Langage Basic CASIO

Bonjour tout le monde !

Dans ce tutoriel, je vais vous apprendre à programmer en langage Basic Casio (ou sur calculatrice Graphique si vous préférez). Vous allez tout apprendre en partant de zéro : comme si vous n'aviez encore jamais allumé une telle calculatrice auparavant .
Ce langage est très simple et donc par conséquent très limité (aussi car le processeur de la calculatrice n'est pas très puissant. Il ne fallait pas vous attendre à un Dual Core de 3 GHz): il ne vous permettra pas de faire des jeux (ou programmes) comparables à ceux que vous auriez pu faire en C (ou avec un autre langage de programmation plus puissant). Cependant, le Basic Casio vous permettra de prendre les bonnes habitudes d'un programmeur : il est un bon échauffement avant l'apprentissage des autres langages .
Sur ce, je vous souhaite BONNE CHANCE !

Partie 1 : Les premières bases

Lorsque vous aurez lu tout ce chapitre, vous serez alors déjà capable de créer certains jeux conséquents (malheureusement n'utilisant pas encore les graphismes mais ça viendra, ne vous inquiétez pas). Nous allons donc voir des petites fonctions très simples et très pratiques .
Si vous connaissez déjà un peu la programmation (un autre langage), vous trouverez sûrement certaines similitudes : ça vous aidera .

Le tout début

Voici expliquées dans cette petite partie les toutes premières bases de la programmation : le matériel à utiliser, les bonnes habitudes à prendre...

Quel matériel ?

Nous voilà lancés, ... Donc, par définition, vous aurez besoin d'une calculatrice. Or le problème est là : toutes les calculatrices n'ont pas le même langage de programmation. Pour ce faire, je vais vous demander de posséder une calculatrice graphique de marque Casio (les vertes). Mais ce n'est pas fini : tous les modèles ne correspondent pas avec ce langage de programmation (complètement ou entièrement). Je vous recommande donc les modèles suivant : Graph 35, Graph 35+, Graph 65, Graph 85, Graph 85 SD, Graph 100, Graph 100+, ... D'autres calculatrices telles que la Graph 25 et la Graph 25+ pourraient convenir mais elles sont trop limitées (la plupart des fonctions que je vous présenterai, n'existeront pas sur cette calculatrice). Je vous la déconseille donc .

Attention, les chemins d'accès des fonctions pourront varier selon les différents modèles. Ceux que je vous indiquerai seront forcément compatibles avec la Graph 65 (car je la possède) et la Graph 35+ (car c'est quasiment le même modèle que la 65). Normalement ce problème ne se posera pas mais il peut exister . Depuis peu, je possède aussi la 85 donc tous les chemins d'accès que vous donne seront aussi compatible avec cette calculatrice.
Un autre problème pourra se poser : il se peut que sur certaines calculatrices certaines fonctions ou instructions n'existent pas ... Pas de panique : envoyez moi un MP (à Ilae) et nous verrons une solution ensemble.

Vous aurez aussi besoin de votre manuel de programmation. Ceci car à la fin de ce dernier se trouve un annexe des fonctions avec leur chemin d'accès. Normalement, je vous les donnerai mais si jamais vous l'oubliez, cet annexe peut se révéler utile .
Voilà, avec ça, vous devriez pouvoir commencer à programmer.

Un programme ? Comment en créer un ?

Un "programme", quel grand mot... Et pourtant pour en créer un nous n'avons pas besoin de capacités hors du commun ; c'est "simplissime" .
Voici donc la démarche à suivre : lorsque votre calculatrice est allumée et que vous êtes sur l'écran principal, grâce aux flèches de direction, mettez en surbrillance l'icône située en bas à gauche de l'écran (PRGM) et appuyez sur [EXE] : vous vous trouverez alors dans le menu programme. Vous pouvez aussi appuyer sur la touche log (ou B) pour vous rendre dans ce menu. Ceci fait vous n'aurez plus qu'à appuyer sur la touche [F3] (NEW) pour créer un nouveau programme, [F2] (EDIT) pour éditer un programme déjà existant ou [F1] (EXE) (ou sur la touche [EXE])pour exécuter un programme déjà existant. Le programme qui devra subir ces actions (exécuter ou éditer) devra être en mis en surbrillance (grâce aux flèches de direction) avant de presser les touches [F2] ou [F1]. Lorsque vous voulez créer un nouveau programme en appuyant sur [F3], peu importe le programme déjà existant en surbrillance.
Si jamais votre calculatrice ne contient pas de programme en mémoire, un tel message s'affichera : "No Programs" et vous ne pourrez alors que presser la touche F3 (NEW).

Note : si jamais vous craignez le plagiat ou même que quelqu'un s'amuse à modifier le code source du programme, vous pouvez le protéger. Et oui en effet, les Casio Graph xx possèdent un système de verrouillage du code source des programmes.
Comment faire ?

C'est assez simple, lorsque vous créez votre programme et que vous entrez le nom, appuyez sur la touche [F5] (représentée sur l'écran par une petite clé) et choisissez un mot de passe puis validez. Ce mot de passe vous sera redemandé à chaque fois que vous voudrez modifier le code ou envoyer un programme (via un câble) à une autre CASIO ou à un ordinateur.
Toutefois, je vous déconseille l'utilisation de ce système car lorsque l'on crée un programme, on a parfois besoin de le tester pour se rendre compte s'il est bien codé ou non (même bien souvent). Si jamais vous mettez un mot de passe et que vous testez, vous ne pourrez pas situer l'erreur dans le code. Alors que sans mot de passe, la calculatrice vous indique l'erreur (en plaçant le curseur sur le caractère suivant).
Pour que la calculatrice vous indique l'erreur, il faut, une fois que le message d'erreur apparaît que vous appuyiez sur une des flèches de direction.

C'est bon, nous sommes prêts à commencer : il n'y a plus qu'à se lancer...
Que ceux qui se sentent prêts me suivent... dans le prochain chapitre où les réjouissances commenceront .

Des Chiffres et des Lettres

Non non, nous n'allons pas créer ce célèbre jeu : ce serait trop compliqué alors que nous ne savons encore rien. Je vais vous apprendre le fonctionnement des lettres et celui des chiffres.

Comment afficher du texte ?

Comment afficher du texte ? Quelle utilité me direz vous ? Eh bien si, dans n'importe quel programme que vous ferez, vous aurez besoin d'afficher du texte : que ce soit une aide, un menu, des indications, ... : vous en aurez obligatoirement besoin alors écoutez bien (enfin lisez bien).
Lorsque vous avez besoin d'afficher du texte, il vous suffit de le mettre entre guillemets. Pour ce faire, il y a deux manières :
- la combinaison de deux touches : [Alpha] puis [F2](") (sur la Graph 85(SD), c'est : [Alpha] puis [x10^x](").
- la pression de [F6] (SYBL) puis celle de [F2] (").
Note : les guillemets d'ouverture et de fermeture sont identiques : il suffit donc de presser la même touche pour fermer et ouvrir les guillemets.
Un petit exemple pour comprendre :

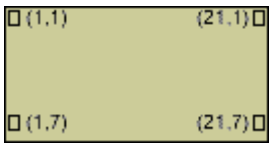
Code : Basic CASIO
```
"BONJOUR"
```

Voilà, lorsque vous exécuterez ce code, le mot BONJOUR s'affichera en haut à gauche. Il est possible que quelque chose d'autre soit affiché à l'écran : normal, vous n'avez pas demandé à la calculatrice d'effacer l'écran avant de débuter votre programme (eh oui, certaines choses restent affichées à l'écran). Pour éviter cela, facile : il vous suffit de mettre juste avant les guillemets, une fonction effaçant l'écran. Cette fonction s'appelle : ClrText. Voici son chemin d'accès : [Shift], [VARS] (Prgm), [F6], [F1] (Clr), [F1] (Text).Il suffit donc de mettre :

Code : Basic CASIO
```
ClrText
"BONJOUR"
```

Lorsque je vais à la ligne entre chaque instruction, je presse la touche [EXE] : faites de même. Cela permettra à la calculatrice de comprendre que l'instruction s'arrête là. Vous pourriez mettre aussi deux points (:) ([Shift], [VARS] (Prgm), [F6], [F5] (:))mais je vous conseille le retour à la ligne car il permet une meilleur lisibilité du programme. Par la suite, nous verrons une troisième fonction ...

Voilà pour l'affichage du texte mais on peut mieux faire : pourquoi ne pas afficher un mot au milieu de l'écran ? Ce serait mieux ... Pour ce faire, une fonction existe : Locate. Voici son chemin d'accès : [Shift], [VARS] (PRGM), [F6], [F4] (I/O), [F1] (Lcte) .Elle prend en compte deux paramètres : en premier l'abscisse du point d'où débutera le texte et en second, l'ordonnée du point d'où débutera le texte. L'abscisse doit être comprise entre 1 (partie gauche de l'écran) et 21 (partie droite de l 'écran). L'ordonnée doit être comprise entre 1 (partie haute de l'écran) et 7 (partie basse de l'écran). Voici une image indiquant les coordonnées des coins, pour mieux comprendre :



Nous allons donc afficher le mot BONJOUR au centre de l'écran.

Code : Basic CASIO
```
ClrText
Locate 8,4,"BONJOUR"
```

Voilà, la calculatrice va d'abord effacer l'écran puis écrire le mot BONJOUR à la position 8,4 (dans notre exemple, le centre de l'écran).
C'est bon, maintenant, nous savons afficher du texte à l'écran. Cool non ? Rassurez moi, vous n'avez pas trouvé ça trop dur ? N'est ce pas ?

Les variables.

Nous voilà dans un des chapitres les plus importants en programmation, les variables... Déclarer une variable est une action très simple. Pour ce faire, il existe une touche située au dessus du bouton Ac/On, sur le clavier de la calculatrice. Une flèche est représentée sur cette touche. La voyez vous ? Bon, elle est bien belle cette flèche mais toute seule, elle ne sert pas à grand chose . Il existe 28 variables (les 26 lettres de l'alphabet, le rhô et le thêta). La syntaxe pour déclarer une variable est : <valeur> -> <variable>.
Je représente la flèche comme ceci : ->.
Un exemple : donner la valeur 10 à la variable D.

Code : Basic CASIO
```
10->D
```

Évident n'est-ce pas ?
Pour donner la même valeur à plusieurs variables se suivant dans l'alphabet, il suffit d'utiliser ce signe là : ~ . Son chemin d'accès est : [F6] (SYBL), [F3] (~).
A chaque début de programme, il faut remettre toutes les variables à zéro. Comme ceci :

Code : Basic CASIO

0->A ~ Z
0->r
0->θ

Ce n'est pas indispensable mais c'est conseillé... sinon vous risquez d'avoir des surprises.

Pour afficher la valeur d'une variable, il suffit d'écrire le nom de la variable.
Pour demander à l'utilisateur de donner lui même une valeur à une variable durant l'exécution d'un programme, il suffit d'utiliser la syntaxe suivante : ? -> A . Le chemin d'accès du point d'interrogation est : [Shift], [VARS] (Prgm), [F4] (?).

