Manuel compilateur Bax58C

Bax58C est un programme qui compile des programmes ***Basic*** pour générer des programmes à destination de l'émulateur **TI58C**.

SOMMAIRE

La **TI58C** est une calculatrice programmable Texas Instruments des années 70/80.

Elle est dotée d'un langage spécifique (LMS : Langage Machine Spécialisé),

- 480 pas de programme **ou** 60 mémoires (1 pas = 2 nibbles, 1 mémoire = 8 octets), le partage pas/ mémoires étant partitionnable. (Ex: 240 pas **et** 30 mémoires)
- mémoire RAM = 480 octets, moins d'un demi K octet (presque 4K bits!),
- un module ROM, appelé "Bibliothèque de base", de 5000 pas est fourni en standard. (de nombreux modules existent en option)
- afficheur diodes rouges 10 chiffres à 7 segments
- mémoire constante (pour la TI58C, pas pour la TI58 ni la TI59 !)

Une imprimante **PC100** complète cette calculatrice programmable.

Pour connecter la **TI58C** à l'imprimante, il faut retirer les accus pour enficher la calculette sur le connecteur de l'imprimante. (Petite astuce judicieuse : les accus peuvent être logés dans une trappe qui en assure la charge.)
Pas d'encre, l'imprimante "brûle" un papier thermique qui a l'inconvénient de s'effacer avec le temps...

Des modules ROM de 5000 pas contenant une vingtaine de programmes utilitaires sont ainsi disponibles:

- ML Master Library (fourni avec la TI)
- MU Math Utilities
- LE Module Loisir (“Leisure Library”)
- SY Module "Surveying"
- EE Module “Electrical Engineering"
- ... et bien d'autres encore !

Une interface télé a aussi été commercialisée par une société d'éléctronique française.
Cette interface possède un receptacle identique au socle de connection de l'imprimante et se branche sur une télévision (Noir&Blanc) avec un cable d'antenne coaxial.
L'affichage correspond exactement à ce qui sort usuellement sur l'imprimante avec le même mode "trace" qui permet de suivre pas à pas l'exécution du programme à debugger. (et couteux en papier avec l'imprimante!)

Le langage des calculatrices **TI-58/TI-58C/TI-59** est considéré comme un langage machine spécialisé (**LMS**) qui utilise la notation algébrique directe (**AOS** [**A***lgebraic* **O***perating* **S***ystem*]).

En 1983, deux étudiants de la "***Naval Postgraduate School***" de Monterey (Californie) avaient écrit une thèse intitulée "*Design and implementation of a Basic Cross-Compiler and virtual memory management for the TI-59 programmable calculator*".
Cette thèse était sur l'implémentation d'un compilateur (écrit en Pascal) de WBASIC vers le langage AOS de la TI-59.

Depuis 3 programmes ont été développés :

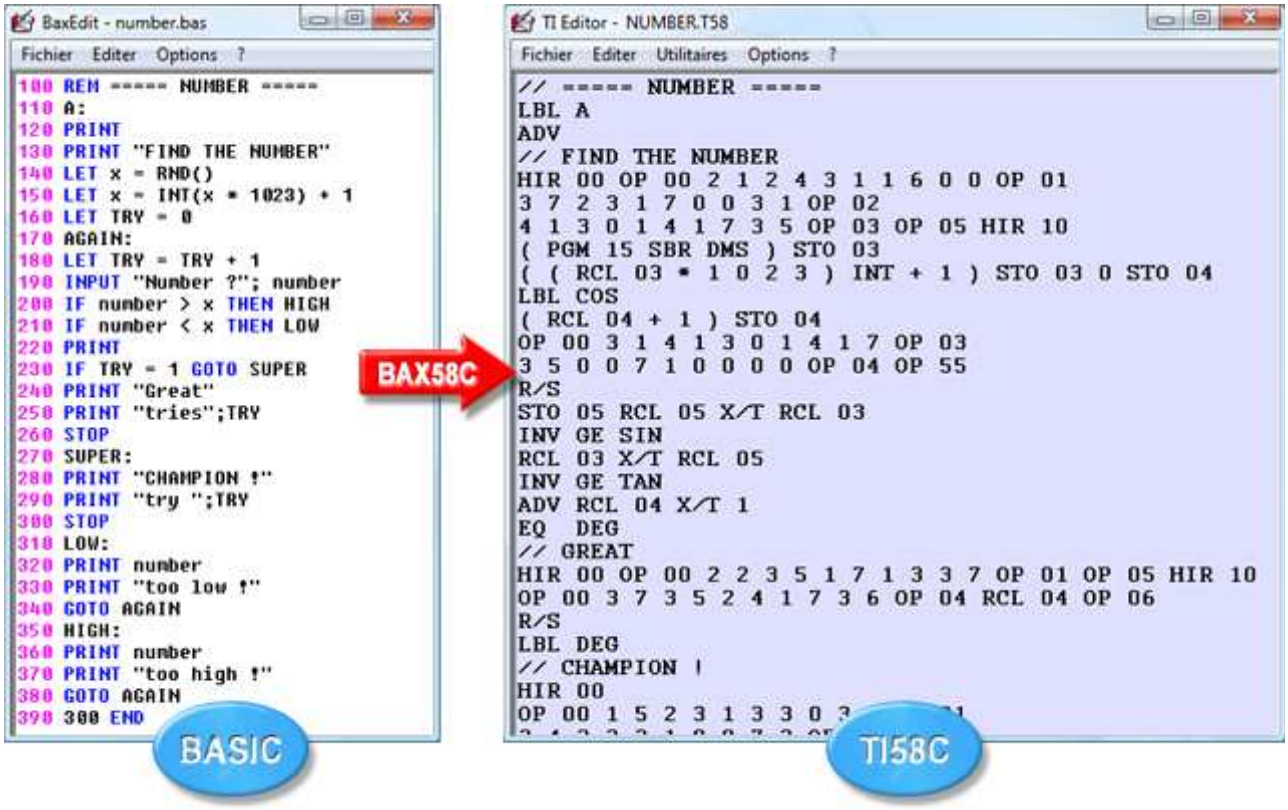
- Compilateur Basic vers **TI59** de **Claudio Larini** : ce compilateur, ***BAX59***, est très fortement inspiré de la thèse de 1983 et reprend l'idée d'une compilation de Basic vers le langage **AOS** de la **TI**.
  (http://www.claudiolarini.altervista.org/bax59.htm)

