \cdot \pi/8) = -0.3827$

**R/S** [-0.7071] ...  $f(10) = -0.7071$ , attendu :  $\sin(10 \cdot \pi/8) = -0.7071$

**R/S** [-0.9239] ...  $f(11) = -0.9239$ , attendu :  $\sin(11 \cdot \pi/8) = -0.9239$

**R/S** [-1.0000] ...  $f(12) = -1$ , attendu :  $\sin(12 \cdot \pi/8) = -1$

**R/S** [-0.9239] ...  $f(13) = -0.9239$ , attendu :  $\sin(13 \cdot \pi/8) = -0.9239$

**R/S** [-0.7071] ...  $f(14) = -0.7071$ , attendu :  $\sin(14 \cdot \pi/8) = -0.7071$

**R/S** [-0.3827] ...  $f(15) = -0.3827$ , attendu :  $\sin(15 \cdot \pi/8) = -0.3827$

... pour  $f(16)$  à  $f(31)$  les échantillons sont répétés comme pour  $f(0)$  à  $f(15)$ .

## ML-35 Loi d'Ohm

->U	->I	->R	->Pui	Pu->R
U	I	R	Pr->U	Pr->I

Le programme ML-35 est utilisé pour calculer la tension, le courant et la résistance selon la loi d'Ohm. Après avoir saisi deux des quantités, la troisième quantité est calculée. De plus, la perte de puissance sur la résistance est calculée.

**Lbl** **A** Saisie tension U

**Lbl** **A'** Calcul tension U (de l'intensité I et de la résistance R)



**Lbl B** Saisie intensité I

**Lbl B'** Calcul intensité I (de la tension U et de la résistance R)

**Lbl C** Saisie résistance R

**Lbl C'** Calcul résistance R (de la tension U et de l'intensité I)

**Lbl D** Saisie perte de puissance P et calcul de la tension U sur la résistance R

**Lbl D'** Calcul perte de puissance P (de la tension U et de l'intensité I)

**Lbl E** Saisie perte de puissance P et calcul de l'intensité I par la résistance R

**Lbl E'** Saisie perte de puissance P et calcul de la résistance R avec la tension U

#### Exemple

**Pgm 35** ... activation du programme de bibliothèque ML-35

**Eng** ... notation ingénieur

**5 A** ... Saisie tension  $U = 5V$

**4 EE 3 +/- B** ... Saisie intensité  $I = 4 \text{ mA}$

**C'** [1.25+3] ... calcul résistance  $R = 1.25 \text{ kohms}$

**D'** [20-3] ... perte de puissance dans la résistance  $P = 20 \text{ mW}$

**1 E** [25] ... perte de puissance de 1, calcul résistance  $R = 25 \text{ ohms}$

**C** ... la résistance calculée  $R = 25 \text{ ohms}$  est reprisee

**B'** [200-3] ... une résistance de 25 ohms donnerait 200 mA à 5 V

# ML-36 Connections série et parallèle

RLC	+RLs,Cp	+RLp,Cs		

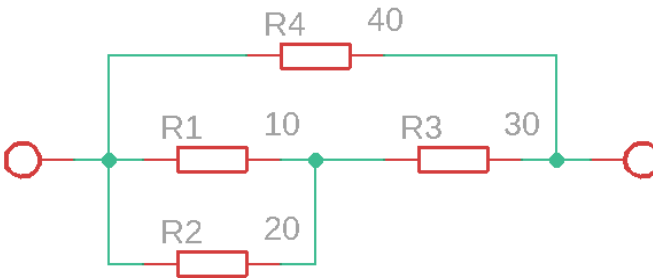
Le programme ML-36 calcule la valeur résultante de résistance, d'inductance ou de capacité, connectée en série ou en parallèle.

**Lbl A** Saisie valeurs de la première résistance, capacité, inductance

**Lbl B** Ajout d'une résistance ou d'une inductance série, d'une capacité parallèle

**Lbl C** Ajout d'une résistance ou d'une inductance parallèle, d'une capacité série

Exemple:



**Pgm 36** ... activation du programme de bibliothèque ML-36

**1 0 A** ... Saisie de la première résistance R1 = 10 ohms

**2 0 C** [6.666...] ... ajout R2 en parallèle = 20 ohms

**3 0 B** [36.666...] ... ajout R3 en série = 30 ohms

**4 0 C** [19.13...] ... ajout R4 en parallèle = 40 ohms

... La résistance qui en résulte est de 19.13... ohms.

## ML-37 (AV-23) Changement de fuseau horaire

				->date'
zone	date	time	dt	->time'

Le programme ML-37 calcule le changement de date et d'heure lors d'un vol vers un autre fuseau horaire. La date est au format MM.JJ (mois, jour), l'heure au format HH.MMSS (heures, minutes, secondes).

**Lbl A** Saisie fuseau horaire (ouest : -, est +)

Le fuseau horaire représente l'écart horaire par rapport à l'heure UTC. À l'ouest du premier méridien de Greenwich, la zone a un signe négatif, à l'est elle a un signe positif.

Hawaï .....	10	Pacifique .....	8
USA Centre .....	6	USA Est .....	5
Greenwich .....	0	Europe centrale .....	-1
Europe de l'Est .....	-2	Bagdad, Moscou, Naïrobi .	-3
Dubaï, Abou Dabi, Bakou .	-4	Chine .....	-8
Japon .....	-9	Est Asutralie .....	-10

Remarque : Le programme AV-23 original utilise des fuseaux horaires pour l'aviation, où le fuseau représente une correction pour convertir l'heure locale en heure UTC. Les fuseaux horaires ont donc le signe opposé.

**Lbl B** Saisie date MM.DD

**Lbl C** Saisie heure départ HH.MMSS

**Lbl D** Saisie temps (delta) à additionner HH.MMSS

La valeur maximale autorisée du delta est de +- 648 heures (soit 27 jours).