Comme je vous l'ai déjà dit, les chemins d'accès que je vous donne peuvent varier d'un modèle à l'autre. Par exemple sur la Graph 100/100+, le point d'interrogation a le chemin d'accès suivant : [Shift], [VARS] (Prgm), [F3] (?). Pour de plus amples informations reportez-vous à votre manuel.

Voilà nous pouvons déjà faire un petit programme demandant l'âge de l'utilisateur et le lui redonnant. On essaie ?
Il suffira de demander à l'utilisateur d'entrer son âge, de l'enregistrer dans une variable et de le redonner après. Essayez de le faire vous-même. Je vais vous mettre la solution ci-dessous.
Bonne chance !

Code : Basic CASIO
```
0->A
ClrText
"Quel est votre age":?->A
"Votre age = ":A
```

Voilà, c'est bon !
Note : lorsque je vais à la ligne, c'est comme si j'appuyais sur [EXE]. Chez vous, ça vous affichera en fin de ligne la flèche avec un angle droit et puis ça ira à la ligne.
Bon ben, voilà, vous savez tout sur les variables.

Un peu de calcul...

Je sais bien que la plupart d'entre vous détestez le calcul, mais en programmation, pour faire un programme conséquent, il vous faudra obligatoirement faire un peu de calcul. Mais rassurez-vous, ce n'est pas bien dur. Dans ce chapitre, nous ne verrons que de très simples calculs.
Nous allons donc commencer par les opérations de base : addition, soustraction, multiplication et division . J'espère que vous les connaissez et ne vous expliquerai donc pas leur fonctionnement . Effectuez un calcul dans un programme (ou à l'extérieur) est très simple : il vous suffit d'écrire votre calcul et le tour est joué . Le seul problème est qu'une fois le calcul effectué (ce que la calculatrice fait quasiment instantanément), la calculatrice oublie le résultat. Normal, vous ne lui avez pas demandé de le retenir. Pour lui demander de le retenir, c'est très simple (comme tout), il suffit de donner le résultat à une variable. Cela donnera donc :

Code : Basic CASIO
```
9+6->A
```

La variable A prendra donc la valeur 15 (9+6). Et ceci marche avec tous les types d'opérations. Mais les capacités de notre chère calculatrice (bien qu'elles soient très limitées), ne s'arrêtent pas là . Elle peut aussi faire des calculs à partir de variables. Ça donne :

Code : Basic CASIO
```
9->A
5->B
A-B->C
```

La variable C vaudra donc 4 (9-5). Pour l'instant, toutes ces histoires de calculs peuvent vous paraître inutiles mais vous verrez qu'au contraire elles sont indispensables .
Bon, pour vérifier que vous avez bien compris, nous allons faire un autre petit programme : une calculatrice n'effectuant que des additions (c'est déjà pas mal). Le programme vous demandera d'entrer le nombre A, puis le nombre B, il les additionnera et vous donnera le résultat. A vous de jouer, je vous donne la réponse ci-dessous.

Code : Basic CASIO
```
0->A~Z
ClrText
"Nombre A":?->A
"Nombre B":?->B
A+B->C
C
```

Évident, n'est-ce pas ? C'est l'application directe de ce chapitre. Si vous n'arrivez pas à faire ça, il vous faut le relire, sinon, vous ne pourrez pas suivre. Et voilà une première marche d'enjambée : pas la plus dure mais ça s'en approche : il faut se mettre dans le bain. Une fois lancé, ça va tout seul. En tout cas, si vous avez compris ça, vous partez sur de bonnes bases : sans ça, vous ne pouvez rien faire. Et puis vous voyez, ce n'est pas bien difficile, vous voyez ce qu'on arrive à faire avec seulement quelques petites fonctions... Ca ne vous donne pas de l'espoir ???

Les boucles et les conditions

Voici, je crois, le chapitre le plus intéressant. C'est grâce à celui là que vous serez capables de faire de véritables petits jeux. Il existe trois grands types de boucles que vous découvrirez dans ce chapitre. Lorsque vous maîtriserez bien ces trois types de boucles, vous pourrez déjà créer des petits jeux marrants.

If, Then, Else, IfEnd

Voici la plus simple des trois et (je pense) la plus utile : les conditions (If, Then, Else, IfEnd).

If, Then, IfEnd

Elle est très simple à utiliser car elle est logique. Je vous indique donc la syntaxe :

If <condition>
Then <actions à exécuter si la condition est respectée>
IfEnd

Simple ? non ? un petit exemple concret alors ... Ce petit exemple sera en fait un programme. La calculatrice vous demandera votre âge et vous dira si vous êtes majeur ou mineur ...

Code : Basic CASIO
```
"Quel est votre age":?->A
If A>=18
Then "Vous êtes majeur"
If End
If A<18
Then "Vous êtes mineur"
If End
```

Le signe ">=" veut dire "supérieur ou égal". Le chemin d'accès de ces signes est : [Shift], [VARS] (Prgm), [F6], [F3] (REL). Et voilà, tous vos symboles sont ici ! 

Le chemin d'accès des fonctions "If", "Then" et "IfEnd" est : [Shift], [VARS] (Prgm), [F1] (COM).
Toutes les fonctions sont là : If , Then, Else (une fonction que nous verront à la fin de cette partie) et I-End (IfEnd)

Pas mal comme fonctions ... n'est ce pas ? c'est pratique . Mais les programmeurs ne les ont pas encore trouvées assez pratiques, ils en ont donc créé une autre : Else.

If, Then, Else, IfEnd

Je ne vais pas vous faire un beau discours, ce serait trop compliqué, je vais vous donner la syntaxe, ce sera plus facile à comprendre (et plus facile à expliquer) :

If <condition>
Then <actions à exécuter si la condition est respectée>
Else <actions à exécuter si la condition n'est pas respectée>

Par Ilae pour planete-casio.fr

IfEnd

Pour bien comprendre, nous allons reprendre le même exemple que précédemment. Au lieu qu'il y ait deux conditions différentes, il n'y en aura qu'une !

Code : Basic CASIO
```
"Quel est votre age":?->A
If A>=18
Then "Vous etes majeur"
Else "Vous etes mineur"
If End
```

Ben voilà, c'est très simple, non ?

Une petite question : comment puis je me souvenir du nom de chaque fonction ?

Vous connaissez un peu d'anglais ?
Et bien dans ce cas, vous devez savoir que "If" veut dire "si" (c'est le début de la condition), "Then", "puis" (ce qu'il y a à exécuter si la condition est respectée), "Else", "sinon" (ce qu'il y a à exécuter si la condition n'est pas respectée) et "IfEnd" , "fin de la condition".
N'oubliez jamais de fermer la condition (IfEnd). Sinon, les fonctions suivantes ne s'exécuteront que si les conditions sont respectées (Then) ou si elles ne le sont pas (Else).

Voilà vous connaissez le premier groupe de ces trois grands groupes de fonctions. Dans la prochaine partie, vous verrez une fonction (n'existant pas directement sur Graph 100/100+) résumant cette boucle ! Elle est très utilisée.

---

code de saut (=>)

Cette fonction n'existe pas directement sur la Graph 100/100+ mais la calculatrice est capable de l'interpréter. Si vous la souhaitez, il faut que vous l'importiez depuis un ordinateur ou une autre calculatrice. Nous verrons la procédure de transfert en Annexe.

Je vous préviens tout de suite, si vous ne connaissez pas cette fonction, vous ne comprendrez jamais les programmes des autres: c'est une des plus utilisées. Mais bon, si vous ne la connaissez pas, vous pourrez quand même programmer sans problème : c'est juste un raccourci de la condition vu précédemment .
Voilà la syntaxe de cette fonction :
<condition> => <action à exécuter si la condition est respectée>
Les conditions et les fonctions à exécuter sont les mêmes que précédemment. Nous allons donc encore reprendre le même exemple :

Code : Basic CASIO
```
"Quel est votre age":?->A
A=>18 => "Vous etes majeur"
A<18 => "Vous etes mineur"
```

Le chemin d'accès de la double flèche est : [Shift], [VARS] (Prgm), [F3] (jump), [F3] (=>).

Vous pouvez mettre plusieurs conditions de suite comme ça:
<condition 1> => <condition 2> => <condition 3> => <action à exécuter>
Si la condition 1 est respectée, il passera à la condition 2 ... et ainsi de suite jusqu'à ce que l'on arrive à l'action à exécuter une fois que toutes les conditions auront été vérifiées .
Voilà, c'était court et simple. Et croyez moi, vous utiliserez beaucoup cette fonction: elle est très pratique  .

Do/LpWhile et While/WhileEnd

En fait, ce n'est pas une boucle que nous allons voir mais deux qui sont assez similaires. Un peu dur à assimiler au début, elles sont en réalité très simples et se révéleront fortes utiles. Si vous avez des petits problèmes dans cette partie, ce n'est pas bien grave : il vous suffira de la relire. Surtout ne l'ignorez pas : malgré le fait qu'elle soit remplaçable par une combinaison d'autres fonctions, elle est très importante.
Ces boucles sont chacune composées de deux fonctions : l'ouverture de la boucle et la fermeture de la boucle. Ce qui différencie ces fonctions est l'endroit où est placé la condition pour en sortir : au début ou à la fin. Et oui, ces fonctions feront en sorte que la boucle s'exécutera tant que la condition sera vraie.

La première : While, WhileEnd

Je vais directement vous mettre la syntaxe puis un exemple, c'est ce qui sera le plus facile :

While <condition>
<actions à exécuter si la condition est respectée>
WhileEnd

Le chemin d'accès des fonctions "While" et "WhileEnd" est : [Shift], [VARS] (Prgm), [F1] (COM), [F6], [F6].
Toutes les fonctions sont là : Whle (While), WEnd (WhileEnd), Do et Lp-W (Lp-While) (deux fonctions que nous verront à la fin de cette partie).

Voici donc l'exemple pour bien comprendre.
Un petit peu d'histoire  .

Code : Basic CASIO
```
0->A
While A=|800
ClrText
"En quelle année, Charlemagne est il devenu empereur ":?->A
WhileEnd
"Bravo, vous avez trouvé"
```

Par Ilae pour planete-casio.fr

"=|" veut dire "différent", c'est le symbole mathématique (égal barré) ; il se trouve dans la calculatrice avec les autres symboles mathématiques : [Shift], [VARS] (Prgm), [F6], [F3] (REL), [F2].

Une petite explication ou vous avez compris ? Je vais quand même la donner à ceux qui n'ont pas compris :

- on initialise la variable A.
- on énonce la condition à l'entrée de la boucle : la boucle s'exécutera tant que A sera différent de 800.
- on efface l'écran
- on pose la question et on demande à l'utilisateur d'entrer la valeur dans la variable A.
- la fin de la condition.
- le message vous disant que vous avez trouvé.