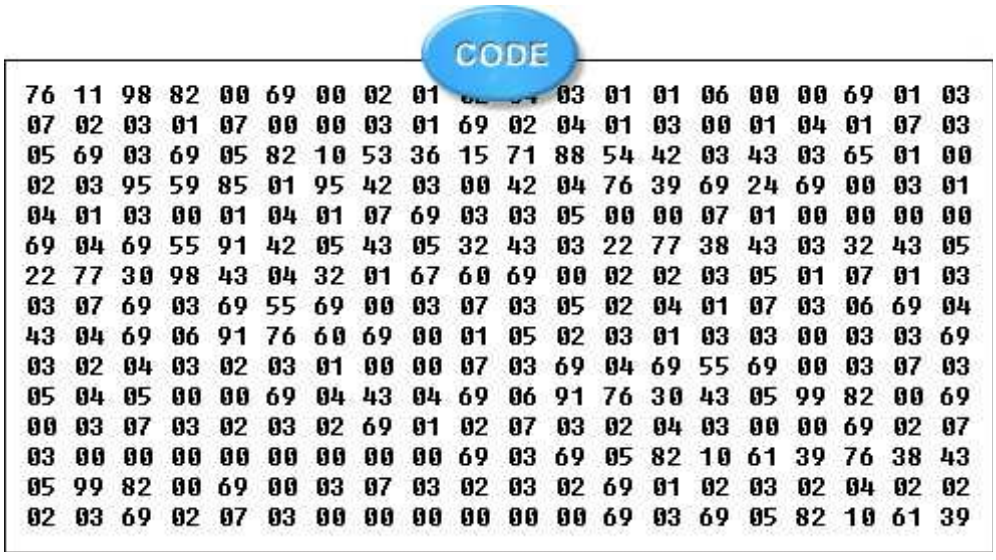
- Compilateur de **Guillaume Tello** : le langage de départ est inspiré du langage de la **TI** amélioré : noms de variable, noms d'étiquettes, de flags, commentaires, LOOP, WHILE, UNTIL, FOR, IF, ELSE, SELECT, CASE ...
  (http://gtello.pagesperso-orange.fr/ti58_f.htm)

- ***T_Compiler*** de **Philippe Tivolle** traduit des programmes écrits en "**T**" qui est un langage de programmation simple orienté objet, modelé sur **Java**. (Langage un peu trop spécifique !)
  (http://ti59compiler.wixsite.com/ti59)

La compilation

Le langage Basic étant plus connu et surtout plus facilement abordable, une compilation de programmes écrits dans ce langage pour donner des programmes dans le langage des calculatrices **TI-58/TI-58C/TI-59** peut être considéré comme un complément logique à l'émulateur **TI58C**.



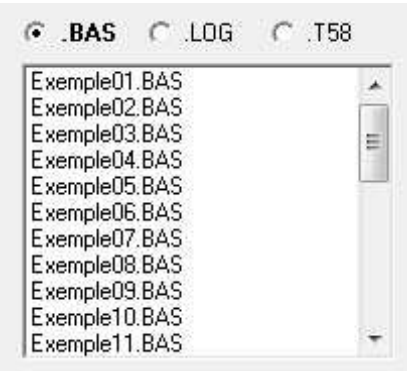


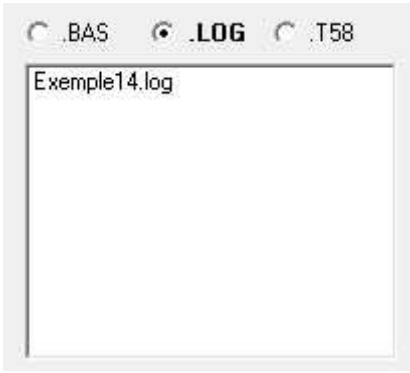
Chaque compilation de programme Basic [***.bas***] produit un programme **TI58C** [***.t58***] et un fichier de trace de la compilation [***.log***] utile à la compréhension du processus et aux éventuelles corrections nécessaires du programme Basic source.

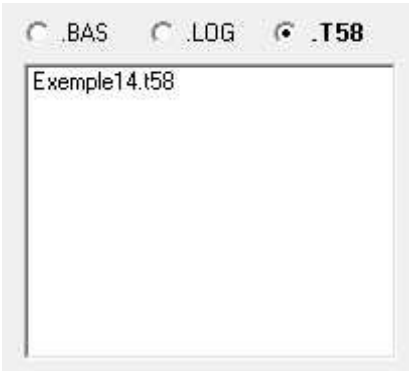
```
============================================================
  Compilateur BAX58C  - 12 / 11 / 2020 09:49:07
============================================================
  C:\ProgramData\TI58C\Basic\Bax58C\Nombre\number.bas
============================================================
  100 REM ===== NUMBER =====
  110 A:
  120 PRINT
  130 PRINT "   FIND THE NUMBER"
  140 LET x = RND()
  150 LET x = INT(x * 1023) + 1
  160 LET TRY = 0
  170 REM --- PLAY ---
  180 AGAIN:
  190 LET TRY = TRY + 1
  200 INPUT "Number ?"; number
  210 IF number > x THEN HIGH
  220 IF number < x THEN LOW
  230 PRINT
  240 IF TRY = 1 GOTO SUPER
  250 REM --- GREAT ---
  260 PRINT "tries";TRY
  270 PRINT "Great !"
  280 STOP
  290 SUPER:
  300 REM --- CHAMPION ---
  310 PRINT "try ";TRY
  320 PRINT "CHAMPION !"
  330 STOP
  340 REM --- TOO LOW ! ---
  350 LOW:
  360 PRINT number
  370 PRINT "  too low !"
  380 GOTO AGAIN
  390 REM --- TOO HIGH ---
  400 HIGH:
  410 PRINT number
  420 PRINT "  too high !"
  430 GOTO AGAIN
  440 300 END
============================================================
                      number.t58
============================================================
                    Steps     :   312
                    Registers :   6
============================================================
                     R E G I S T E R S
                  Basic Name      Reg #
                  ----------      -----
                  Reserved         00
                  Reserved         01
                  Reserved         02
                  x            =>  03
                  TRY          =>  04
                  number       =>  05
============================================================
                      L A B E L S
                  Basic Addr      TI58 LBL
                  ----------      --------
                  A            =>   A
                  AGAIN        =>   COS
                  HIGH         =>   SIN
                  LOW          =>   TAN
                  SUPER        =>   DEG
============================================================

------------------------------------------------ Basic Line 100
//  ===== NUMBER =====
------------------------------------------------ Basic Line 110
LBL A
------------------------------------------------ Basic Line 120
ADV
------------------------------------------------ Basic Line 130
//    FIND THE NUMBER
HIR 00 OP 00
2 1 2 4 OP 01
3 1 1 6 0 0 3 7 2 3 OP 02
1 7 0 0 3 1 4 1 3 0 OP 03
1 4 1 7 3 5 0 0 0 0 OP 04
OP 05 HIR 10

------------------------------------------------ Basic Line 140
0 RND STO 03
------------------------------------------------ Basic Line 150
RCL 03 * 1 0 2 3 = INT + 1 = STO 03
------------------------------------------------ Basic Line 160
0 STO 04
------------------------------------------------ Basic Line 170
//  --- PLAY ---
------------------------------------------------ Basic Line 180
LBL COS
------------------------------------------------ Basic Line 190
OP 24
------------------------------------------------ Basic Line 200
OP 00  3 1 4 1 3 0 1 4 1 7 OP 03
3 5 0 0 7 1 0 0 0 0 OP 04
OP 55 R/S
STO 05
------------------------------------------------ Basic Line 210
RCL 05 X:T RCL 03 INV GE SIN
------------------------------------------------ Basic Line 220
RCL 03 X:T RCL 05 INV GE TAN
------------------------------------------------ Basic Line 230
ADV
------------------------------------------------ Basic Line 240
RCL 04 X:T 1 EQ DEG
------------------------------------------------ Basic Line 250
//  --- GREAT ---
------------------------------------------------ Basic Line 260
OP 00  3 7 3 5 2 4 1 7 3 6 OP 04 RCL 04 OP 06
------------------------------------------------ Basic Line 270
OP 00
2 2 3 5 1 7 1 3 3 7 OP 03
7 3 0 0 0 0 0 0 OP 04
OP 55
------------------------------------------------ Basic Line 280
R/S
------------------------------------------------ Basic Line 290
LBL DEG
------------------------------------------------ Basic Line 300
//  --- CHAMPION ---
------------------------------------------------ Basic Line 310
OP 00  3 7 3 5 4 5 0 0 OP 04 RCL 04 OP 06
------------------------------------------------ Basic Line 320
OP 00
1 5 2 3 1 3 3 0 3 3 OP 03
2 4 3 2 3 1 0 0 7 3 OP 04
OP 55
------------------------------------------------ Basic Line 330
R/S
------------------------------------------------ Basic Line 340
//  --- TOO LOW ! ---
------------------------------------------------ Basic Line 350
LBL TAN
------------------------------------------------ Basic Line 360
RCL 05 PRT
------------------------------------------------ Basic Line 370
//   too low !
HIR 00 OP 00
3 7 3 2 3 2 OP 01
2 7 3 2 4 3 0 0 OP 02
7 3 0 0 0 0 0 0 0 0 OP 03
OP 05 HIR 10
------------------------------------------------ Basic Line 380
GTO COS
------------------------------------------------ Basic Line 390
//  --- TOO HIGH ---
------------------------------------------------ Basic Line 400
LBL SIN
------------------------------------------------ Basic Line 410
RCL 05 PRT
------------------------------------------------ Basic Line 420
//   too high !
HIR 00 OP 00
3 7 3 2 3 2 OP 01
2 3 2 4 2 2 2 3 OP 02
7 3 0 0 0 0 0 0 OP 03
OP 05 HIR 10
------------------------------------------------ Basic Line 430
GTO COS
------------------------------------------------ Basic Line 440
============================================================
```