**Lbl E** Calcul heure arrivée HH.MMSS

**Lbl E'** Calcul date arrivée MM.DD

Le programme ne prend pas en compte les années bissextiles, février compte 28 jours. Lors du survol de la date du 29 février d'une année

bissextile, la date obtenue doit être corrigée d'un jour.

**Exemple 1:**

Le vol de Tuscon (zone -7) à New York (zone -5) dure 5 heures et 22 minutes. Le départ est le 31 décembre à 22h10.

**Pgm 37** ... activation du programme de bibliothèque ML-37

**7 +/- A** ... zone de départ = -7

**1 2 | 3 1 B** ... date de départ = 31 décembre

**2 2 | 1 0 C** ... heure de départ = 22:10

**5 | 2 2 D** ... temps de vol = 5 heures 22 minutes

**5 +/- A** ... zone d'arrivée = -5

**E** [5.32] ... Calcul heure d'arrivée = 5:32:00

**E'** [1.01] ... Calcul date d'arrivée = 1er janvier

**Exemple 2:**

Vous souhaitez voler de Honolulu, Hawaï (zone -10) à New Delhi, Inde (zone +5,5). La durée du vol sera de 35 heures 27 minutes avec escale. Le départ se fera le 19 septembre à 8h40.

**Pgm 37** ... activation du programme de bibliothèque ML-37

**1 0 +/- A** ... zone de départ = -10

**9 | 1 9 B** ... date de départ = 19 septembre

**8 | 4 0 C** ... heure de départ = 8:40

**3 5 | 2 7 D** ... temps de vol = 35 heures et 27 minutes (avec escale)

**5 | 5 A** ... zone d'arrivée = +5.5

**E** [11.37] ... Calcul heure d'arrivée = 11:37:00

**E'** [9.21] ... Calcul date d'arrivée = 21 septembre

Votre contact à New Delhi sera retardé de 6 jours. Rendez-vous à

l'aéroport le 27 septembre à 15h00. Si l'heure de vol reste la même, quand devez-vous décoller d'Honolulu ?

**5** | **5** | **A** ... zone d'arrivée = +5.5

**9** | **27** | **B** ... date d'arrivée = 27 septembre

**15** | **C** ... heure d'arrivée = 15:00

**35** | **27** | **+/-** | **D** ... temps de vol = -35 heures 27 minutes (décalage temporel vers l'arrière)

**10** | **+/-** | **A** ... zone de départ = -10

**E** [12.03] ... Calcul heure de départ = 12:03:00

**E** [9.25] ... Calcul date de départ = 25 septembre

### Exemple 3:

Si l'heure à Chicago (zone -6) est 21h15 et que la date est le 23 novembre, quelle est la date et l'heure à Prague (zone 1) ?

**Pgm** **37** ... activation du programme de bibliothèque ML-37

**6** | **+/-** | **A** ... première zone = -6

**11** | **23** | **B** ... première date = 23 novembre

**21** | **15** | **C** ... première heure = 21:15

**0** | **D** ... pas de delta horaire : heure constante

**1** | **A** ... deuxième zone = +1

**E** [4.1500] ... heure calculée = 4:15:00

**E** [11.24] ... date calculée = 24 novembre

## ML-38 (MU-06) Tri

			Geom	Mean
N	Enter	Sort	View	Median

Le programme ML-38 utilise la méthode de tri de données de Donald Shell (1924-2015) qui publia son algorithme en juillet 1959.

**Lbl A** Saisie nombre d'éléments N (max. 100)

**Lbl B** Saisie des données

Saisir l'index de départ puis appuyer sur **B**, puis saisir séquentiellement chaque valeur en validant avec **R/S**.

**Lbl C** Tri des données

**Lbl D** Affichage des données

Saisir l'index de départ puis appuyer sur **D**, les données s'affiche séquentiellement. Valeur suivante en appuyant sur **R/S**.

**Lbl D'** Calcul diamètre géométrique

**Lbl E** Calcul médiane (valeur moyenne des données)

**Lbl E'** Calcul moyenne arithmétique

Exemple:

**Pgm 38** ... activation du programme de bibliothèque ML-38

**1 0 A** ... Saisie nombre d'éléments N = 10

**0 B** ... saisie à partir de l'index 0

**1 0 | 6 R/S** ... Saisie donnée d0 = 10.6

**5** | **1** | **2** | **R/S** ... Saisie donnée d1 = 5.12

**1** | **1** | **R/S** ... Saisie donnée d2 = 11

**9** | **2** | **R/S** ... Saisie donnée d3 = 9.2

**4** | **3** | **+/-** | **R/S** ... Saisie donnée d4 = -4.3

**1** | **4** | **5** | **+/-** | **R/S** ... Saisie donnée d5 = -1.45

**4** | **R/S** ... Saisie donnée d6 = 0.4

**3** | **7** | **R/S** ... Saisie donnée d7 = 37

**1** | **R/S** ... Saisie donnée d8 = 0.1

**8** | **3** | **R/S** ... Saisie donnée d9 = 8.3

**C** ... tri des données

**0** | **D** ... affichage à partir de l'index 0

[-4.3] ... affichage donnée d0 = -4.3

**R/S** [-1.45] ... affichage donnée d1 = -1.45

**R/S** [0.1] ... affichage donnée d2 = 0.1

**R/S** [0.4] ... affichage donnée d3 = 0.4

**R/S** [5.12] ... affichage donnée d4 = 5.12

**R/S** [8.3] ... affichage donnée d5 = 8.3

**R/S** [9.2] ... affichage donnée d6 = 9.2

**R/S** [10.6] ... affichage donnée d7 = 10.6

**R/S** [11] ... affichage donnée d8 = 11

**R/S** [37] ... affichage donnée d9 = 37

**E** [8.3] ... affichage médianu = 8.3

**E** [7.597] ... affichage moyenne arithmétique = 7.597

**D** [3.6508...] ... affichage diamètre géométrique = 3.6508...

## ML-39 (MU-09) Décomposition en facteurs premiers

Prime				

Le programme ML-39 décompose un entier en facteurs premiers.