Tant que A sera différent de 800, "la boucle tournera en boucle" (c'est la cas de le dire ) : elle vous posera toujours la question. Si jamais A vaut 800, la calculatrice sortira de la boucle et ira à la fonction suivante.

La seconde : Do, LpWhile

Personnellement, je préfère celle-là, je la trouve meilleure car la condition est à la fin mais dès fois, il vaut mieux l'autre !
syntaxe :

Do
<actions à exécuter si la condition est respectée>
LpWhile <condition>

C'est exactement la même chose que précédemment sauf que la condition est à la fin de la boucle ; nous allons donc prendre le même exemple :

Code : Basic CASIO
```
0->A
Do
ClrText
"En quelle année, Charlemagne est il devenu empereur ":?->A
LpWhile A=|800
"Bravo, vous avez trouvé"
```

Je ne vous l'explique pas : c'est le même que précédemment.

Si jamais vous connaissez un peu le C, vous avez du remarquer que cette fonction existe aussi en C : c'est la même .

Faites bien attention : s'il existe deux boucles différentes, c'est qu'elles ont deux utilités différentes. En effet, avec While et WhileEnd, si la condition n'est pas respectée, les actions à exécuter ne seront jamais lues par la calculatrice. Or si vous utilisez Do et LpWhile, elles seront lues au moins une fois ... Adaptez donc en fonction de vos besoins.

Est ce que la aussi, je peux m'en rappeler grâce à l'anglais ? comme avec les conditions vu précédemment ?

Et bien oui : "While" veut dire "tant que". Ça donne donc : tant que A est différent de 800 (pour notre exemple) ... "WhileEnd" voudrait dire (si ça existait) "fin de tant que" donc fin de la boucle. Do veut dire "fais". Ça donne : fais marcher les fonctions qui suivent. Utile l'anglais, n'est ce pas ? Vous n'êtes pas encore convaincu ? Avec tout l'anglais qu'on risque de voir dans ces cours, vous devriez voir cette langue différemment ...  .

For, To, Step, Next

Cette boucle inclut en elle même un compteur ce qui est très utile lorsque vous voulez faire marcher les fonctions à l'intérieur de la boucle un certain nombre de fois. Et puis elle est simple à utiliser. Mais bon, si vous m'écoutez, tout est très simple ; c'est vrai qu'une fois compris, c'est toujours simple ...

For, To, Next

Cette fonction demande une petite chose : une variable et c'est tout.
comme d'habitude, la syntaxe :

```
For <la valeur de départ que vous donnez à la variable> To <la valeur d'arrivée de la variable>
<les actions à exécuter>
Next
```

Le chemin d'accès des fonctions "For", "To" et "Next" est : [Shift], [VARS] (Prgm), [F1] (COM), [F6].
Toutes les fonctions sont là : For , To, Step (une fonction que nous verront à la fin de cette partie) et Next

Je reconnais que comme ça, ça ne parait pas évident mais un petit exemple vous aidera à mieux comprendre : imaginez que vous ayez copier sur votre voisin et que vous ayez eu comme punition à copier 10 fois "Je ne dois pas tricher" et bien à la calculatrice, ce serait un jeu d'enfant :

Code : Basic CASIO

```
For 1->A To 10
"Je ne dois pas tricher"
Next
```

Par Ilae pour planete-casio.fr

Alors, les explications :

- la variable A prend comme valeur de départ 1 (For 1->A) et devra s'arrêter lorsqu'elle aura atteint la valeur 10 (To 10).
- la phrase s'affiche à l'écran.
- la variable A se voit augmenter sa valeur de 1 (Next).

La boucle s'exécutera alors jusqu'à ce que la variable A ait atteint la valeur 10 ; c'est à dire lorsqu'elle aura écrit 10 fois la phrase.
Voilà déjà un premier avantage à être programmeur ... non ? les punitions faites en 3 minutes, c'est pas beau ça ?  .

Est on obligé d'augmenter (avec la fonction Next) la valeur de 1 ? est ce qu'on pourrait l'augmenter de 2 par exemple ?

Ben on ne peut pas dire que vous n'êtes pas curieux. Mais de toute manière, cette question tombe à pic : c'est exactement ce que je veux vous montrer après. Il est donc possible de choisir le pas (et oui, c'est comme ça que ça s'appelle) pour augmenter la valeur différemment.

For, To, Step, Next

C'est exactement la même syntaxe, sauf que sur la ligne où l'on déclare la valeur de départ et celle d'arrivé, on rajoute le pas :
For <valeur de départ> To <valeur d'arrivée> Step <pas>
Je n'ai pas vraiment d'idée pour trouver un bon exemple ; on va donc en faire un débile.
Créons un programme qui compte de 2 en 2 en partant de 10 et en allant à 30 :
Pour se faire, je vais utiliser une nouvelle fonction stoppant le programme à chaque fois que l'opération aura été faite. Ceci car sinon, la calculatrice affichera les fonctions quasiment instantanément et vous n'aurez pas le temps de les voir ; ce sera même tellement rapide qu'elle même n'aura pas le temps de les afficher. Cette fonction est un petit triangle noir et est appelée Display ; son chemin d'accès est : [Shift], [VARS] (Prgm), [F5] . Comme ce petit triangle noir n'existe pas sur mon clavier, j'utiliserai ce symbole pour le signaler : "¤". Cette fonction mettra en pause le programme et tant que vous n'appuierez pas sur [EXE], il restera en pause. Retenez cette fonction, car c'est aussi une des plus utilisée.

Note : une fois cette fonction mise, votre curseur ira directement à la ligne : c'est normal. Je vous avais parlé d'instructions qui permettaient de séparer les différentes instructions (le retour à la ligne et les deux points). Voici donc la dernière ...

Code : Basic CASIO
```
For 10->A To 30 Step 2
A¤
Next
```

Pas besoin de vous expliquer ce programme, vous l'avez compris ?

Vous devez vous dire (vu les exemples que j'ai choisis) que cette fonction ne sert jamais hors vous vous trompez ; lorsque vous ferez des jeux, elle vous sera plus qu'utile.

Nous allions oublier notre petit cours d'anglais ?

Et bien "For 10->A" (pour notre premier exemple) "pour un donne A", "To 30" "à 30", "Step 2" "avec un pas de 2" et "Next" "continue". C'est sûr, c'est moins évident (un peu plus tiré par les cheveux) mais ça se tient quand même. Et bien voilà, avec tout ça, vous êtes déjà capable de faire des petits jeux et même des petits programmes mathématiques. C'est d'ailleurs ce que nous allons faire dans le chapitre suivant : un petit jeu très simple. Je vous donnerai aussi des idées de petits programmes pour que vous puissiez vous entraîner.

Un petit TP : le jeu du + ou -

Voilà, maintenant, vous possédez assez de compétences ( ) pour réaliser quelques petits jeux. Votre premier vrai programme sera donc le jeu du plus ou moins .

Le but ....

Presque tous les programmeurs (quelque soient les langages) sont passés par là : la réalisation du jeu du plus ou moins . Il est simple et résume tout ce que nous avons vu jusque là.
Le but est très facile à comprendre : la calculatrice tirera un nombre au hasard et vous devrez le deviner. Pour le deviner, vous devrez donner des nombres et la calculatrice vous dira si le nombre qu'elle aura tiré au sort est plus grand ou plus petit.
Prêt à vous lancer ? ...

Quelques indications avant de commencer

Avant tout, je voudrai vous donner quelques petites indications sans quoi vous n'arriverez à rien .

comment tirer un nombre au hasard ?

Et oui, vous ne le savez pas . Vous étiez prêts à vous lancer sans même savoir le commencement de votre programme ...
Je ne vous expliquerai pas comment fonctionne cette fonction car je ne veux pas vous embrouiller avec (c'est assez compliqué) et vous n'en aurez pas besoin. Je vous donnerai donc juste la syntaxe :

Int <le nombre de possibilités>Ran#+1-><la variable à laquelle vous voulez assignez ce nombre généré aléatoirement>.

Le chemin d'accès de la fonction "Int" : [Optn], [F6], [F4] (num), [F2] (Int)
Le chemin d'accès de la fonction "Ran#" : [Optn], [F6], [F3] (prob), [F4] (Ran#)

Je vais vous donner un petit exemple pour clarifier vos esprits .

Code : Basic CASIO
```
Int 1000Ran#+1 ->A
```

Voilà, si vous exécutez ce programme, la variable A prendra une valeur entière comprise entre 1 et 1000 .
Retenez bien ce code car dans beaucoup de jeux vous en aurez besoin et j'espère que vous créerez d'autres jeux d'ici le moment où je vous expliquerai ce code .

Quels groupements de fonctions utiliser ?

Vous pourriez en utiliser 3 types :
- If, Then, Ifend
- While, WhileEnd
- Do, LpWhile

Mais je vous déconseille la première (bien que ce soit possible) car il vous faudrait encastrer des boucles dans d'autres boucles, ce serait trop compliqué alors qu'avec les autres c'est très simple .

Quelles instructions utiliser ?

Et bien quasiment toutes celles que nous avons vu : l'affichage de texte, tout ce qui va avec les variables, la double flèche (vous pourriez utiliser la boucle de condition mais utiliser plutôt son raccourci), le petit triangle noir (lorsque vous afficherez des messages tels que "Bravo, vous avez trouvé". Ne l'utilisez pas pour vos messages tels que "+ GRAND" ou "+ PETIT".

Informations générales

Vous appellerez votre programme "MYST" (comme mystère).
La variable contenant le nombre mystère s'appellera X.
La variable contenant le nombre choisi par l'utilisateur s'appellera J.
Faites en sorte qu'il y ait 100 possibilités de solutions .
Si jamais vous y arrivez très facilement (ce qui devrait être le cas de tous), rajoutez un compteur qui comptera le nombre de coups que vous mettrez pour trouver le nombre mystère .
Si vous avez bien tout compris jusqu'à maintenant, créer un tel programme, ne devrait pas vous prendre plus d'une demie heure (Personnellement, j'en ai pour 5 min mais bon, je connais le principe par cœur et je connais le chemin d'accès de mes fonctions par cœur

(j'espère que vous aussi, commencez à les connaître).).

Voilà, je crois que je vous ai tout dit ... Ah non, j'allais oublier : Bonne Chance

---

Corrigé

Bon j'espère que vous avez réussi ... et surtout, ne regardez pas la solution si vous n'y êtes pas arrivé (sauf si vraiment, vous bloquez). Essayer plutôt de voir là où ça ne va pas et relisez le cours au cas où . Si jamais il y a un message d'erreur appuyer sur la flèche de gauche et la calculatrice vous enverra à l'endroit où s'est produit l'erreur : vous pourrez donc savoir pourquoi ça a buggé . Si vous ne trouvez pas, rendez vous à l'endroit du cours où l'utilisation de la fonction qui vous pose problème a été expliqué.
Voici donc la correction : j'ai fait le minimum, j'ai réduit mon programme au maximum (ou presque). Si vous avez le même code, c'est parfait . Si il est différent mais qu'il marche, c'est très bien aussi (c'est le but) mais essayez quand même de comprendre mon code .