Utilisation du programme

Les programmes Basic à compiler sont d'extension **.BAS** et leur compilation génère un fichier "trace" de la compilation d'extension **.LOG** ainsi qu'un programme pour l'émulateur d'extension **.T58**

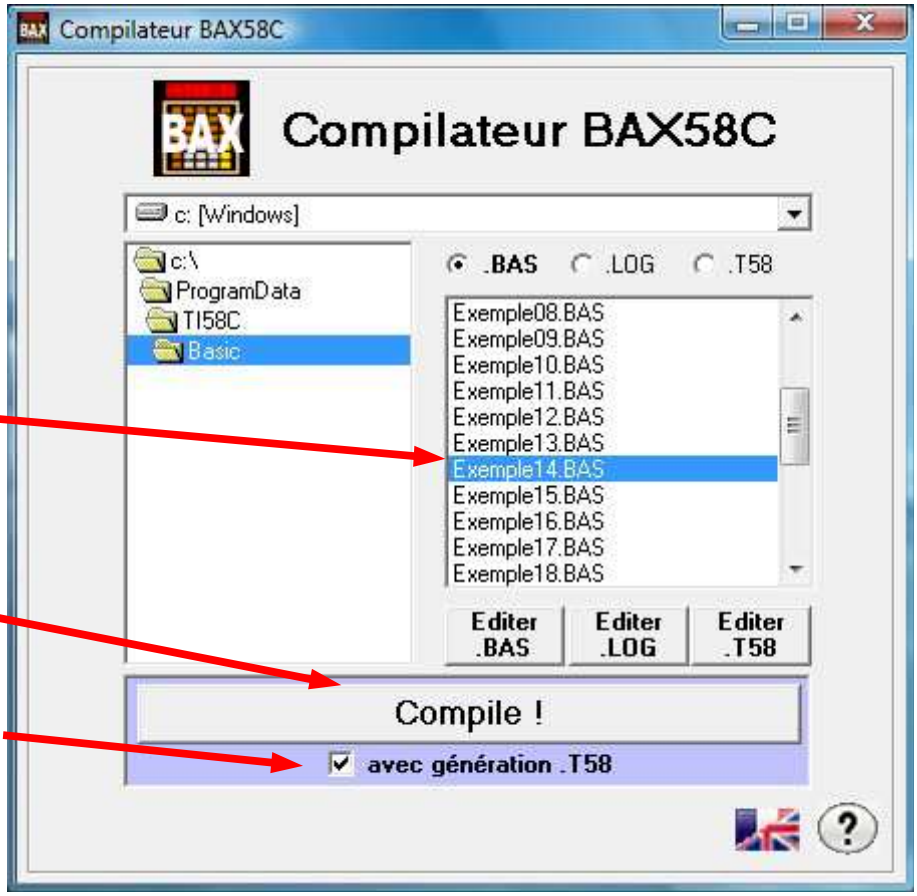






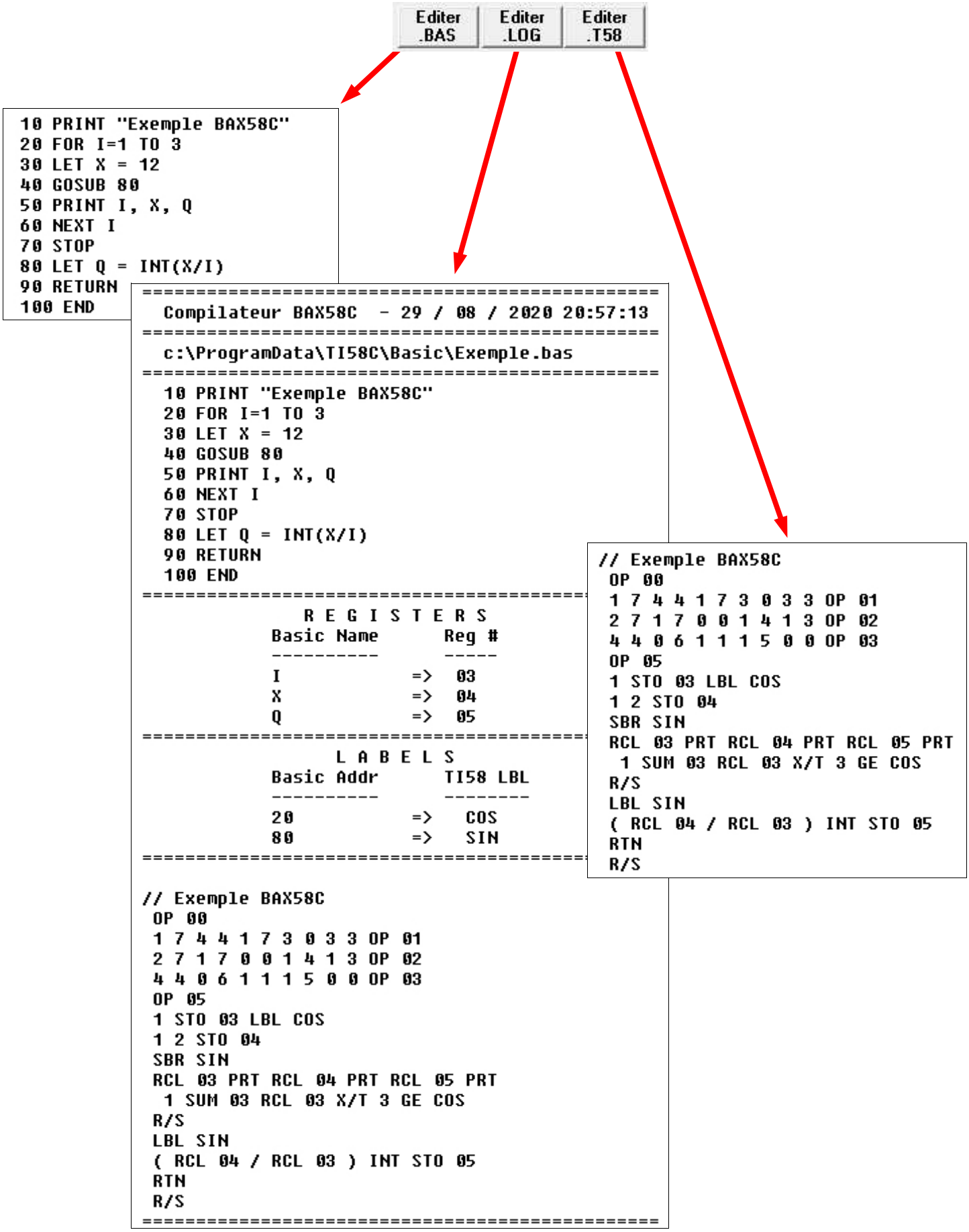
- Sélectionner le programme à compiler
- Cliquer sur **Compile !**