**Lbl** **A** Saisie nombre entier.

Saisir le nombre puis appuyer sur **A**. Le programme trouver le premier facteur premier. Appuyer sur **R/S** pour trouver le facteur premier suivant jusqu'à facteur premier = 1 (dernier facteur premier trouvé).

Exemple:

**Pgm** **39** ... activation du programme de bibliothèque ML-39

**9** **8** **7** **6** **5** **4** **3** **2** **1** **A** ... Saisie nombre à décomposer

[3] ... affichage 1er facteur

**R/S** [3] ... affichage 2ème facteur

**R/S** [17] ... affichage 3ème facteur

**R/S** [17] ... affichage 4ème facteur

**R/S** [379721] ... affichage 5ème facteur

**R/S** [1] ... dernier facteur

Rozklad čísla 987654321 = 3 \* 3 \* 17 \* 17 \* 379721 \* 1



## ML-40 (MU-21) Arithmétique avec des variables

->A	->B	->C	->D	->E
A	B	C	D	E

Le programme ML-40 permet des calculs faciles avec des variables.

**Lbl** **A** à **E** Rappel de la variable A à E

**Lbl** **A'** à **E'** Enregistrer les valeurs des variables A à E

Exemple:

Calcul de l'accélération  $a = 2 * d / t^2$  ( $d$  = distance,  $t$  = temps) et de la vitesse  $v = a * t$ .

**Pgm** **40** ... activation du programme de bibliothèque ML-40

**2** **5** **A'** ... stocke distance  $d = 25$  dans la variable A

**1** **7** **B'** ... stocke temps  $t = 1,7$  dans la variable B

**2** **x** **A** **:** **B** **x**<sup>2</sup> **=** [17.30...] ... Calcul l'accélération  $a = 2 * d / t^2$

**x** **B** **=** [29.41...] ... Calcul vitesse  $v = a * t$

**1** **5** **B'** ... stocke temps  $t = 1.5$  dans la variable B

**2** **x** **A** **:** **B** **x**<sup>2</sup> **=** [22.22...] ... Calcul l'accélération  $a = 2 * d / t^2$

**x** **B** **=** [33.33...] ... Calcul vitesse  $v = a * t$

**1** **3** **B'** ... stocke temps  $t = 1.3$  dans la variable B

**2** **x** **A** **:** **B** **x**<sup>2</sup> **=** [29.58...] ... Calcul l'accélération  $a = 2 * d / t^2$

**x** **B** **=** [38.46...] ... Calcul vitesse  $v = a * t$

## ML-41 (MU-14) Interpolation

N	Enter	x->f(x)		

Le programme ML-14 interpole les données saisies avec un polynôme d'ordre (N-1), en utilisant la méthode d'Aitken.

**Lbl A** Saisie nombre d'échantillons N (max.33)

**Lbl B** Saisie des données (x,y)

Saisir l'index de départ puis appuyer sur **B**, puis saisir séquentiellement chaque valeur : x(i) et valider avec **R/S** puis y(i) et valider avec **R/S**. Répéter jusqu'à i = N.

**Lbl C** Calcul valeurs interpolées x->f(x)

Saisir x et appuyer sur **C** pour calcul de la valeur interpolée puis continuer en appuyant sur **C** ou **R/S**.

Exemple:

**Pgm 41** ... activation du programme de bibliothèque ML-41

**5 A** ... saisie nombre d'échantillons N = 5

**0 B** ... saisie des données à partir de l'index n = 0


**6 R/S 2 2 5 7 R/S** ... saisie échantillon (x0, y0) = (0.6, 0.2257)

**7 R/S 2 5 8 0 R/S** ... saisie échantillon (x1, y1) = (0.7, 0.2580)

**9 R/S 3 1 5 9 R/S** ... saisie échantillon (x2, y2) = (0.9, 0.3159)

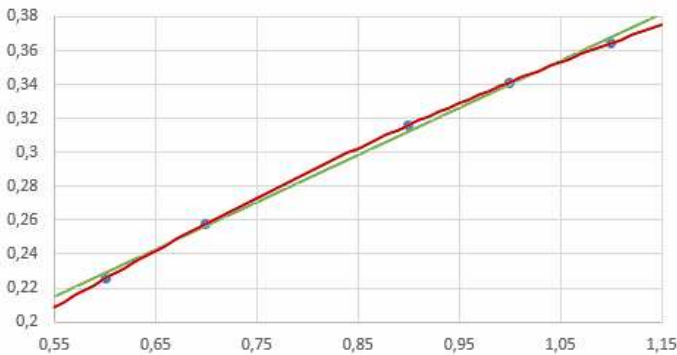
**1 R/S 3 4 1 3 R/S** ... saisie échantillon (x3, y3) = (1, 0.3413)

**1 1 R/S 3 6 4 3 R/S** ... saisie échantillon (x4, y4) = (1.1, 0.3643)

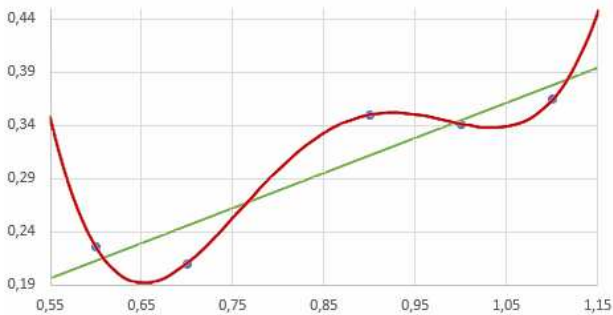
   [0.28811] ... interpolation  $f(0.8) = 0.28811$

Remarque : L'interpolation diffère de l'approximation en ce que lors de l'approximation les points sont entrelacés avec une courbe essayant d'imiter le parcours des données avec une courbe uniforme et lisse, par contre, l'interpolation observe strictement le passage de la courbe à travers les points de données, même au prix d'un tracé ondulé de la courbe.