Note : ce n'est pas forcément le code le plus condensé qu'il existe ... mais il est déjà pas mal .

Code : Basic CASIO
```
0->A~Z
ClrText
Int100Ran#+1->X
Do
"NOMBRE MYSTERE"?->J
J<X => "+ GRAND"
J>X => "+ PETIT"
LpWhile J=|X
"BRAVO VOUS AVEZ TROUVE"¤
```

Même si je vous l'ai déjà dit, je vous le redit : ¤ correspond au petit triangle noir et =| correspond au signe différent de (égal barré).

J'aurai pu insérer les deux points (:) entre le texte "NOMBRE MYSTERE" et le point d'interrogation (?) mais ils ne sont pas indispensable. Ils servent juste à permettre de faire afficher le point d'interrogation à la ligne.

Pour la correction de celui qui inclue le compteur, la voici :

Code : Basic CASIO
```
0->A~Z
ClrText
Int100Ran#+1->X
Do
"NOMBRE MYSTERE"?->J
J<X => "+ GRAND"
```

Par Ilae pour planete-casio.fr

```
J>X => "+ PETIT"
A+1->A
LpWhile J=|X
"BRAVO VOUS AVEZ TROUVE"
"NOMBRE DE COUPS=":A¤
```

A étant la variable servant de compteur . Si vous avez trouvé ça, vous deviendrez grâce à votre ingéniosité, un jour, un grand programmeur ! .

Conclusion

Bon, j'espère que vous avez tous réussi à créer un programme qui marche et que vous avez au moins compris mon code. Si tel est le cas, vous pouvez continuer les cours. Sinon il faut absolument que vous relisiez les passages flous sinon, vous allez être largués dans les prochains chapitres .
Bon comme promis, voici d'autres idées de programmes que vous pouvez réaliser au stade où nous en sommes :
- un petit théorème de Pythagore : dans un triangle rectangle, le carré de l'hypoténuse est égal à la somme des carrés des deux autres côtés . Ce programme pourra calculer le troisième coté si vous donner les deux autres . Attention, ce n'est pas évident .
- des petites questions (genre QCM) avec les réponses numérotées 1), 2), 3) ...
- une amélioration du jeu que nous venons de réaliser : ajout d'un compteur (si ce n'est pas déjà fait), un paramètre permettant de changer le nombre de possibilités, ajout d'une fonction permettant de jouer à deux ...
-et plein d'autres encore, trouvez en vous même ... mais bon, ça restera limité tant que nous n'auront pas vu les Labels et le getkey (les deux prochains chapitres). Après, vous serez capables de faire de bons programmes (sans graphismes malheureusement) ...

Les labels

Si vous avez déjà programmer dans un autre langage, ce mot peut vous être familier . Sinon, il doit vous être totalement étranger ... Dans ce cas, ne vous en faites pas car dans tous les programmeurs que je connais, tous les maîtrisent : quelques petites fautes au début car certaines parties ont été mal comprises mais elles sont vite éclaircies ...

Définition

Pour faire simple, nous allons commencer par la partie Anglais tout de suite, ça éclaircira un peu les choses . En anglais, "label" veut dire "étiquette" ... "étiquette", ça n'a rien a voir avec la programmation me diriez vous ... Et bien si mais je l'avoue, c'est un peu tordu . Je vais vous

expliquer : dans le code source d'un programme, il est possible de mettre des sortes d'étiquettes et de dire au programme, aller à l'étiquette tant . C'est surtout très utile lorsque c'est combiné avec les conditions .

Utilisation

Il est fort probable que cette définition ne vous ai pas bien aidé à comprendre . C'est assez normal, pour comprendre, il faut un exemple d'utilisation ainsi que quelques autres indications ...

Les fonctions

Il existe deux fonctions relatives aux labels : Lbl et Goto . Lbl est un raccourci de label (jusque là, ça va) et Goto vient (encore) de l'anglais et signifie (j'espère que vous l'avez compris) "aller à" (du verbe "go").Le chemin d'accès est : [Shift], [VARS] (Prgm), [F3] (JUMP) et [F1] (Lbl) ou [F2] (Goto) .

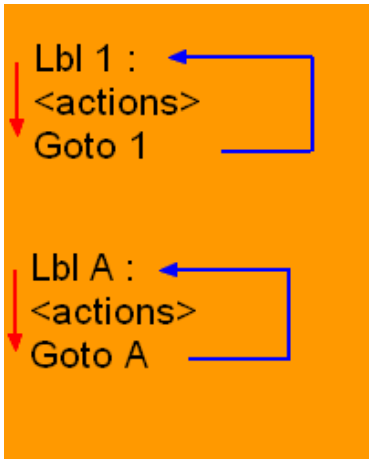
La syntaxe et les paramètres

Pour qu'un label marche, il lui faut un nom . Ceci pour qu'il puisse être déclaré et que l'on puisse dire "aller au label portant ce nom là" (c'est le rôle de la fonction Goto). Mais vous n'allez pas créer les noms vous même : ils sont précréés (si on peut appeler ça des noms). Il y en a 38 : les 26 lettres de l'alphabet, le rhô, le thêta et les 10 chiffres (de 0 à 9). Voici donc venir la syntaxe :

Lbl <nom du label>
<toutes les actions que vous voulez>
Goto <nom du label>

Et oui, c'est une sorte de boucle ...

Voici un petit schéma pour mieux comprendre :



Un exemple

Nous allons imaginer que nous entrons dans un restaurant, que nous prenions un choix et que la calculatrice nous affiche le prix :

Code : Basic CASIO
```
0->A~Z
Lbl 0
"Bonjour, que voulez vous ?"
"1) un steak"
"2) un poisson"
"3) une salade"
"4) une boisson"
?->A
A=1=>Goto 1
A=2=>Goto 2
A=3=>Goto 3
A=4=>Goto 4
A>4=>Goto 0
Lbl 1
"Vous avez choisi le steak"
"Ca vous fera 10 ?"¤
Goto 0
Lbl 2
"Vous avez choisi le poisson"
"Ca vous fera 8,5 ?"¤
Goto 0
Lbl 3
"Vous avez choisi la salade"
"Ca vous fera 6 ?"¤
Goto 0
Lbl 4
"Vous avez choisi la boisson"
"Ca vous fera 2 ?"¤
Goto 0
```

Bon voilà, c'est un code tout simple et sans grand intérêt mais ça devrait vous aider à comprendre . C'est vrai, nous aurions pu faire ça avec les fonctions If, Then, IfEnd mais l'avantage des labels est que si jamais vous devez à la fin de votre programme(par exemple) réexécuter ce que vous avez déjà exécuter dans des labels, il vous suffit de dire Goto <nom du label>. Vous comprendrez par la suite qu'il existe d'autres avantages à cette utilisation mais je vous laisse les découvrir tout seul car sinon, je vais vous embrouiller.

Attention

Les labels ont beau être très pratiques, ils sont très capricieux, il arrive qu'il plantent alors qu'il n'y a aucune erreur. Ceci arrive si votre code est trop long, s'il y a trop d'espace entre le Goto et le Label auquel le Goto renvoie, si vous utilisez beaucoup de labels ... Méfiez vous en : utilisez les modérément (et non comme moi au début ) et favorisez plutôt les boucles et les conditions vues dans le chapitre précédent, elles sont plus sûre .
Voilà un bon point de fait : avec le chapitre suivant, vous devriez bientôt être capable de faire des jeux conséquents . Pour vous entraîner à faire marcher les Labels, je vous propose de

refaire le jeu du plus ou du moins (le TP précédent) en remplaçant les Do et LpWhile (ou les While et WhileEnd) par des Labels . Si vous avez bien compris, c'est même plus facile à réaliser qu'avec les boucles que nous avons utilisé précédemment . Je vous met le code ci-dessous . Bonne chance !

Code : Basic CASIO
```
Lbl A
0->A~Z
ClrText
Int100Ran#+1->X
Lbl 0
"NOMBRE MYSTERE"?->J
J<X => "+ GRAND"
J>X => "+ PETIT"
J=X => Goto 1
Goto 0
Lbl 1
"BRAVO VOUS AVEZ TROUVE"¤
Goto A
```

Voilà c'est fait !

Il est possible que par la suite, dans mes exemples, que je n'aille pas à la ligne après mes labels mais que je mette les deux points (:)(chemin d'accès : [Shift], [VARS] (Prgm), [F6], [F5] (:).). Je trouve ça plus clair .

Le Getkey

Le Getkey est une fonction très utile mais pas très facile à utiliser car il prend un paramètre en compte difficile à déterminer .

Le but

Cette fonction est une sorte de variable : si jamais vous appuyer sur une touche pendant l'exécution du programme, la valeur de la touche sera assignée à la variable Getkey (même si ce n'est pas vraiment une variable). Imaginons que vous créiez un jeu du style Snake II et que vous vouliez faire diriger votre serpent vers la droite, il vous faudra appuyer sur la flèche de droite . Or si il n'y a rien pour recevoir cette pression de la touche, votre appui, n'aura servi à rien (et oui, la calculatrice ne peut pas deviner toute seule que le serpent devra aller à droite). C'est pourquoi il faudra se servir de cette fonction .

Utilisation

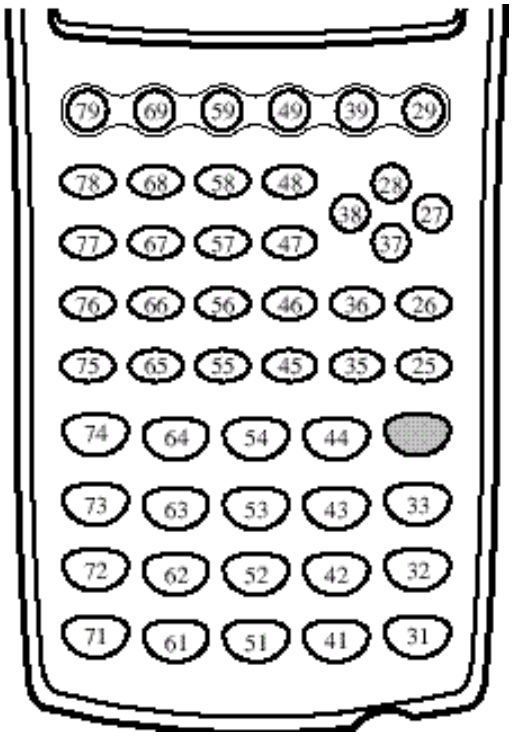
On va donc apprendre à l'utiliser car ce n'est pas évident . Dans 95 % des cas, il se place dans une boucle : ceci car la calculatrice lit les informations très vite et à moins que vous ne soyez très rapide, il me semble difficile que vous ayez le temps d'appuyer sur la touche avant que la calculatrice soit passée à la ligne suivante. Il faut donc le placer dans une boucle pour que la calculatrice le lise plusieurs fois et qu'au moins une fois, lors de la pression d'une touche (qui dure assez longtemps comparé à la vitesse de la calculatrice), le Getkey prenne une valeur . Mais bon, les explications ne sont pas vraiment importantes : ce qui importe vraiment, c'est la syntaxe :

Code : Basic CASIO
```
0->Z
Lbl 0 : Getkey -> Z
Z=31 => Goto 1
Goto 0
Lbl 1 : "Vous avez appuyer sur [EXE]"
```

J'ai utilisé la variable Z car c'est elle que j'utilise d'habitude et j'ai mis la valeur 31 car c'est celle qui correspond à la touche EXE . Vous avez compris comment ça fonctionne ?
Je n'étais pas obligé de passer par la variable Z, j'aurai directement pu mettre : Getkey=31 => Goto 1 .