en choisissant au préalable
si le programme TI58 doit être généré ou pas (simulation)



A la fin de la compilation, un récapitulatif s'affiche et le fichier **.LOG** peut être consulté.

A tout moment, les fichiers **.BAS** (programme Basic), **.LOG** (“Trace” de compilation) et **.T58** (programme TI58c) peuvent être ouverts pour édition ou consultation.



Le langage Basic

Depuis l'origine du langage Basic, Darmouth College en Octobre 1964, de nombreuses versions ont émergées.
Chaque système informatique interactif a eu sa version, des plus puissants mainframe aux “ordinateurs de poche"...

Quand aux PC que ce soit sous Windows, Linux ou autres OS les variantes sont pléthores. (QuickBasic, QBasic, Bywater Basic, True Basic, PowerBasic, Turbo Basic, wxBasic, Liberty Basic, SmallBasic, FreeBasic...)

La plus répandue, Qbasic et les versions QB45 et QB64, ont inspiré le Basic utilisable avec le compilateur Bax58C.
Toutes les fonctionnalités de Qbasic n'ont pas été implémentées et des variantes ont dû être introduites pour s'adapter au langage cible.

Chaque ligne du programme Basic commence par un numéro de ligne suivi d'au moins un espace puis d'une étiquette ou d'une instruction.

Une instruction pourra contenir :

- du texte <*texte*>,
- des noms de variables <*variable*>,
- des valeurs numériques ou alphanumériques <*valeur*> (les valeurs alphanumériques étant délimitées par des guillemets **“**...**"**),
- des expressions d'affectation <*expression*>,
- des expressions conditionnelles <*condition*>,
- des adresses de branchement : numéros de ligne ou étiquettes <*ligne*> ou <*étiquette*>.

<étiquette>:

définition d'étiquette de branchement.

Syntaxe :

<*étiquette*>**:**

<*étiquette*> pouvant être n'importe quel nom alphanumérique ou un label TI58C (A, B, C, D, E, A', B', C', D', E')

Exemple :

```
110 A:
  .../...
240 Calcul:
  .../...
380 FinProg:
  .../...
```

Equivalent TI58C :

```
LBL A
  .../...
LBL COS
  .../...
LBL DEG
  .../...
```

Remarques:

Toutes les étiquettes "Basic" sont traduites en labels TI (LBL COS...) sauf les étiquettes correspondantes à des touches de fonction TI qui sont reprises à l'identique (A... E, A'... E')

CALL

appel de sous-routine externe.

Syntaxe :

CALL <étiquette> : { [<valeur>#] <valeur> | <variable> }

Variante :
LET <variable> =CALL <étiquette> : { [<valeur>#] <valeur> | <variable> }
ou
LET <variable> =CALL(<étiquette> : { [<valeur>#] <valeur> | <variable> })

<valeur># numéro ou code du module de librairie (01, 02, 03,... ou ML, ST, RE,...)
<*valeur*> numéro de programme (suffixe du nom de programme).
<*variable*> nom de la variable contenant le numéro de programme (suffixe du nom de programme).

Exemple :

```
150 CALL A:01
220 CALL B:Var
260 CALL WRI:MU#01
310 CALL A:PH#05
```

Equivalent TI58C :

```
LPG 01 A
LP* 03 B
LIB 10 PGM 01 SBR WRI
LIB 14 PGM 05 A
```

CHAIN

chargement de programme.

Syntaxe :

CHAIN { <valeur> | <variable> }

<*valeur*> numéro de programme (suffixe du nom de programme).
<*variable*> nom de la variable contenant le numéro de programme (suffixe du nom de programme).

Exemple :

```
120 CHAIN 01
140 CHAIN Var
```

Equivalent TI58C :

LDP 01
LD* 03

CLEAR

effacement mémoire.

Syntaxe :

CLEAR

Exemple :

230 CLEAR

Equivalent TI58C :

CMS

CLS

effacement écran.

Syntaxe :

CLS

Exemple :

230 CLS

Equivalent TI58C :

CLR

DIM

définition de variables.

Syntaxe :

DIM <*variable*> [, <*variable*> [, <*variable*>] ...]

<*variable*> pouvant être

- une variable numérique : nom alphanumérique
- une variable alphanumérique : nom alphanumérique se terminant par $
- un drapeau (ou flag) : nom alphanumérique se terminant par % (10 maxi)
- un tableau : nom alphanumérique suivi de (<*valeur*>)

Exemple :

```
110 REM === variables numériques
120 DIM Montant, Taux
130 REM === variables alphanumériques
140 DIM NOM$, Message$
150 REM === drapeaux
160 DIM Flag1%, Flag2%, Flag3%
170 REM === tableaux
180 DIM MyArray(10)
```

Equivalent TI58C :

aucun, l'instruction DIM permet de réserver, en précompilation, les registres à utiliser dans le programme TI58C.

Remarques:

Variable alphanumérique : si déclarée par DIM (Ex: DIM Var$) une variable prend 4 registres (20 caractères alphanumériques) si non déclarée une variable ne prend qu'un registre (5 caractères alphanumériques)

DO...LOOP

boucle conditionnelle.