Graphique d'interpolation de points selon l'exemple. La courbe d'interpolation (en rouge) a ici un très bon tracé. Pour comparer l'approximation de régression linéaire (en vert).



Si l'on change les deuxième et troisième points de l'exemple par les valeurs  $y=0,21$  et  $0,35$ , la courbe d'interpolation (en rouge) essaie toujours de maintenir le passage de la courbe par les points donnés et prend une forme plutôt ondulée. A titre de comparaison, approximation de régression linéaire à nouveau (en vert).



## ML-42 (MU-16) Minimax

Max	x->crit	next	f(x)	f'(x)

Le programme ML-42 recherche les minima et les maxima d'une fonction  $f(x)$ . Avant d'utiliser ce programme, il est nécessaire de créer la fonction  $f(x)$  dans un programme principal qui démarre avec **Lbl A** et qui convertit  $x$  en  $f(x)$ . Les fonctions ne doivent pas utiliser **H** ou **CLR**.

Lors de la recherche de points critiques, le programme divise l'intervalle depuis la valeur initiale de 'x' jusqu'à la valeur maximale de 'x' en 100 parties dans lesquelles il effectue la recherche. Si la dérivée de la fonction au début et à la fin de la partie change de signe, c'est un point critique et le programme trouvera l'endroit exact par la méthode de la bissectrice de l'intervalle.

**Lbl A** Saisie valeur 'x' maximale pour rechercher les points critiques

**Lbl B** Trouver le point critique « x » à partir du début donné « x »

Si le point critique est trouvé, le registre T contient le type de point : -1 maximum, +1 minimum. Si le point critique n'est pas trouvé, la valeur initiale 'x' clignote sur l'écran.

**Lbl C** Trouver le prochain point critique

Si un autre point critique est trouvé, le registre T contient le type de point : -1 maximum, +1 minimum. Si le point critique n'est pas trouvé, le dernier point critique trouvé 'x' clignote sur l'écran. Continuez la recherche du point suivant en appuyant sur **C** ou **R/S**.

**Lbl D** Calcul valeurs de fonction pour 'x' donné

**Lbl E** Calcul dérivée de la fonction pour 'x' donné

Avant de calculer la dérivée, il est nécessaire de saisir le maximum de 'x' en utilisant **A** et le minimum de 'x' en utilisant **B**, à partir desquels le

programme prépare l'épsilon 'x' pour le test de dérivée..

Exemple,  $f(x) = x^3 - x^2 - x + 2$ :

**RST** **LRN** ... activation du mode programmation

**Lbl** **A'** ... étiquette de début de fonction

**(** **STO** **10** **x^2** **x** **RCL** **10** ...  $x^3$

**-** **RCL** **10** **x^2** ...  $-x^2$

**-** **RCL** **10** **+** **2** **)** ...  $-x + 2$

**RTN** ... fin de fonction A' (**INV** **SBR**)

**LRN** ... sortie du mode programmation

**Pgm** **42** ... activation du programme de bibliothèque ML-42

**2** **A** ... Saisie maximum pour trouver le point critique = 2

**1** **+/-** **B** [-0.33333...] ... trouver le premier point critique  $x_1 = -0.33333...$

**D** [2.1851...] ... valeur de la fonction au point critique  $f(x) = 2.1851...$

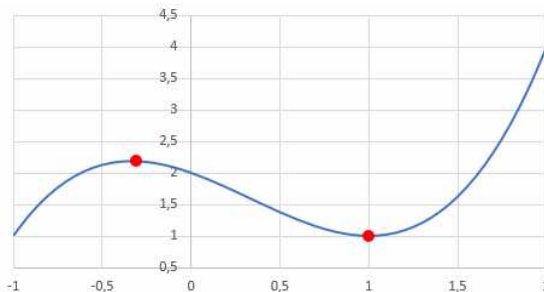
**x<>t** [-1] ... le point critique est le maximum

**C** [1] ... trouver le prochain point critique  $x_2 = 1$

**D** [1] ... valeur de la fonction au point critique  $f(x) = 1$

**x<>t** [1] ... le point critique est le minimum

**C** [1] clignotant ... il n'y a pas d'autre point critique



## ML-43 Somme de contrôle

CCITT	CCITTB	Dallas	XOR	INIT
CRC32	IBM	Modbus	Kermit	XModem

Le programme ML-43 calcule la somme de contrôle des données saisies à l'aide de diverses méthodes. Avant de calculer, appuyer d'abord sur la touche **E**, qui prépare le calcul d'une nouvelle série de données et fait passer la calculatrice en mode HEX. Saisir ensuite les octets de données (en code HEX) et choisir la méthode appropriée pour calculer la valeur de somme de contrôle suivante.

### **Lbl** **A** Calcul CRC-32 (32 bits)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-32 (32 bits). Octet suivant avec **R/S** ou **A**.

Le CRC-32-IEEE802.3 utilisé calcule à l'aide du polynôme  $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$  initialisé avec 0xFFFFFFFF et est utilisé en Ethernet et MPEG2.

### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> CBF43926

FC 05 4A -> A8E10F6D

### **Lbl** **B** Calcul CRC-IBM (16 bits)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-IBM (16 bits). Octet suivant avec **R/S** ou **B**.