Tableau de valeurs

Le problème demeure ici, les touches ont des valeurs assez difficile à retenir . Voici une image avec le Getkey de chaque touche (c'est celle qui est dans le manuel de votre calculatrice) :



La touche AC/On , n'a pas de Getkey, c'est normal car elle sert à stopper les programmes.

Voilà, à vous de l'apprendre. Non, ce n'est pas la peine, il y a une solution : il vous suffit de créer un petit programme qui vous donnera la valeur de la touche pressée .

Code : Basic CASIO
```
Lbl 0 :0->Z
Getkey ->Z
ClrText
Locate 5,5,Z
Goto 0
```

Voilà, maintenant, vous pouvez réalisez des jeux qui ressemblent à quelque chose . Beaucoup de petits jeux sont réalisables du moment qu'ils

n'ont pas de graphismes . Le chapitre suivant est d'une importance cruciale : sans lui, vous ne resterez toujours qu'à de petits jeux et ne pourrez jamais créer des jeux de plate-forme (Bomberman, Mario, ...). Ce sont les matrices et les listes mais je ne vous en dit pas plus pour l'instant ...

Encore des variables

Comme moi, durant des mois, vous devez penser que 28 variables devraient vous suffire amplement . Ceci est vrai seulement si vos jeux restent limités. Mais dès qu'ils prendront une certaine ampleur, vous aurez besoin de centaines de variables . Or nous n'en avons que 28 ... Comment faire ? ... Et bien c'est très simple, il suffit de créer des tableaux dont les cases serviraient de variables . Il y a deux types de tableau : les listes et les matrices .

Listes

Les listes ne sont pas des tableaux à proprement dit ; ce sont plutôt des colonnes . Il en existe 6 : Liste 1, Liste 2, ... Liste 6. Vous l'avez sûrement, déjà remarqué, il existe un menu dans votre calculatrice s'intitulant ainsi. C'est dans celui là que la calculatrice travaillera lorsque dans votre programme apparaîtra une fonction relative aux Listes .
Comme je vous l'ai dit, les cases des tableaux (ici les listes) sont des variables : on leur assigne donc une valeur comme on en assigne une à une variable normale . Mais avant de déclarer une valeur à une des cases, il faut créer la liste en question ...

Création d'une Liste

Pour ce faire, il y a deux solutions :
- la première crée une liste avec comme valeur, à chaque case 0 . Voici la syntaxe :

<nombre de lignes voulues> -> Dim list <numéro de la liste voulue>

Le chemin d'accès de la fonction Dim : [Optn], [F1] (List), [F3] (Dim)
Le chemin d'accès de la fonction List : [Optn], [F1] (List), [F1] (List)

-la seconde solution crée une liste avec les valeurs que vous lui demandez d'insérer :

{<valeur 1>, <valeur 2>, <valeur 3>} -> List <numéro de liste voulue>.

Assigner une valeur

Je le disais tout à l'heure, assigner une valeur à une case de liste ou à une variable, c'est quasiment pareil .
Je vous montre :

<valeur> -> List <n° de la liste>[<n° de la ligne>]

Un petit exemple : je veux donner la valeur 3 à la ligne 2 de la liste 1

Code : Basic CASIO
3->List 1[2]

SI vous voulez que cela fonctionne, il faut que la liste soit déjà créée : la ligne que vous mettez (dernier paramètre) doit être compris dans les dimensions de la liste que vous avez créée au début .

Récupérer la valeur d'une case

Et bien c'est pareil qu'avec les variables sauf qu'au lieu de mettre une lettre, vous mettez :

List <n° de la liste>[<ligne de la liste>]

Un petit exemple : nous allons récupérer la valeur que nous avons enregistré tout à l'heure :

Code : Basic CASIO
List 1[2]¤

Voilà la valeur de cette case sera affichée à l'écran.

Est ce que les calculs sont les même qu'avec les variables ? Peut on additionner deux cases ?

Oui, il vous suffit d'additionner deux cases en prenant leur nom : c'est exactement pareil qu'avec les variables . Vous pouvez même faire des choses plus intéressantes (que vous ne pouvez pas faire avec les variables) mais nous verrons ça plus tard ...

Matrices

Les matrices, elles, sont vraiment des tableaux ; ce sont plutôt Il en existe 27 : Mat A, Mat B,... Mat Z, Mat Ans. Il existe aussi un menu dans votre calculatrice s'intitulant ainsi. C'est dans celui là que la calculatrice travaillera lorsque dans votre programme apparaîtra une fonction relative aux matrices .
Le fonctionnement est quasiment le même que celui des listes :

Création d'une Matrice

Pour ce faire, il y a aussi deux solutions :
- la première crée une matrice avec comme valeur, à chaque case 0 . Voici la syntaxe :

{<nombre de lignes voulues>,<nombre de colonnes voulues>} -> Dim Mat <nom de la matrice voulue>

Le chemin d'accès de la fonction Dim : [Optn], [F1] (List), [F3] (Dim)
Le chemin d'accès de la fonction Mat : [Optn], [F2] (Mat), [F1] (Mat)

-la seconde solution crée une matrice avec les valeurs que vous lui demandez d'insérer :

[[<valeur 1>, <valeur 2>, <valeur 3>],[<valeur 4>, <valeur 5>, <valeur 6>],[<valeur 7>, <valeur 8>, <valeur 9>]] -> Mat <nom de matrice voulue>.

Ceci donnera une matrice de 3 lignes sur 3 colonnes . En fait, la calculatrice crée une nouvelle ligne, à chaque fois que vous fermez les crochets et que mettez une virgule.

Assigner une valeur

C'est pareil que pour une variable ou une Liste :

<valeur> -> Mat <nom de la matrice>[<n° de la ligne>,[n° de la colonne]

Un petit exemple : je veux donner la valeur 3 à la ligne 2 et à la colonne 2 de la Matrice A .

Code : Basic CASIO
```
3->Mat A[2,2]
```

Si vous voulez que cela fonctionne, il faut que la matrice soit déjà créée : les paramètres que vous entrez doivent être compris dans les dimensions de la matrice que vous avez créée au début .

Récupérer la valeur d'une case

Et bien c'est pareil qu'avec les variables et les Listes :

Mat <Lettre de la matrice>[<colonne de la matrice>,<ligne de la matrice>]

Un petit exemple : nous allons récupérer la valeur que nous avons enregistré tout à l'heure :

Code : Basic CASIO

Mat A[2,2]¤

Voilà la valeur de cette case sera affichée à l'écran.

Là aussi, des calculs sur les matrices sont possibles mais nous verrons ça plus tard : ce n'est pas encore au programme mais ne vous en faîtes pas, ça arrivera bien assez vite !
Voilà, ça y est les matrices et les listes n'ont plus de secrets pour vous ... enfin non mais nous les verrons plus tard . Pour l'instant, vous ne verrez pas bien l'intérêt de ces tableaux mais vous comprendrez dès que vous ferez des programmes conséquents qu'ils sont indispensables .

Les compteurs

Alors, nous voilà encore repartis ... Mais rassurez vous, cette partie est courte et assez facile à comprendre (et à retenir) : pratiquement que de la logique. Ça ne devrait pas vous prendre plus de 15 minutes.

A+1->A

Si vous êtes un minimum logique, un petit peu curieux, ... vous aurez trouvé ceci tout seul (avant même que vous ayez lu cette partie, je dirai même après la partie sur les variables). Mais si vous ne l'avez pas trouvé tout seul, ne vous inquiétez pas, c'est évident (même pour l'âne le plus débile).
J'en ai déjà vaguement fait allusion lors de la création du jeu du + ou du -

Nous allons donc faire des opérations sur les variables ... et oui, il suffit de rajouter un nombre une variable et hop, le tour et joué. Non ? Pas convaincu ? Une petite démonstration alors ; je vais me servir de la boucle While :

Code : Basic CASIO
```
0->A
While A<15
A+1->A
A¤
WhileEnd
```

Alors, qu'est ce que vous en pensez ? Évident non ? Pourquoi ne pas y avoir pensez avant ? En fait, il suffit d'ajouter un nombre (qui peut même être une variable) à une variable et d'assigner cette nouvelle valeur à cette même variable. J'espère que vous avez compris, car sinon, ça ne sert à rien de continuer, c'est la partie la plus évidente de toute la programmation sur calculatrice (enfin je pense). Je ne vous donne pas de syntaxe, ça ne vous servirait à rien ... Pourquoi faire long quand on peut faire compliqué ?

Isz

Là par contre, c'est un peu plus compliqué car il y a une fonction à retenir mais ça reste toujours très facile. Le plus dur est de savoir que cette fonction a une double utilité ce que peu de monde sait. Mais vous vous le saurez. 
Nous allons commencer par la partie la plus simple : et bien tout d'abord, c'est un compteur (sans blague ... ). Voici un petit code qui vous fera comprendre :

Code : Basic CASIO
```
0->A
While A<15
Isz A
A¤
WhileEnd
```

Le chemin d'accès de la fonction Isz est [Shift], [VARS] (Prgm), [F3] (Jump), [F4] (Isz)

Et bien comme vous le constatez par vous même, ça fait la même chose que tout à l'heure. Pour l'instant, vous ne pouvez y voir qu'une seule différence, la valeur que l'on ajoute est forcément 1. Or il en existe une, que nous allons voir tout de suite. Essayez le même code sauf que au lieu de donner la valeur 0 à A au début donner lui -1, comme ça :

Code : Basic CASIO
```
-1->A
While A<15
Isz A
A¤
WhileEnd
```

Normalement la première valeur affichée à l'écran devrait être un zéro (et oui -1+1=0 non ?) or ce n'est pas ce qui se produit ...  ne vous en faîtes pas, ce n'est pas un bug de la part de votre calculatrice, c'est même normal ...  Car cette fonction, lorsque la variable se trouvant derrière elle est égale à zéro, la fonction suivante est sautée : l'exécution est donc directement passée à fonction WhileEnd sans affiché la valeur de A. Personnellement, je n'ai jamais eu à me servir de ce saut mais on ne sait jamais  .