Syntaxe :

[DO] { WHILE | UNTIL } <*expression*>
LOOP

Exemple :

```
 25 LET CPT = 0
 30 DO UNTIL CPT = 10
 40    LET CPT = CPT + 1
 50    PRINT CPT
 60 LOOP

 25 LET VAL = 0
 30 WHILE VAL < 5
 40    LET VAL = VAL + 1
 50    PRINT VAL
 60 LOOP
```

Equivalent TI58C :

```
0 STO 03
LBL COS
 ( RCL 03 + 1 ) STO 03
 RCL 03 PRT
 1 0 X:T RCL 03 INV EQ COS

0 STO 09
LBL DEG
 ( RCL 09 + 1 ) STO 09
 RCL 09 PRT
 5 X:T RCL 09 INV GE DEG
```


END

fin de programme.

Syntaxe :

END

Exemple :

230 END

Equivalent TI58C :

R/S

FOR...NEXT

boucle selon compteur.

Syntaxe :

FOR <*variable*> **=** { <*variable*> **|** <*valeur*> } TO { <*variable*> **|** <*valeur*> } [STEP { <*variable*> **|** <*valeur*> }]
NEXT <*variable*>

Exemple :

```
20 FOR I=3 TO 1 STEP -1
25     FOR Z=1 TO 4
40     NEXT Z
45 NEXT I
```

Equivalent TI58C :

```
 3 STO 03 LBL COS
 1 STO 04 LBL SIN
 1 SUM 04 RCL 04 X/T 4 GE SIN
 DSZ 03 COS
```


GOSUB

appel de sous-programme.

Syntaxe :

GOSUB { <*ligne*> | <*étiquette*> }

<*ligne*> faisant référence à un numéro de ligne de programme existant.
<*étiquette*> faisant référence à une étiquette déclarée

Exemple :

```
110 GOSUB 240
  .../...
240 GOSUB FinProg
  .../...
380 FinProg:
  .../...
```

Equivalent TI58C :

SBR COS

GOTO

branchement inconditionnel.

Syntaxe :

GOTO { <*ligne*> | <*étiquette*> }

<*ligne*> faisant référence à un numéro de ligne de programme existant.
<*étiquette*> faisant référence à une étiquette déclarée

Exemple :

```
110 GOTO 240
  .../...
240 GOTO FinProg
  .../...
380 FinProg:
  .../...
```

Equivalent TI58C :

GTO COS

IF

branchement conditionnel.

Syntaxe :

IF <*condition*> THEN [GOTO] { <*ligne*> | <*étiquette*> }

<condition> peut être :

- [NOT] <*drapeau*> = { 0 | 1 | TRUE | FALSE }
- [NOT] { <*variable*> | <*expression*> | <*valeur*> } { **>** | **<** | **>=** | **<=** | **<>** | **=** } { <*variable*> | <*expression*> | <*valeur*> }

Exemple :

```
20 IF A > 10 THEN GOTO 200
30 IF A < 20 THEN 300
40 IF A >= 10 GOTO FinProg
50 IF A <= 20 THEN GOTO 300
60 IF A <> 20 THEN 300
70 IF A = 10 GOTO 200
 80 IF Flg% THEN GOTO 200
100 IF NOT DRAP% GOTO 300
```

Equivalent TI58C :

```
 RCL 03 X:T 1 0 INV GE COS
 2 0 X:T RCL 03 INV GE SIN
 1 0 X:T RCL 03 GE X2
 RCL 03 X:T 2 0 GE SIN
 2 0 X:T RCL 03 INV EQ SIN
 RCL 03 X:T 1 0 EQ COS
 IFF 0 COS
 INV IFF 1 SIN
```

INPUT

entrée de donnée.

Syntaxe :

INPUT [{ <*variable*> | <*valeur*> } ,] [<étiquette>**:**]<*variable*>

<étiquette> doit correspondre à un label TI58C (A, B, C, D, E, A', B', C', D', E')

Exemple :

```
  30 INPUT VAL
  40 INPUT NOM$; N$
130 INPUT "Rang ?";RANG
180 INPUT "Choix ?";A:Var
```

Equivalent TI58C :

```
 R/S STO 03
 OP 00 RCL 04 OP 03 OP 55 R/S STO 05
 OP 00  3 5 1 3 3 1 2 2 0 0 OP 03
  7 1 0 0 0 0 0 0 0 0 OP 04
 OP 55 R/S STO 06
OP 00  1 5 2 3 3 2 2 4 4 4 OP 03
0 0 7 1 0 0 0 0 0 0 OP 04
OP 55 R/S LBL A
STO 07
```


LET

affectation de données.

Syntaxe :

LET <*destination*> = { <*expression*> | <*variable*> | <*valeur*> } | { TRUE | FALSE }

<*destination*> pouvant être

- une variable numérique,
- une variable alphanumérique,
- un drapeau (ou flag),
- une cellule d'un tableau.

<*expression*> peut être constituée de variables, de valeurs, de fonctions, d'opérateurs et de paranthèses :

- fonctions : EXP(), LOG(), SIN(), COS(), TAN(), ATN(), ABS(), SQR(), INT(), FRAC(), SGN(), RND(), ASC(), CHR()
- opérateurs : + - * / ^ &

Exemple :

```
 45 LET A(1) = B(3) + C(5)
 50 LET Val0 = EXP(F) + SIN(G)
 60 LET Val1 = 0.0000001
 70 LET X% = TRUE
 80 LET F = 3
 100 LET F = F * X
 120 LET G = G + ( 1 / F )
 130 LET D = G - E

 170 DIM A$
 180 LET A$ = CHR(13) & CHR(14) & CHR(15) & CHR(16) & "ZZZ" & CHR(17) & CHR(33) & "YYY"
```

Equivalent TI58C :

```
RCL 15 + RCL 22 = STO 03
RCL 24 INV LNX + RCL 25 SIN = STO 23
0 . 0 0 0 0 0 0 1 STO 26
STF 00
3 STO 24
RCL 24 * RCL 27 = STO 24
RCL 25 + ( 1 / RCL 24 ) = STO 25
RCL 25 - RCL 29 = STO 28

1 3 1 4 1 5 1 6 4 6 STO 03
4 6 4 6 1 7 3 3 4 5 STO 04
4 5 4 5 0 0 0 0 0 0 STO 05
0 STO 06
```


ON...GOTO

branchement conditionnel.

Syntaxe :

ON <*variable*> GOTO { <*ligne*> | <*étiquette*> } [{ <*ligne*> | <*étiquette*> } [, { <*ligne*> | <*étiquette*> }] ...]

Exemple :

```
30 ON VALEUR GOTO 100, 200, 300
90 ON COMPTEUR GOTO Etiq1, Etiq2, Etiq3, Etiq4
```

Equivalent TI58C :

```
RCL 03 1 EQ COS 2 EQ SIN 3 EQ TAN
RCL 07 1 EQ DEG 2 EQ GRD 3 EQ RAD 4 EQ X2
```

PRINT

impression.