Le CRC-IBM utilisé calcule le polynôme  $x^{16} + x^{15} + x^2 + 1$  initialisé à 0 et est utilisé dans les contrôleurs de disques et le bus Dallas Maxim 1-Wire.

### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> BB3D

FC 05 4A -> 9742

### **Lbl** **C** Calcul CRC-Modbus (16 bits)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-Modbus (16 bits). Octet suivant avec **R/S** ou **C**.

Le CRC-Modbus utilisé calcule le polynôme  $x^{16} + x^{15} + x^2 + 1$  avec initialisation 0xFFFF et est utilisé dans le protocole de communication Modbus.

#### **Échantillons de contrôle:**

"123456789" = 31 32 33 34 35 36 37 38 39 -> 4B37

FC 05 4A -> 5733

### **Lbl** **D** Calcul CRC-Kermit (16 bits)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-Kermit (16 bits). Octet suivant avec **R/S** ou **D**.

Le CRC-Kermit utilisé (le CRC-CCITT d'origine) calcule le polynôme  $x^{16} + x^{12} + x^5 + 1$  initialisé à 0 et est utilisé dans le protocole de communication Kermit.

#### **Échantillons de contrôle:**

"123456789" = 31 32 33 34 35 36 37 38 39 -> 8921

FC 05 4A -> 71BA

### **Lbl** **E** Calcul CRC-XModem (16 bits)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-XModem (16 bits). Octet suivant avec **R/S** ou **E**.

Le CRC-XModem utilisé calcule le polynôme  $x^{16} + x^{12} + x^5 + 1$  avec l'initialisation 0 et est utilisé dans le protocole de communication XModem. Cette méthode CRC est également utilisée par la calculatrice ET-58 pour

vérifier en interne l'intégrité de la mémoire ROM, car il existe une méthode de calcul simple et rapide, elle convient donc à une utilisation dans de petits appareils..

#### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> 31C3

FC 05 4A -> 8048

#### **Lbl** **A'** Calcul CRC-CCITT (16 bitů)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-CCITT (16 bits). Octet suivant avec **R/S** ou **A'**.

Le CRC-CCITT utilisé calcule le polynôme  $x^{16} + x^{12} + x^5 + 1$  avec initialisation 0xFFFF.

#### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> 29B1

FC 05 4A -> 4CD4

#### **Lbl** **B'** Calcul CRC-CCITT-B (16 bitů)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-CCITT-B (16 bits). Octet suivant avec **R/S** ou **B'**.

Le CRC-CCITT-B utilisé calcule le polynôme  $x^{16} + x^{12} + x^5 + 1$  avec initialisation 0x1D0F.

#### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> E5CC

FC 05 4A -> 9144

#### **Lbl** **C'** Calcul CRC-Dallas (8 bitů)



Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-Dallas (8 bits). Octet suivant avec **R/S** ou **C**.

Le CRC-Dallas utilisé calcule le polynôme  $x^8 + x^5 + x^4 + 1$  initialisé à 0 et est utilisé sur le bus Dallas Maxim 1-Wire.

#### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> A1

FC 05 4A -> F1

#### **Lbl D** Calcul CRC-XOR (16 bit)

Saisie octet en code HEX (00...FF) et calcul de la valeur par la méthode CRC-XOR (16 bits). Octet suivant avec **R/S** ou **C**.

Le CRC-XOR utilisé calcule une somme de contrôle avec une opération XOR de rotation gauche, initialisée à 0. Il est utilisé dans les petits appareils pour vérifier l'intégrité de la mémoire ROM, en raison d'une utilisation légère. Si l'instruction de rotation des bits n'est pas disponible, elle peut être remplacée par l'instruction ADD et l'ajout ultérieur du transfert de retenue..

#### Échantillons de contrôle:

"123456789" = 31 32 33 34 35 36 37 38 39 -> 406A

FC 05 4A -> 0760

#### Exemple:

**Pgm 43** ... activation du programme de bibliothèque ML-43

**E** ... initialiser une nouvelle séquence CRC

**1 2 A** ... Saisie 1er octet = 12 (en code HEX) avec calcul CRC-32

**3 4 R/S 5 6 R/S** ... Saisie 2ème et 3ème octets = 34 et 56 (HEX)

[D9C1A93A] ... résultat CRC-32 = D9C1A93A (HEX)

## ML-44 (LE-09) Jeu Mastermind

Guess				Start

Dans le jeu Mastermind, il faut deviner un nombre à 4 chiffres 1...9 (sans occurrences multiples de chiffres), la calculatrice affiche le nombre de chiffres devinés.

**Lbl** **A** Saisie nombre à 4 chiffres 1...9

Saisir un nombre à 4 chiffres et valider avec **A**. Le résultat est renvoyé sous la forme d'un nombre N.R : N représente le nombre de chiffres aux positions correctes et R le nombre de chiffres aux positions incorrectes. Si le résultat indique 4,0, le nombre a été correctement deviné.

**Lbl** **E** Nouveau jeu

*Remarque : Normalement, il n'est pas nécessaire d'initialiser le générateur de nombres aléatoires, mais pour éviter la reproductibilité des nombres aléatoires, il est possible d'initialiser le générateur nombres aléatoires par l'opération **Op** **52**.*

Exemple:

**Pgm** **44** ... activation du programme de bibliothèque ML-44

**2** **0** **7** **Op** **52** ... initialisation du générateur aléatoire (facultatif)

**1** **2** **3** **4** **A** [1.1] ... 1 chiffre bien placé, 1 chiffre mal placé

**5** **6** **7** **8** **A** [1.0] ... 1 chiffre bien placé

**1** **2** **3** **9** **A** [1,2] ... 1 chiffre bien placé, 2 chiffres mal placés

**9** **2** **3** **8** **A** [1.2] ... 1 chiffre bien placé, 2 chiffres mal placés

**2** **9** **3** **5** **A** [0.4] ... 4 chiffres mal positionnés

**5** **2** **9** **3** **A** [4.0] ... succès, le nombre est 5293

## ML-45 (LE-12) Jeu Acey-Deucy

	?Num3	?Bank		
Start	Num1	Num2	Odds	Bet

Dans le jeu ML-45 Acey-Deucy, il faut parier sur l'apparition, ou pas, d'un nombre dans un intervalle de deux nombres aléatoires Num1 et Num2 générés par le programme. Selon les chiffres, il vous offrira une chance de gagner (un multiple de la mise). Après avoir placé votre pari, le programme calcule le troisième numéro Num3. Si Num3 se situe dans l'intervalle Num1 à Num2, vous gagnez le pari multiplié par la cote. Si tu ne gagnes pas, tu perds le pari.