Dsz

Cette fonction est pratiquement la même que la précédente à une légère différence : au lieu de faire +1à chaque passage, elle fait -1. La deuxième utilité de cette fonction (le saut) persiste donc encore. Et bien voilà, que dire de plus ? je vais vous donner un petit exemple pour bien comprendre.

Code : Basic CASIO
```
1->A
While A>-15
Dsz A
A¤
WhileEnd
```

Le chemin d'accès de la fonction Isz est [Shift], [VARS] (Prgm), [F3] (Jump), [F5] (Dsz)

Vous remarquerez que j'ai réuni les deux exemples précédents en un : la diminution de 1 à chaque fois et le saut lorsque la variable est égale à zéro.

For, To, Step, Next

J'espère que vous vous souvenez de cette fonction, nous l'avons vu dans le chapitre des boucles et des conditions. Je n'ai donc rien à ajouter ...
Pourquoi en avoir parlé ici alors ?

Vous ne m'avez pas laisser de temps de finir ... je l'ai juste mis car cette fonction est aussi considérée comme un compteur, car elle permet d'augmenter ou de diminuer la valeur d'une variable. Ça y est, vous connaissez tout des compteurs et des variables ... cool non ? Rendez vous dans la prochaine partie ... 

Les sous programmes

Voici donc venu le dernier chapitre de cette grande patrie. Dans ce chapitre, je vais vous apprendre à créer des sous-programmes et à effectuer des trajets entre ces programmes durant leur exécution .

Prog

Pour ce faire, une fonction est déjà créée : Prog . Elle permet durant l'exécution d'un programme de se rendre dans un autre programme.
Voici sa syntaxe : Prog "<nom du programme>" .
C'est aussi simple que ça .
Le chemin d'accès de la fonction est [Shift], [VARS] (Prgm), [F2] (Ctl), [F1] (Prog)

Pour que cette fonction marche, il faut que le programme où la calculatrice doit se rendre existe. C'est évident mais c'est une erreur assez fréquente.

Cette seule fonction aurait pu suffire mais les programmeurs en ont décidé autrement : ils en ont créée une autre ...

Return

Cette dernière permet donc d'effectuer un retour vers le programme où vous étiez avant (avant que la fonction Prog ne soit exécutée). Cette fonction est nommée Return (et oui, encore notre anglais car Return signifie retour). Pour qu'elle fonctionne, elle n'a besoin d'aucun paramètre (n'est ce pas beau la vie ?). Son chemin d'accès est le suivant : [Shift], [VARS] (Prgm), [F2] (Ctl), [F2] (Rtrn).
Vous n'aurez besoin d'utiliser ces fonctions que si vos programmes seront volumineux (plus de 2 Ko au minimum)et normalement elles se placent dans des boucles de condition .

Si jamais au lieu d'utiliser cette fonction pour revenir dans le programme initial, vous réutilisez une fonction Prog alors ça plantera.

Convenances

Pour plus de simplicité, une règle s'est mise en place (même si elle est peu appliquée) : les noms des sous-programmes doivent commencer par un espace (alors que les programmes "sources" commencent leur nom par un caractère). C'est sûr, vous n'êtes pas obligé de faire cela mais c'est recommandé pour une meilleure clarté dans votre liste de programme.

Exemple pour mieux comprendre

Voici donc un petit exemple illustrant cette dernière partie : nous allons créer un programme qui demandera l'âge de l'utilisateur et un autre sui fera une réflexion par rapport à cet âge . Je suis d'accord avec vous, c'est complètement débile mais c'est juste pour illustrer : nous n'allons pas nous amuser à créer un programme énorme pour rien.
Voici donc le code du programme demandant l'âge et se nommant "AGE":

Code : Basic CASIO
```
ClrText
"Quel age avez vous"?->A
Prog " AGE2"
```

Voici le code du programme faisant la réflexion sur l'âge et se nommant " AGE2":

Code : Basic CASIO
```
A>18=>"Vous êtes majeur"
A<18=>"Vous êtes mineur"
A=18=>"Vous êtes majeur mais de peu"
Return
```

Voilà donc un petit exemple, débile j'en conviens mais illustrant bien ce cours .

Faîtes bien attention : le premier caractère du nom de mon sous-programme est un espace. Le nom est : " AGE2" et non "AGE2" . Il faut donc mettre un espace comme premier caractère après avoir ouvert les guillemets derrière la fonction Prog !

Bon voilà je crois que vous savez tout sur les sous-programmes : vous n'apprendrez plus rien sur ce chapitre . Simple, n'est ce pas ? Ce que je vous propose de faire pour voir si vous avez bien compris, c'est de reprendre le jeu du + ou - et de le refaire en mettant la boucle de vérification dans un sous-programme (il vous faudra utiliser les fonctions If, Then, IfEnd). C'est inutile car ce programme n'est pas volumineux mais c'est juste pour vous exercer. Bonne chance ! Ça y est, on y est arrivé, le plus dur est fait. Vous avez acquis les bases de la programmation sur calculatrice graphique . Bravo! Si vous créiez un programme vous pourriez même prendre le nom de "programmeur" ... Cool non ? Vous pouvez d'ailleurs maintenant créer des jeux conséquents : petits jeux genre Snake (délicat mais vous pouvez remplacer le serpent par une étoile, ce sera plus facile), morpion sans graphismes (penser aux matrices), labyrinthes (mais toujours sans graphismes donc très limité) ... Vous pouvez voir que ce qui vous manque maintenant, ce sont principalement les graphismes . Et bien qu'attendez vous ? A l'attaque de la prochaine partie ...

Partie 2 : Les Graphismes

Voilà, après avoir appris les bases, pour que vous puissiez faire de beaux programmes (et principalement des jeux), vous aurez absolument besoin des graphismes. Car le moyen d'affiche du texte et des variables au début (grâce à la fonction Locate), n'est pas génial : pas très beau, limité, l'écran est vite plein ... Il existe don un autre moyen, d'où l'existence de cette partie (et oui, je n'aurai pas créé une partie pour rien)

Le View-Window

Le view window, noté souvent V-Window (surtout dans le manuel) est la base de cette partie : c'est indispensable car si vous ne connaissez pas ça, vous ne pourrez pas placé les différentes formes pré-enregistrées là où vous souhaitez. Donc retenez bien. En fait, il existe deux écrans en un seul : l'écran normal (celui que vous utilisez pour faire des calculs ou pour programmez) et un écran graphique (celui que nous allons apprendre à utiliser). Notez bien que ces deux écrans ne sont pas superposables et sont deux écrans virtuels ...

Une sorte de repère

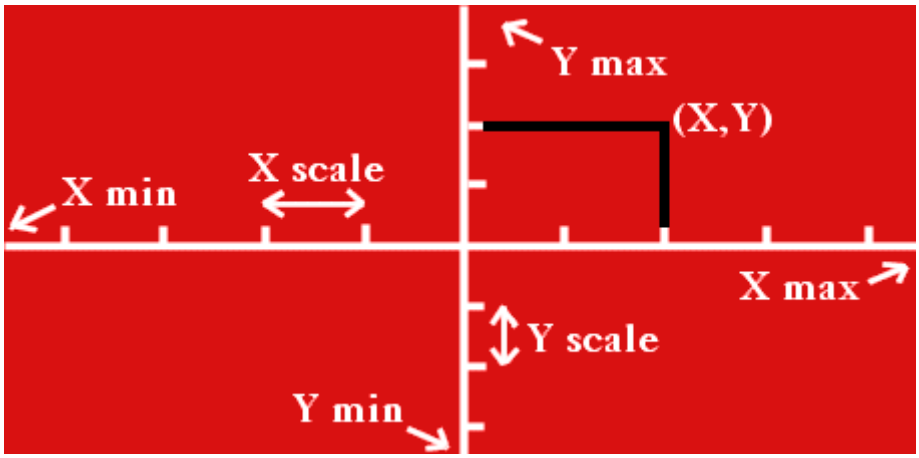
Si vous avez un minimum de connaissances sur les repères orthonormaux, cela ne devrait pas être difficile à comprendre (mais je crois que cela se voit au collège donc c'est bon ...). Car en fait, c'est aussi simple que ça : vous avez juste à créer un repère orthonormal et donnez les coordonnées du centre, des extrémités ... des formes géométriques que vous voulez insérer. Mais nous allons refaire un peu d'anglais pour commencer car ça me servira pour après :

View Window signifie fenêtre d'affichage. Évident non ? A partir de maintenant, je n'utiliserai presque que ce terme.
.
Il suffit donc de paramétrer quelques petites choses et le tour est joué. Je vais vous donner la syntaxe et nous essaierons de comprendre après.

Viewwindow <X min>,<X max>,<X scale>,<Y min>,<Y max>,<Y scale>

Et qu'est ce que ça signifie tout ça ? Car il y a certaines choses qui ne sont pas très claires ...

Ne vous en faîtes pas, je vais vous expliquer. Mais d'abord, le chemin d'accès de la fonction view window : [Shift], [F3] (V-WIN), [F1] (V-Win). Ensuite, les X signifient les abscisses (partie horizontale du repères) et les Y les ordonnées (partie verticale du repère). Je vous ai bien dit, si vous ne connaissez pas ces deux mots (ordonnées et abscisses), vous aurez des difficultés à comprendre. X min, correspond à l'emplacement de la plus petite valeur de présente sur l'axe des abscisses, X max, à la plus grande, Y min à la plus petite valeur présente sur l'axe des ordonnées et Y max, la plus grande. Jusque là, ça va, pas trop compliqué, sauf qu'il y a encore deux paramètres (le troisième et le dernier) que nous n'avons pas expliqué. Ce sont les échelles, c'est à dire l'écart entre deux points d'abscisses. C'est assez compliqué à expliquer mais en général (presque tout le temps), on utilise la valeur 1.
Voici une image pour mieux comprendre les paramètres pris en compte :



Nous allons prendre un petit exemple pour bien comprendre. Je vous donne le code et on l'expliquera ensemble.

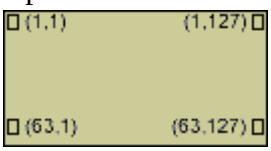
Code : Basic CASIO
View Window -6.3,6.3,1,-3.1,3.1,1

Bon, je ne fais pas dans l'imaginatif ; c'est le réglage initial (celui que vous utilisez pour tracer une courbe quelconque dans le menu Graph si vous n'avez pas modifié ces paramètres. Nous aurons donc une abscisse minimale de -6.3, une abscisse maximale de 6.3, une ordonnée minimale de -3.1, une ordonnée maximale de 3.1 et les échelles sur les deux axes sont de 1 (le plus simple et le plus usuel (à part dans le repère trigonométrique)). Cela nous fait donc un repère d'axes au milieu de l'écran. Or ce n'est pas le plus pratique il y a des nombres décimaux et négatifs ... pourquoi faire compliqué quand on peut faire simple : nous allons utiliser une fenêtre d'affichage avec que des nombres positifs et entiers. Pour ce faire, il faut rentrer ces

paramètres-ci, on les expliquera après.