Syntaxe :

PRINT [{ <*variable*> | <*valeur*> } [, { <*variable*> | <*valeur*> }]]

Exemple :

```
200 PRINT VAL
210 PRINT "N ";RANG
220 PRINT "F(N)";LONG
240 PRINT
260   PRINT "--------------------"
```

Equivalent TI58C :

```
RCL 04 PRT
 OP 00  3 1 0 0 OP 04 RCL 03 OP 06
 OP 00  2 1 5 5 3 1 5 6 OP 04 RCL 05 OP 06
ADV
// --------------------
 HIR 00 OP 00
 2 0 2 0 2 0 2 0 2 0 OP 01
 OP 02
 OP 03
 OP 04
 OP 05 HIR 10
```


PRINT TAB

impression alphanumérique avec espacement.

Syntaxe :

PRINT TAB({<*variable*> | <*valeur*>} ; { <*variable*> | <*valeur*> | “*” | “.” }]

Exemple :

```
200 PRINT TAB(8);"*"
220 PRINT TAB(42);"."
230 PRINT TAB(8);"XYZ"
```

Equivalent TI58C :

```
8 OP 07
4 2 OP 77
HIR 00 OP 00
4 4 4 5 OP 02
4 6 0 0 0 0 0 0 0 0 OP 03
OP 05 HIR 10
```


PRINT USING

impression avec masque.

Syntaxe :

PRINT USING {<*masque*> | <*adresse*>} [, { <*variable*> | <*valeur*> }]

Exemple :

```
200 PRINT USING ".#####^^^",N
210 PRINT USING 260, PI

260  :  USING "#.####"
```

Equivalent TI58C :

```
ENG FIX 5
RCL 04 PRT
FIX 4
PI PRT
```


REM

introduction d'un commentaire dans le programme.

Syntaxe :

REM [<*texte*>]

Exemple :

110 REM ***** Ceci est un commentaire *****
120 REM

Equivalent TI58C :

// ***** Ceci est un commentaire *****
NOP

Remarques:

Dans le cas de plusieurs commentaires successifs, seul le dernier sera conservé par l'émulateur TI58C.

Les instructions REM non suivis de texte commentaire seront traduites en NOP.

RETURN

retour de sous-programme.

Syntaxe :

RETURN

Exemple :

```
110 GOSUB 240
  .../...
240 REM ===== SOUS-PROGRAMME
  .../...
280 RETURN
```

Equivalent TI58C :

RTN

SET

paramètres.

Syntaxe :

SET <*parametre*> [, <*parametre*> [, <*parametre*>]]

<*parametre*> peut être :

- ENG, NOT ENG, ENG=TRUE, ENG=FALSE, ENG TRUE, ENG FALSE...
- DEG, RAD, GRD
- FIX={ 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | TRUE | FALSE }...

Exemple :

```
230 SET ENG FALSE
240 SET FIX=4, DEG
260 SET NOT ENG = FALSE
```

Equivalent TI58C :

```
INV ENG
FIX 4  DEG
ENG
```


SLEEP

pause.

Syntaxe :

SLEEP [<*valeur*>]

Exemple :

230 SLEEP
250 SLEEP 3

Equivalent TI58C :

PAU
PAU PAU PAU

SOUND

joue une note de musique.

Syntaxe :

SOUND <*valeur1*> , { <*valeur2*> | <*variable2*> }

<*valeur1*> note de la gamme (C1,C1#,D ,...,F3#,G3,G3#) ou fréquence en Hertz (selon tableau)

1	**A**	**A#**	**B**	**C**	**C#**	**D**	**D#**	**E**	**F**	**F#**	**G**	**G#**
	la	**la#**	**si**	**do**	**do#**	**ré**	**ré#**	**mi**	**fa**	**fa#**	**sol**	**sol#**
	110	116.54	123.47	130.81	138.59	146.83	155.56	164.81	174.61	185	196	207.65
2	**A**	**A#**	**B**	**C**	**C#**	**D**	**D#**	**E**	**F**	**F#**	**G**	**G#**
	la	**la#**	**si**	**do**	**do#**	**ré**	**ré#**	**mi**	**fa**	**fa#**	**sol**	**sol#**
	220	233.08	246.94	**261.63**	**277.18**	**293.66**	**311.13**	**329.63**	**349.23**	**369.99**	**392**	**415.3**
3	**A**	**A#**	**B**	**C**	**C#**	**D**	**D#**	**E**	**F**	**F#**	**G**	**G#**
	la	**la#**	**si**	**do**	**do#**	**ré**	**ré#**	**mi**	**fa**	**fa#**	**sol**	**sol#**
	440	466.16	493.88	523.25	554.37	587.33	622.25	659.26	698.46	739.99	783.99	830.61

{ <*valeur2*> | <*variable2*> } temporisation (de 1 à 10) après la note ou nom de la variable contenant la valeur de la temporisation après la note.

Exemple :

```
130 SOUND 440, 2.5
140 SOUND "C2#", Tempo
```

Equivalent TI58C :

2 . 5 X/T 1 3 0 4 SND
RCL 03 X/T 1 5 0 3 6 7 SND

STOP

arrêt du programme.

Syntaxe :

STOP

Exemple :

130 STOP

Equivalent TI58C :

R/S

SWAP

échange de données.

Syntaxe :

SWAP <*variable*> , <*variable*>

Exemple :

```
 40 SWAP VALEUR1,VALEUR2
 50 SWAP A(IND),B(IND)
```

Equivalent TI58C :

```
 RCL 16 EXC 15 STO 16

// ===== A(n) =====
 LBL COS ( CE + 2 ) STO 00 RTN
// ===== B(n) =====
 LBL SIN ( CE + 7 ) STO 00 RTN

 ( RCL 17 SBR COS RCL 00 STO 01 )
 ( RCL 17 SBR SIN RCL 00 STO 02 )
 RC* 02 EX* 01 ST* 02
```


Fonctions

Les fonctions sont utilisables dans les expressions d'affectations (LET)

Basic		TI58C
ABS(...)	valeur absolue d'un nombre.	IXI
ASC(...)	retourne le code PC100	x x
ATN(...)	arctangente d'un nombre	INV TAN
CHR(...)	retourne le code PC100	x x
COS(...)	cosinus d'un angle	COS
EXP(...)	e (base des logarithmes népériens) élevé à une puissance	INV LNX
FRAC(...)	partie décimale d'un nombre	INV INT
INT(...)	partie entière d'un nombre.	INT
LOG(...)	logarithme népérien d'un nombre	LNX
RND(...)	nombre aléatoire. ([<mini> [, <maxi>]])	<mini> X/T <maxi> RND
SGN(...)	change le signe d'un nombre	+/-
SIN(...)	sinus d'un angle.	SIN
SQR(...)	racine carrée d'un nombre	SQR
TAN(...)	tangente d'un angle.	TAN

INKEY (*variable*)

attente de frappe de caractère.

Syntaxe :

<*variable*> = { INKEY | INKEY$ }

Exemple :

```
 30 monchoix = INKEY
```

Equivalent TI58C :

```
KEY STO 03
```


DATE (*variable*)

date du jour.