**Lbl A** Début nouveau jeu de (crédit = 1000)

**Lbl B** Génération du premier nombre Num1 = 1...1000

**Lbl C** Génération du deuxième nombre Num2 = 1...1000

**Lbl D** Calcul cote (multiple de pari)

**Lbl E** Saisie montant du pari et résultat

**Lbl B'** Affichage du troisième nombre Num3 = 1...1000

**Lbl C'** Affichage solde crédit

*Remarque : Normalement, il n'est pas nécessaire d'initialiser le générateur de nombres aléatoires, mais pour éviter la reproductibilité des nombres aléatoires, il est possible d'initialiser le générateur nombres aléatoires par l'opération **Op 52**.*

Exemple:

**Pgm 45** ... activation du programme de bibliothèque ML-45

**0 Op 52** ... initialisation du générateur aléatoire (facultatif)

- A** [1000] ... nouveau jeu (crédit=1000)
- B** [1] ... génère le premier nombre Num1 = 1
- C** [118] ... génère le deuxième nombre Num2 = 118
- D** [7.55] ... calcul cote = 7.55
- 25 E** [-25] ... saisie montant du pari = 25, résultat perdu = -25
- B'** [825] ... affichage du troisième nombre Num3 = 825
- C'** [975] ... solde crédit = 975
- B** [38] ... génère le premier nombre Num1 = 38
- C** [636] ... génère le deuxième nombre = 636
- D** [0.67] ... calcul cote = 0.67
- 100 E** [67.22] ... pari = 100, résultat gagné =  $100 \cdot 0.6722 \dots = 67.22 \dots$
- B'** [181] ... affichage du troisième nombre Num3 = 181
- C'** [1042.22] ... solde crédit = 1042.22

## ML-46 (LE-13) Jeu de craps

?Last	?Bet			
Roll	Bet	Start	?Bank	Dice

Au ML-46 Craps, vous lancez deux dés. Si vous obtenez un 7 ou un 11 au premier lancer, vous gagnez. Si vous obtenez un 2, un 3 ou un 12, vous perdez. Dans les autres cas, vous jouez à nouveau.

Si vous obtenez un 7 au lancer suivant, vous perdez. Si vous obtenez à nouveau le même numéro que le premier lancer, vous gagnez. Dans les autres cas, vous jouez à nouveau.

Le lancer des dés est indiqué par le nombre M.N, où M et N sont les nombres sur les dés 1...6. Seule la somme des nombres sur les dés est importante, pas leur ordre.

**Lbl A** Lancer des dés

**Lbl A'** Affichage dernier lancer

**Lbl B** Saisie mise

Si vous gagnez, la mise est ajoutée au crédit, si vous perdez, elle est soustraite. Tant que la mise n'est pas modifiée, elle reste applicable.

**Lbl B'** Affichage mise en cours

**Lbl C** Début nouveau jeu de (crédit = 1000)

**Lbl D** Affichage crédit

**Lbl E** Lancer un dé 1...6 (sans effet sur le jeu)

*Remarque : Normalement, il n'est pas nécessaire d'initialiser le générateur de nombres aléatoires, mais pour éviter la reproductibilité des nombres aléatoires, il est possible d'initialiser le générateur nombres aléatoires par l'opération **Op 52**.*

Exemple:

**Pgm 46** ... activation du programme de bibliothèque ML-46

**0 Op 52** ... initialisation du générateur aléatoire (facultatif)

**C** ... nouveau jeu (crédit=1000)

**2 5 B** ... mise = 25

**A** [1.1] ... premier lancer  $1+1 = 2$ , perdu !

**D** [975] ... affichage solde crédit = 975

**A** [5.1] ... premier lancer  $5+1 = 6$

**A** [4.2] ... lancer suivant  $4+2 = 6$ , gagné !

**D** [1000] ... affichage solde crédit = 1000

**A** [1.3] ... premier lancer  $1+3 = 4$

**A** [4.3] ... lancer suivant  $4+3 = 7$ , perdu !

**D** [975] ... affichage solde crédit = 1000

**A** [4.6] ... premier lancer  $4+6 = 10$

**A** [4.4] ... lancer suivant  $4+4 = 8$

**A** [2.2] ... lancer suivant  $2+2 = 4$

**A** [1.6] ... lancer suivant  $1+6 = 7$ , perdu

**A** [4.5] ... premier lancer  $4+5 = 9$

**A** [5.3] ... lancer suivant  $5+3 = 8$

**A** [5.6] ... lancer suivant  $5+6 = 11$

**5 0 0 B** ... nouvelle mise = 500

**A** [5.4] ... lancer suivant  $5+4 = 9$ , gagné !

**D** [1450] ... affichage solde crédit = 1450

## ML-47 (LE-14) Jeu Atterrir sur Mars

?Burn				
Burn	Fuel	Vel	Alt	Start

Le jeu ML-47 simule un atterrissage sur Mars. Vous démarrez le jeu avec une altitude de 2603 pieds, un taux de descente de 487 pieds par seconde et une réserve de carburant de 630. Pour un atterrissage réussi, la vitesse doit être au maximum de 6, si les amortisseurs sont encore capables d'amortir... Si vous perdez du carburant au-dessus du sol, la nacelle tombera en chute libre.