Code : Basic CASIO
```
ViewWindow 1,127,1,1,63,1
```

En fait l'écran possède 127 pixels sur 63, d'où ces nombres : comme ça, chaque pixel aura des coordonnées entières. Notez que nous avons gardé 1 pour échelle sur les deux axes. Le point en bas à gauche de l'écran aura donc comme coordonnées 1,1, le point en bas à droite 127,1, le point en haut à gauche 1,63, le point en haut à droite 127,63. Notez que l'on met toujours les abscisses en premier (comme en mathématiques).
Voici une image pour mieux comprendre cette nouvelle définition de l'écran :



Cette image est une propriété de :
http://www.casioexpert.com/

Voilà, c'est bon, vous savez régler la fenêtre d'affichage mais vous avez du remarquer qu'il existait des axes ou une grille de points ... (cela varie selon les paramètres de votre calculatrice)... il faut donc les enlever car pour faire des graphismes, ce n'est pas génial.

Les axes et les autres paramètres

Pour changer ces paramètres, et bien il existe des fonctions, pourquoi changer ? Et mieux encore, elles sont toutes répertoriées au même endroit. Je vais donc me contenter de donner les chemins d'accès et quelques brèves explications. C'est pas beau la vie ?

Les axes

Ce sera le principal obstacle mais un obstacle simple à franchir :
[Shift], [MENU] (Setup), [F4] (AXES) et puis [F2] (Off) pour les enlever. Ceci s'affichera alors :

Code : Basic CASIO
```
AxesOff
```

Ils seront alors effacées lorsque durant l'exécution du programme, on arrivera à cette ligne du code.
Pour les rétablir , il suffit de faire [F1] (On) au lieu de [F2] (Off) à la fin du chemin d'accès.
Le code affiché sera donc :

Code : Basic CASIO
```
AxesOn
```

La grille

Il arrive qu'une grille des différents points de coordonnées entières soit affichée (si votre fenêtre d'affichage est large, vous aurez un écran noir car il y aura beaucoup de coordonnés. [Shift], [MENU] (Setup), [F3] (GRID) et puis [F2] (Off) pour l'enlever.
Ceci s'affichera alors :

Code : Basic CASIO
GridOff

Elle sera alors effacée lorsque durant l'exécution du programme, on arrivera à cette ligne du code.
Pour la rétablir , il suffit de faire [F1] (On) au lieu de [F2] (Off) à la fin du chemin d'accès. Le code affiché sera donc :

Code : Basic CASIO
GridOn

Les labels

Il arrive que le nom des axes (X et Y) soit affichée.
[Shift], [MENU] (Setup), [F5] (LABL) et puis [F2] (Off) pour les enlever. Ceci s'affichera alors :

Code : Basic CASIO
LabelOff

Ils seront alors effacés lorsque durant l'exécution du programme, on arrivera à cette ligne du code.
Pour les rétablir , il suffit de faire [F1] (On) au lieu de [F2] (Off) à la fin du chemin d'accès. Le code affiché sera donc :

Code : Basic CASIO
LabelOn

Voilà, ce sont les principaux obstacles que vous rencontrerez, il en existe d'autres mais qui sont très rares, si jamais ça vous arrive, il existe ce forum.

Rétablir les paramètres

Et oui, il ne suffit pas de faire des modifications, il faut savoir les rétablir (autrement qu'avec des fonctions) pour tracer des graphiques normaux (en cours pas exemple car au départ ce n'est pas un jeu cet outil). Mais vous allez voir, c'est encore plus simple que les fonctions mais pour ce faire, il va falloir sortir du menu où nous avons travaillé jusqu'à maintenant et vous

rendre dans le menu RUN (1, celui pour les calculs) ou le menu 5 (Graph, celui pour tracer des courbes).

Le View-Window

Il faut faire [Shift], [F3] (V-WIN) et là comme pas magie, une fenêtre s'ouvre (ce serait presque Windows XP) avec plein de paramètres où il faut éviter de toucher à tout. Je vais vous indiquer que faire, il suffirait de remplacer les valeurs présentes pour X min, X max, X scale, Y min, Y max et Y scale par celles d'origine mais c'est laborieux et source d'erreurs, il y a beaucoup plus simple. Faites simplement [F1] (INIT) et vous retrouverez les réglages de la fenêtre d'affichage d'origine. Vous pouvez aussi faire [F3] (STD) pour avoir un réglage un peu plus large mais ce n'est pas nécessaire. Normalement, vous devriez voir ça en seconde générale. N'appuyez pas sur [F2] (TRIG) tant que vous n'aurez pas vu ça en cours, c'est un peu plus compliqué (ça correspond aux fonctions trigonométriques). Vous avez peut être remarquer que si vous descendez la zone de sélection jusqu'en bas, trois autres réglages apparaissent, ne les toucher pas car c'est encore autre chose de plus compliqué aussi.

Les autres réglages

C'est un peu le même genre de réglages : une fenêtre avec des paramètres va s'ouvrir suite à une combinaison de touches : [Shift], [MENU] (Setup). Alors là, ne touchez à rien que je ne vous aurai pas indiqué (ou alors que vous ne sachiez pas) car vous pourriez vous retrouver avec des choses bizarres après et être obligé de faire un RESET car vous n'aurez pas réussi à retrouver le champ modifié. Donc évitez de toucher à tout ; si vous avez un doute, reportez vous à votre manuel. Grâce aux flèches haut et bas, cherchez les paramètres Grid, Axes et Label (ils se suivent) et grâce aux touches [F1] (On) et [F2] (Off), changez ces paramètres. Je n'ai pas besoin de vous expliquer plus, vous avez compris ? Et bien voilà, on est prêt à se lancer dans les graphismes ... Excusez moi de vous avoir ennuyer avec cette partie mais c'était un passage obligatoire. Vous me remercierez par la suite .

Les Points et les lignes

Voici donc le début réel des graphismes : dans cette partie, nous allons apprendre à tracer des points puis des lignes.

Points

Il existe deux fonctions pour afficher des points à l'écran : nous allons apprendre à nous servir des deux mais je vous déconseille l'utilisation de la seconde car elle est assez compliquée.

J'utiliserai toujours une fenêtre d'affichage avec les réglages suivants :
Viewwindow 1,127,1,1,63,1.

Par Ilae pour planete-casio.fr

Plot On

En fait là, il existe trois sous-fonctions : Plot On, Plot Off, Plot Chg. J'imagine que grâce à votre anglais, vous avez compris le but des deux premières fonctions mais la troisième j'en doute. Je vais donc les récapituler les trois.
Plot On : afficher un point
Plot Off : effacer un point
Plot Chg : changer de statut un point de l'écran : s'il est affiché, il s'efface et s'il n'est pas affiché, il s'affiche.
La syntaxe maintenant :
Plot On <abscisse>, <ordonnée>
Plot Off <abscisse>, <ordonnée>
Plot Chg <abscisse>, <ordonnée>
En fait, c'est la même syntaxe pour chacune de ces trois fonctions.
Le chemin d'accès est le suivant : [Shift], [F4] (SKTH) , [F6], [F1] (Plot) puis [F2] (Pl-On) pour Plot On ou [F3] (Pl-Off) pour Plot Off ou [F4] (Pl-Chg) pour Plot Chg.

Pxl On

Dans notre cas, cette fonction ne sera pas très utile car nous avons défini un repère qui donne à chaque pixel de l'écran des coordonnées. En effet, cette fonction aura pour but d'afficher un pixel. Elle ne vous sera utile que si vous utilisez un ViewWindow différent de celui que je vous ai indiqué car dans celui-là un point ne correspondra pas forcément à un pixel …
Dans le but de faire un cours complet, je vous donne aussi ses caractéristiques :
Pxl On : afficher un point
Pxl Off : effacer un point
Pxl Chg : changer de statut un point de l'écran : s'il est affiché, il s'efface et s'il n'est pas affiché, il s'affiche.
La syntaxe maintenant :
Pxl On <ordonnée>, <abscisse>
Pxl Off <ordonnée>, <abscisse>
Pxl Chg <ordonnée>, <abscisse>
En fait, c'est la même syntaxe pour chacune de ces trois fonctions. Mais attention, l'origine du repère n'est pas le coin bas gauche de l'écran ! C'est le coin situé en haut à gauche (comme pour le Locate si vous vous souvenez). Ne l'oubliez pas ! Pour être plus précis, voici quelques explications : le comptage commence en haut à gauche et se prolonge suivant les axes d'un repère orthogonal. Le point en haut à gauche aura donc comme coordonnées 1,1 et le point en bas à droite aura comme coordonnées 63,127. Je vous donne les coordonnées des autres angles afin que vous compreniez parfaitement (rien en vaut un bon exemple) : le point en haut à droite : 1,127 et le point en bas à gauche : 63,1.
Le chemin d'accès est le suivant : [Shift], [F4] (SKTH) , [F6], [F6], [F3] (Pixl) puis [F1] (On) pour Pxl On ou [F2] (Off) pour Pxl Off ou [F3] (Chg) pour Pxl Chg.

Lignes

Nous allons maintenant apprendre à dessiner des lignes. Certes nous ne pourrions qu'afficher des points mais ce ne serait pas très judicieux. En effet, il existe une (enfin deux) fonctions permettant de tracer des lignes en fonction du repère.

Line

Voici la première fonction : elle permet de joindre deux points définis précédemment. En effet, afin que cette fonction marche, il faut tout d'abord définir deux points (grâce aux instructions que nous avons apprises ci-dessus) puis ensuite insérer la fonction Line.
Voici donc une syntaxe possible :
Plot On <abscisse 1>,<ordonnée 1>
Plot On <abscisse 2>, <ordonnée 2>
Line

Grâce à ce code, les points 1 et 2 (correspondant aux coordonnées 1 et 2) seront liés. Le chemin d'accès de cette fonction est le suivant : [Shift], [F4] (Sketch), [F6], [F2] (Line), [F1] (Line).
Voici un petit exemple pour comprendre : nous allons lier le coin en haut à gauche avec le coin en bas à droite. Nous utilisons toujours le même repère que précédemment.

Code : Basic CASIO
```
Cls
Plot On 1,63
Plot On 127,1
Line
```

Note : recopiez le Cls aussi : c'est une fonction dont je vous expliquerai le but et le fonctionnement une fois que nous en aurons fini avec les lignes. Voici son chemin d'accès : [Shift], [F4] (Sketch), [F1] (Cls).
Et voilà, en trois lignes nous avons afficher 63 points … Mais il existe une solution qui nous permettra de l'afficher en une ligne … 

F-Line

La voici cette fonction magique. Cette dernière est beaucoup plus pratique et beaucoup plus utilisée. Ceci car nous n'avons pas besoin de définir les deux points qui seront liés avant : tout est défini en une seule fonction. Je vous donne la syntaxe, cela vous éclairera et vaudra mieux qu'un long paragraphe.
F-Line <abscisse 1>,<ordonnée 1>,<abscisse 2>,<ordonnée 2>

Je pense que vous comprenez le principe : la fonction F-Line va lier les deux points définis (point 1 par les coordonnées 1 et le point 2 par les coordonnées 2). Voici son chemin d'accès : [Shift], [F4] (Sketch), [F6], [F2] (Line), [F2] (F-Line).