Syntaxe :

{ DATE | DATE$ } (["<*valeur*>"])

<*valeur*> peut être :

yyyyy			
yyyymm	yyyy.mm	yyyy/mm	yyyy-mm
yyyymmdd	yyyymm.dd	yyyymm/dd	yyyymm-dd
yyyy.mmdd	yyyy/mmdd	yyyy-mmdd	
mmddyyyy	mmdd.yyyy	mmdd/yyyy	mmdd-yyyy
ddmmyyyy	ddmm.yyyy	ddmm/yyyy	ddmm-yyyy

Exemple :

```
230 LET A = DATE()
250 PRINT DATE$("yyyy-mm")
```

Equivalent TI58C :

```
1 3 NOW STO 05
1 2 NOW PRT
```


TIME (*variable*)

heure courante.

Syntaxe :

{ TIME | TIME$ } (["<*valeur*>"])

<*valeur*> peut être :

hh	mm	ss	
hhmm	hh.mm	hh:mm	hh-mm
hhmmss	hhmm.ss	hhmm:ss	hhmm-ss
hh.mmss	hh:mmss	hh-mmss	

Exemple :

```
230 LET A = TIME()
250 PRINT TIME$("hh.mmss")
```

Equivalent TI58C :

```
 2 5 NOW STO 05
 2 7 NOW PRT
```


PI (*constante*)

PI.

Syntaxe :

PI

Valeur : 3.141592653589

Exemple :

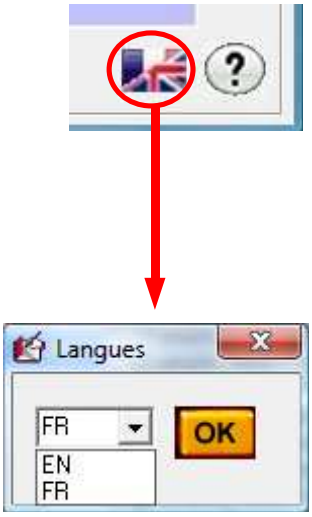
```
230 LET A = PI
250 PRINT PI
```

Equivalent TI58C :

```
PI STO 05
PI PRT
```


Langues

Dans le programme Bax58c il est possible de changer de langue.
En standard 2 fichiers sont fournis : Bax58cFR.lan (français) et Bax58cEN.lan (anglais) mais vous pouvez créer votre propre fichier langue Bax58cXX.lan en traduisant un des fichiers existants.



Paramètres spéciaux SHIFT+ALT+I

Les paramètres de configuration du compilateur Bax58C peuvent être stockés

- soit dans un fichier **Bax58C.ini** local (répertoire de l'application),
- soit dans un fichier **Bax58C.ini** dans le répertoire Windows (systemroot),
- soit dans le registre de Windows.

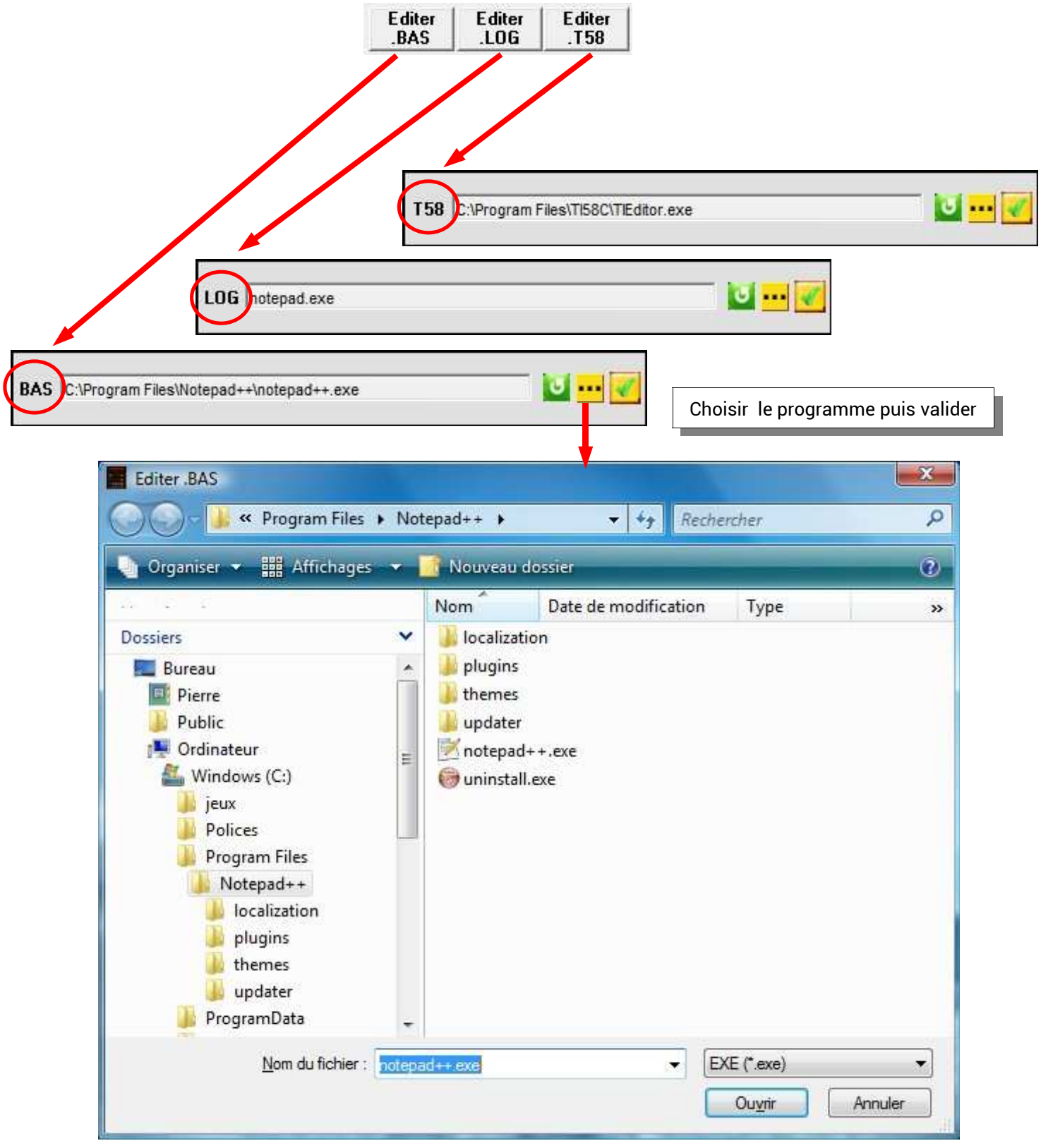
⇒ Choisir ***fichier INI local*** supprime le fichier **Bax58C.ini** du dossier Windows et crée le fichier local **Bax58C.ini**,
⇒ Choisir ***fichier INI Windows*** supprime le fichier **Bax58C.ini** du dossier de l'application et crée le fichier Windows **Bax58C.ini**,
⇒ Choisir ***le registre*** supprime le fichier **Bax58C.ini** (local ou Windows) et crée les paramètres dans le registre.

Cliquer sur le bouton “Backup” permet de faire une sauvegarde des paramètres dans

- un fichier **Bax58C***aaaammjjhhmmss*.**ini** pour les paramètres généraux

Choix des éditeurs

Les fichiers manipulés par le compilateur **Bax58C** peuvent être édités avec des programmes externes tels que *Notepad*.
Pour chaque type (**.BAS**, **.LOG**, **.T58**) l'éditeur utilisé peut être paramètré en cliquant, avec le bouton droit de la souris, sur le bouton "*Editer*" de chaque type.