Initialement, la combustion d'une unité de carburant fournira une accélération de 1 pied/sec<sup>2</sup>. À mesure que le poids du module diminue, le rendement des moteurs augmente. La gravité sur Mars est de 13 pieds/sec<sup>2</sup>.

**Lbl A** Allumage des moteurs avec une consommation de carburant spécifiée de 0 à 75

**Lbl A'** Affichage du dernier allumage des moteurs

**Lbl B** Affichage alimentation en carburant 'f'

**Lbl C** Affichage vitesse 'v'

**Lbl D** Affichage hauteur 'h'

**Lbl E** Début nouveau jeu

Exemple:

**Pgm 47** ... activation du programme de bibliothèque ML-47

**E** ... nouveau jeu, taux de descente -487, altitude 2603, carburant 630

**7 5 A** ... allumage 75, v = -422, h = 2149

5 0 A ... allumage 50,  $v = -381$ ,  $h = 1747$

5 0 A ... allumage 50,  $v = -339$ ,  $h = 1387$

5 0 A ... allumage 50,  $v = -296$ ,  $h = 1069$

2 5 A ... allumage 25,  $v = -280$ ,  $h = 782$

B [380] ... affichage solde carburant = 380

2 5 A ... allumage 25,  $v = -263$ ,  $h = 510$

7 5 A ... allumage 75,  $v = -184$ ,  $h = 286$

7 5 A ... allumage 75,  $v = -101$ ,  $h = 144$

5 0 A ... allumage 50,  $v = -47$ ,  $h = 70$

B [155] ... affichage solde carburant = 155

2 5 A ... allumage 25,  $v = -26$ ,  $h = 34$

2 0 A ... allumage 20,  $v = -11$ ,  $h = 15$

1 3 A ... allumage 13,  $v = -6$ ,  $h = 6$

1 5 A ... allumage 15,  $v = +2$ ,  $h = 4$

B [82] ... affichage solde carburant = 82

6 A ... allumage 6,  $v = -3$ ,  $h = 4$

1 0 A ... allumage 10,  $v = -1$ ,  $h = 2$

8 A ... atterrissage en douceur avec vitesse d'impact  $v = -2.82$

B [58] ... affichage du carburant restant = 58



## ML-48 Jeu de nim

Turn				N->Start

Lorsque vous jouez à Nim, vous commencez avec N allumettes. Chaque joueur prend 1 à 3 allumettes. Le joueur qui prend la dernière allumette perd. Celui qui a perdu la partie commence la partie suivante.

*Remarque : La calculatrice commet intentionnellement des erreurs tactiques occasionnelles pour donner au joueur une meilleure chance de gagner.*

**Lbl** **A** Le joueur retire 1 à 3 allumettes

**Lbl** **E** Saisie du nombre de pierres N et début d'une nouvelle partie

Exemple:

**RST** ... juste pour l'exemple, le jeu sera lancé par le joueur

**Pgm** **48** ... activation du programme de bibliothèque ML-48

**1** **5** **E** ... commencer une nouvelle partie avec 15 allumettes

**3** **A** ... le joueur retire 3 allumettes

[3] ... la calculatrice retire 3 allumettes

[9] ... reste 9 allumettes

**2** **A** ... le joueur retire 2 allumettes

[2] ... la calculatrice retire 2 allumettes

[5] ... reste 5 allumettes

**1** **A** ... le joueur retire 1 allumette

[3] ... la calculatrice retire 3 allumettes

[1] ... reste 1 allumette

**1** **A** ... le joueur retire la dernière allumette donc perdu !

## ML-49 Mesure du temps de réaction

Start				

Le programme ML-49 est utilisé pour mesurer le temps de réaction. Après avoir appuyé sur **A**, le programme attend un moment (un temps aléatoire, de sorte que le moment ne soit pas prévisible), puis il commence à mesurer le temps et la tâche consiste à appuyer sur n'importe quel bouton de la calculatrice (sauf les boutons **2nd**, **GTO** et **R/S**). Il affiche le temps de réaction en secondes, avec une résolution de 10 ms.

**A** Début d'une nouvelle mesure

## ML-50 (LE-20) Jeu de bataille navale

?Range	?Bear	?Display		Clear
Range	Bear	Fire		Start

En raison des dégâts de combat, votre frégate n'a ni radar ni moteur fonctionnels. Les dernières informations indiquaient qu'un sous-marin s'approchait à une distance de 15 000 mètres d'une direction inconnue, avec une vitesse initiale de 30 nœuds. Une vitesse de 30 nœuds correspond à une distance parcourue d'environ 1 000 mètres par minute. Votre frégate est équipée de 15 torpilles, capables de tirer 1 torpille par minute. Une fois le coup de feu tiré, les renseignements secrets sont

capables de rapporter le résultat du coup en fonction de la distance par rapport à la cible :

- plus de 500 : Aucun dégât sous-marin. Le sous-marin se dirige droit vers vous.

- 50 à 500 : Dégâts partiels sous-marins. La vitesse du sous-marin est réduite de 6 nœuds (soit 200 yards par minute, la vitesse minimale est de 6 nœuds) et le sous-marin tente de s'échapper en changeant de direction de 45°. Avec un nouveau raté de plus de 500, le sous-marin continue de foncer droit sur vous.

- Moins de 50 : Naufrage du sous-marin.

Votre frégate coulera si elle a tiré toutes ses torpilles sans couler le sous-marin, ou si le sous-marin s'approche à moins de 500 mètres de vous.