A propos

affiche le numéro de version.

Programme : Calcul de e

Programme eFR.bas / eFR.t58

[E] initialisation et premier calcul [A] calcul suivant

$$e = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \ldots\ldots + \frac{1}{n!} + \ldots$$

```
================================================
  Compilateur BAX58C  - 30 / 09 / 2020 10:10:04
================================================
  C:\ProgramData\TI58C\Basic\e\eFR.bas
================================================
  100 REM ===== CALCUL DE E =====
  110 E:
  120 GOSUB TRAITS
  130 PRINT "     CALCUL DE E    "
  140 GOSUB TRAITS
  150 LET A = 1
  160 ENCORE:
  170 GOSUB BOUCLE
  180 PRINT A, E
  190 GOSUB TRAITS
  200 STOP
  210 A:
  220 LET A = A + 1
  230 GOTO ENCORE
  240 REM ===== CALCUL =====
  250 BOUCLE:
  260   LET E = 1
  270   FOR I=1 TO A
  280     LET FACT = 1
  290     FOR J=1 TO I
  300       LET FACT = FACT * J
  310     NEXT J
  320     LET E = E + 1 / FACT
  330   NEXT I
  340 RETURN
  350 REM ===== LIGNE DE EGAL =====
  360 TRAITS:
  370   PRINT "===================="
  380 RETURN
  390 END
================================================
                  eFR.t58
================================================
              Steps     :   161
              Registers :   8
================================================
              R E G I S T E R S
            Basic Name      Reg #
            ----------      -----
            Reserved          00
            Reserved          01
            Reserved          02
            A             =>  03
            E             =>  04
            I             =>  05
            FACT          =>  06
            J             =>  07
================================================
                 L A B E L S
            Basic Addr      TI58 LBL
            ----------      --------
            E             =>   E
            A             =>   A
            TRAITS        =>   COS
            ENCORE        =>   SIN
            BOUCLE        =>   TAN
            270           =>   DEG
            290           =>   GRD
================================================

//  ===== CALCUL DE E =====
LBL E
SBR COS
//      CALCUL DE E
HIR 00 OP 00
1 5 1 3 2 7 1 5 4 1 OP 02
2 7 0 0 1 6 1 7 0 0 OP 03
1 7 0 0 0 0 0 0 0 0 OP 04
OP 05 HIR 10

SBR COS
1 STO 03
LBL SIN
SBR TAN
RCL 03 PRT RCL 04 PRT
SBR COS
R/S
LBL A
OP 23
GTO SIN
//  ===== CALCUL =====
LBL TAN
1 STO 04
1 STO 05 LBL DEG
1 STO 06
1 STO 07 LBL GRD
( RCL 06 * RCL 07 ) STO 06
1 SUM 07 RCL 07 X/T RCL 05 GE GRD
( RCL 04 + 1 / RCL 06 ) STO 04
1 SUM 05 RCL 05 X/T RCL 03 GE DEG
RTN
//  ===== LIGNE DE EGAL =====
LBL COS
// ====================
HIR 00 OP 00
6 4 6 4 6 4 6 4 6 4 OP 01
OP 02
OP 03
OP 04
OP 05 HIR 10

RTN
R/S
================================================
```


Programme : Suite de Fibonacci

Programme fiboFR.bas / fiboFR.t58

[A] initialisation , saisie rang [R/S] —> calcul

```
================================================
  Compilateur BAX58C  - 30 / 09 / 2020 10:11:58
================================================
  C:\ProgramData\TI58C\Basic\Fibonacci\fiboFR.bas
================================================
  110 REM ===== FIBONACCI =====
  120 A:
  130 INPUT "Rang ?";RANG
  140 IF RANG > 49 GOTO A
  150 LET LARG = 0
  160 LET LONG = 1
  170 FOR I=1 TO RANG-1
  180   LET RESERV = LONG + LARG
  190   LET LARG = LONG
  200   LET LONG = RESERV
  210 NEXT I
  220 GOSUB TIRETS
  230 PRINT "N ";RANG
  240 PRINT "F(N)";LONG
  250 GOSUB TIRETS
  260 PRINT
  270 STOP
  280 B:
  290 LET RANG = RANG + 1
  300 GOTO 140
  310 TIRETS:
  320   PRINT "--------------------"
  330 RETURN
  340 END
================================================
                 fiboFR.t58
================================================
              Steps     :   155
              Registers :   8
================================================
              R E G I S T E R S
            Basic Name      Reg #
            ----------      -----
            Reserved         00
            Reserved         01
            Reserved         02
            RANG         =>  03
            LARG         =>  04
            LONG         =>  05
            I            =>  06
            RESERV       =>  07
================================================
                 L A B E L S
            Basic Addr      TI58 LBL
            ----------      --------
            A            =>   A
            B            =>   B
            140          =>   COS
            170          =>   SIN
            TIRETS       =>   TAN
================================================

//  ===== FIBONACCI =====
LBL A
OP 00  3 5 1 3 3 1 2 2 0 0 OP 03
7 1 0 0 0 0 0 0 0 0 OP 04
OP 55 R/S
STO 03
LBL COS
RCL 03 X:T 4 9 INV GE A
0 STO 04
1 STO 05
1 STO 06 LBL SIN
( RCL 05 + RCL 04 ) STO 07
RCL 05 STO 04
RCL 07 STO 05
1 SUM 06 RCL 06 X/T  ( RCL 03 - 1 ) GE SIN
SBR TAN
OP 00  3 1 0 0 OP 04 RCL 03 OP 06
OP 00  2 1 5 5 3 1 5 6 OP 04 RCL 05 OP 06
SBR TAN
ADV
R/S
LBL B
OP 23
GTO COS
LBL TAN
// --------------------
HIR 00 OP 00
2 0 2 0 2 0 2 0 2 0 OP 01
OP 02
OP 03
OP 04
OP 05 HIR 10

RTN
R/S
================================================
```


Avertissement aux lecteurs

Les informations contenues dans cet ouvrage sont données à titre indicatif et n'ont aucun caractère exhaustif voire certain. A titre d'exemple non limitatif, cet ouvrage peut vous proposer une ou plusieurs adresses de sites Web qui ne seront plus d'actualité ou dont le contenu aura changé au moment où vous en prendrez connaissance.

Aussi, ces informations ne sauraient engager la responsabilité de l'auteur.

L'auteur ne pourra être tenu responsable de toute omission, erreur ou lacune qui aurait pu se glisser dans ce livre ainsi que des conséquences, quelles qu'elles soient, qui résulteraient des informations et indications fournies ainsi que de leur utilisation.

Les produits cités dans cet ouvrage sont protégés, et les marques déposées par leurs titulaires de droits respectifs. Cet ouvrage n'est ni édité, ni produit par le(s) propriétaire(s) de(s) programme(s) sur le(s)quel(s) il porte et les marques ne sont utilisées qu'à seule fin de désignation des produits en tant que noms de ces derniers.