Les conditions de départ peuvent être légèrement différentes la prochaine fois que vous jouerez. Si vous souhaitez toujours démarrer la partie avec les mêmes conditions, utilisez la fonction **E** avant la partie.

**Lb| A** Réglage de la distance d'impact des torpilles en yards

Si réglage sur 0, aucune torpille tirée, mais le sous-marin se déplace.

**Lb| A'** Affichage de la distance d'impact de la torpille en cours

**Lb| B** Réglage de la direction de la torpille en degrés

**Lb| B'** Affichage de la direction de la torpille en cours

**Lb| C** Tirer une torpille avec la direction et la distance définies.

Chaque étape dure 1 minute de jeu, pendant laquelle le sous-marin se rapproche. L'état est affiché sous la forme du nombre XXXX.NN où XXXX est la distance jusqu'à la cible et NN est le nombre de torpilles restantes. XXXX.00 clignotant signifie qu'il n'y a plus de torpilles. 0.NN indique des cibles à plus de 5 000 mètres par minute. XX.NN clignotant signifie que le sous-marin a coulé.

**Lb| C'** Affichage le résultat du dernier tir

**Lbl E** Commencer un nouveau jeu

L'état initial est 15000,15, c'est-à-dire portée du sous-marin 15 000 yards et 15 torpilles restantes.

**Lbl E'** Réinitialiser les registres de jeu à leur état par défaut

Exemple:

**Pgm 50** ... activation du programme de bibliothèque ML-50

**E'** ... réinitialiser les registres de jeu (facultatif)

**E** [15000.15] ... nouvelle partie, distance 15000, 15 torpilles

... la portée attendue du sous-marin est de 15 000 - 1 000 = 14 000 yards

**14000A** ... réglage de la portée des torpilles à 14 000 yards

**90B** ... réglage de la direction de la torpille à 90 degrés

**C** [0.14] ... distance de la cible à plus de 5 000 mètres

... la portée attendue du sous-marin est de 14 000 - 1 000 = 13 000 yards

**13000A** ... réglage de la portée des torpilles à 13 000 yards

**180B** ... réglage de la direction de la torpille à 180 degrés

**C** [0.13] ... distance de la cible à plus de 5 000 yards

... la portée attendue du sous-marin est de 13 000 - 1 000 = 12 000 yards

**12000A** ... réglage de la portée des torpilles à 12 000 yards

**270B** ... réglage de la direction de la torpille à 270 degrés

**C** [478.12] ... distance de la cible par 478 yards. Le sous-marin a ralenti à 24 nœuds, soit 800 yards par minute, et a changé de direction de 45°.

... la portée attendue du sous-marin est de  $12000 - 800/2 = 11600$  yards

**1 1 6 0 0 A** ... réglage de la portée des torpilles à 11600 yards

**2 7 2 B** ... réglage de la direction de la torpille à 272 degrés

**C** [1266.11] ... distance en dessous de 5 000 yards, le sous-marin continue en ligne droite avec une vitesse de 800 yards par minute

... la portée attendue du sous-marin est de  $11600 - 800 = 10800$  yards

**1 0 8 0 0 A** ... réglage de la portée des torpilles à 10800 yards

**2 6 8 B** ... réglage de la direction de la torpille à 268 degrés

**C** [425.10] ... distance de la cible à 425 mètres. Le sous-marin a ralenti à 18 nœuds, soit 600 yards par minute, et a changé de direction de 45°.

... la portée attendue du sous-marin est de  $10800 - 600/2 = 10500$  yards.

**1 0 5 0 0 A** ... réglage de la portée des torpilles à 10500 yards

**2 6 5 B** ... réglage de la direction de la torpille à 265 degrés

**C** [164.09] ... distance de la cible à 164 mètres. Le sous-marin a ralenti à 12 nœuds, soit 400 mètres par minute, et a changé de direction de 45°.

... la portée attendue du sous-marin est de  $10500 - 400/2 = 10300$  yards

**1 0 3 0 0 A** ... réglage de la portée des torpilles à 10300 yards

**2 6 4 B** ... réglage de la direction de la torpille à 264 degrés

**C** [181.08] ... distance de la cible à 181 mètres. Le sous-marin a ralenti à 6 nœuds, soit 200 mètres par minute, et a changé de direction de 45°.

... la portée attendue du sous-marin est de  $10300 - 200/2 = 10200$  yards

**1 0 2 0 0 A** ... réglage de la portée des torpilles à 10200 yards

2 6 1 B ... réglage de la direction de la torpille à 261 degrés

C [255.07] ... distance de la cible à 255 mètres. Le sous-marin continue à une vitesse de 6 nœuds, soit 200 yards par minute, et est dévié de 45°.

... la portée attendue du sous-marin est de  $10200 - 200/2 = 10100$  yards

1 0 1 0 0 A ... réglage de la portée des torpilles à 10100 yards

2 6 2 B ... réglage de la direction de la torpille à 262 degrés

C [176.06] ... distance de la cible à 176 mètres. Le sous-marin continue à une vitesse de 6 nœuds, soit 200 yards par minute, et est dévié de 45°.

... la portée attendue du sous-marin est de  $10100 - 200/2 = 10000$  yards

1 0 0 0 0 A ... réglage de la portée des torpilles à 10000 yards

2 6 3 B ... réglage de la direction de la torpille à 263 degrés

C [100.05] ... distance de la cible à 100 mètres. Le sous-marin continue à une vitesse de 6 nœuds, soit 200 mètres par minute et est dévié de 45°.

... la portée attendue du sous-marin est de  $10000 - 200/2 = 9900$  yards

9 9 0 0 A ... réglage de la portée des torpilles à 9900 yards

2 6 4 B ... réglage de la direction de la torpille à 264 degrés

C [26.05] *clignotant* ... touché avec une portée de 26 mètres, il reste 5 torpilles