Prenons le même exemple que précédemment :

Code : Basic CASIO
```
Cls
F-Line 1,63,127,1
```

Ce code tracera exactement la même chose mais en prenant deux fois moins de place … Et croyez moi, il ne faut pas négliger ce caractère … Nous en reparlerons dans les chapitres suivants.
Je vous déconseille donc l'utilisation de cette première fonction et de favoriser l'utilisation du F-Line.
Il existe encore deux fonctions mais elles ne permettent que de tracer des droites complètes (d'un bout de l'écran à l'autre).

Horizontal

Comme je vous le disais précédemment et comme vous pouvez le deviner, cette fonction va nous permettre de tracer une ligne horizontale traversant tout l'écran. Voici la syntaxe :
Horizontal <ordonnée>
Je vous signale que nous travaillons toujours avec le même ViewWindow. Voici son chemin d'accès : [Shift], [F4] (Sketch), [F6], [F5] (Hztl).
Si nous voulons tracer une droite horizontale au milieu de l'écran, il vous faudra taper le code suivant :

Code : Basic CASIO
```
Cls
Horizontal 32
```

En effet, dans notre cas (dans notre configuration d'affichage), 32 est l'ordonnée correspondant au milieu de l'écran.

Vertical

Cette fonction est la même mais permet de tracer une droite verticale et non horizontale.
Vertical <abscisse>
Voici son chemin d'accès : [Shift], [F4] (Sketch), [F6], [F4] (Vert).
Pour tracer une droite horizontale au milieu de l'écran, il faudra utiliser le code suivant :

Code : Basic CASIO
```
Cls
Vertical 64
```

En effet, dans notre cas, 64 est l'abscisse correspondant au milieu de l'écran.
Si jamais nous regroupons les deux derniers codes, nous aurons séparer l'écran en quatre cadres identiques.

Voilà, nous avons vu toutes les fonctions permettant de tracer des lignes « prédéfinies » par la calculatrice. Nous allons maintenant passer à un autre élément important…

Le texte

Nous avons déjà vu une fonction permettant d'afficher du texte et bien nous allons en voir une deuxième. Mais avant, on va faire un petit rappel : n'essayez pas d'afficher du texte sur cet écran (graphique : celui défini grâce au ViewWindow) grâce à la fonction Locate. En effet celle là sert à l'autre écran (celui composé de 21 colonnes et de 7 lignes). Notre fonction Locate est donc inutile dans ce chapitre. C'est pour ça que nous allons en voir une autre. Certes, comme pour les lignes, nous pourrions afficher le texte point par point mais ce n'est pas vraiment pas pratique, voire impossible !

La fonction Text

J'ai donc une solution à vous proposer : la fonction Text. Vous noterez qu'elle se comporte de la même manière que les fonctions Pxl On, Pxl Off, Pxl Chg. Elle ne prend donc pas compte de la fenêtre d'affichage que vous avez défini puisqu'elle tient seulement compte des pixels. Le comptage commence en haut à gauche et se prolonge suivant les axes d'un repère orthogonal. Le point en haut à gauche aura donc comme coordonnées 1,1 et le point en bas à droite aura comme coordonnées 63,127. Je vous donne les coordonnées des autres angles afin que vous compreniez parfaitement (rien en vaut un bon exemple) : le point en haut à droite : 1,127 et le point en bas à gauche : 63,1.
La syntaxe est donc : Text <ordonnée>, <abscisse>, « texte à afficher »
Et voici donc son chemin d'accès : [Shift], [F4] (Sketch), [F6], [F6], [F2] (Text).
Nous allons donc afficher un petit texte en haut à gauche de l'écran et un autre en bas à droite.

Code : Basic CASIO
```
Cls
Text 1,1, « coin en haut à gauche »
Text 58,42, « coin en bas à droite »
```

Vous noterez que j'ai mis 58 comme ordonnée et non 63. En effet, c'est le haut de l'écriture qui est pris en compte lorsqu'on affiche grâce aux coordonnées.
Vous aurez sûrement remarquer que cette écriture est plus fine et plus jolie que l'affichage du Locate.

Un petit TP pour bien exploiter ces fonctions

Et bien nous voilà lancés... Vous verrez si vous avez bien tout compris jusqu'ici ce sera très facile et si ce n'est pas le cas, ça permettra de remettre de l'ordre dans la case CASIO de votre esprit.

Par Ilae pour planete-casio.fr

Que faire ? J'ai trop (peu) d'idées... C'est bon j'ai trouvé un petit programme très simple.
Nous allons créer un petit cadre dans lequel figureront :

- votre pseudo
- votre prénom
- votre âge
- et votre modèle de calculatrice

Avouez c'est un peu débile mais ce sera un exemple qui pourra être repris (en partie) lorsque vous ferez l'image de lancement de votre premier jeu.
J'allais oublier : je veux que votre cadre soit tracé et centré ! Il faudra donc l'adapter à la longueur de vos "identifiants". Je vous l'accorde, ce n'est pas très clair mais vous le comprendrez par vous même et ce sera mieux (on apprend de ses erreurs).
Voici donc un peu ce que vous devriez obtenir :

Notez que les positions ne sont pas parfaites (les informations ne sont pas écrites exactement au milieu de l'écran) mais ce n'est pas le but. Ce que je veux, c'est vous faire utiliser les fonctions vues précédemment.
Sur ce je vous souhaite bonne chance.

Si jamais vous bloquez voici quelques indications ci-dessous mais ne les lisez que si vous ne trouvez pas.

Indication n°1 :
Avez vous bien réglé le View-Window ?

Indication n°2 :
Avez vous bien pensé à effacer l'écran avant d'insérer le code des graphismes ? Grâce à la fonction Cls ?

Indication n°3 :
Les principales fonctions à utiliser sont les fonctions F-line et Text. Vous pouvez en utiliser d'autres mais je vous conseille ces dernières.

Indication n°4 :
Avez vous bien fait attention aux différents critères pris en compte par ces fonctions ?
Pour le F-Line, c'est par rapport à la fenêtre d'affichage.
Pour le Text, c'est par rapport aux pixels.
Pour plus d'informations, relisez les chapitres précédents.

Par Ilae pour planete-casio.fr

Et bien voilà, je ne peux pas vous aider plus... Si jamais vous n'avez pas réussi, ne désespérez pas, relisez les parties précédentes de ce cours et réessayez. Il n'y a pas de raison que vous n'y arriviez pas. Et surtout n'abandonnez jamais.

Voici donc le code de la solution. Normalement vous n'en avez pas besoin car si votre code marche, gardez le et si il ne marche pas, continuez de chercher !

Code : Basic CASIO

```
ViewWindow 1,127,1,1,63,1
Cls
Text 16,55,"ILAE"
Text 26,53,"FLORENT"
Text 36,53,"15 ANS"
Text 46,52,"GRAPH 65"
F-Line 50,51,84,51
F-Line 50,51,50,12
F-Line 50,12,84,12
F-Line 84,12,84,51
```

Il est assez simple. N'est-ce pas ?

Nous allons rajouter quelques petites choses à ce programme afin de continuer de s'entrainer. Vous me suivez toujours ?
Votre mission, si vous l'acceptez, sera de colorer le fond de l'écran de votre calculatrice. Pas avec un stylo... Mais mieux vaut un dessin plutôt que de longues phrases :

Et bien on y va...

Ah non, j'allais oublier : je vais poser une condition qui vous fera réfléchir mais qui vous sera très utile. Je ne veux pas que vous demandiez à la calulatrice :

- afficher une ligne à la ligne 1
- afficher une ligne à la ligne 2
- afficher une ligne à la ligne 3
- ...

Ce serait très long et inutile : la calculatrice mettrait un temps fou à tout exécuter (déjà qu'elle est lente alors là...). Je veux donc que vous trouviez une solution.
Comme pour le TP précédent, je vais vous donner des indications. Ne les lisez que si vous ne trouvez pas tout seul. Bonne chance.

Indication n°1 :
Utilisez les fonctions Vertical et Horizontal. Ce sera plus facile.

Indication n°2 :
Utilisez les compteurs For, To, (Step,) Next afin de créer une boucle et n'avoir que quelques lignes de code.

Comme pour le précédent exemple si jamais vous ne trouvez pas, ne vous découragez pas, c'est "normal" car c'est assez difficile ce que je vous demande. Réessayez et si jamais vous ne trouvez pas, envoyez moi un MP, je vous donnerai une indication. Si je ne réponds pas au bout de 3 jours, regardez la solution ci-dessous mais je vous le déconseille vraiment. Ne l'oubliez pas.

Voici donc la solution :

Code : Basic CASIO
```
ViewWindow 1,127,1,1,63,1
Cls
Text 16,55,"ILAE"
Text 26,53,"FLORENT"
Text 36,53,"15 ANS"
Text 46,52,"GRAPH 65"
F-Line 50,51,84,51
F-Line 50,51,50,12
F-Line 50,12,84,12
F-Line 84,12,84,51
For 0->A To 12
Horizontal A
Next
For 51->A To 64
Horizontal A
Next
For 0->A To 50
Vertical A
Next
For 84->A To 127
Vertical A
Next
```

Avouez que ce n'était pas évident. Si vous avez trouvé tout seul, sans lire les indications, je vous en félicite car c'était loin d'être évident. Vous ferez un excellent programmeur.
Notez que nous pourrions encore réduire ce code (en taille) mais il vous manque encore quelques notions... Bien que vous devriez pouvoir les deviner...
Mais nous reparlerons de ça plus tard... Pour l'instant, continuons notre progression...

Les différents modes d'effacement

Nous allons maintenant voir les différentes fonctions et principes d'effacement de l'écran graphique de l'écran.

Cls ~ Text « » ~ ClrGraph

Cls

Cette fonction, dont je vous ai déjà donné le chemin d'accès ci-dessus, permet d'effacer toute la partie visible (vous comprendrez ce terme par la suite) de l'écran. Elle ne prend en compte aucun paramètre : il suffit de la placer dans le code.
Retenez bien cette fonction car bien que très simple, elle est très usuelle.

Text « »

Cette astuce va permettre d'effacer une partie bien définie de l'écran contrairement à la fonction Cls qui efface tout l'écran. Il suffit en fait d'afficher grâce à la fonction Text des espaces. Une fois qu'on le dit, ça paraît évident. N'est ce pas ?

ClrGraph

Cette dernière fonction permettant d'effacer l'écran va vous permettre de comprendre le terme que j'ai employé pour la fonction Cls (partie visible). En effet, elle ne fait pas qu'effacer l'écran ; elle rétablit aussi tous les paramètres d'origine (axes, grille, labels, ViewWindow …).
Comme la fonction Cls, elle ne prend en compte aucun paramètre. Son chemin d'accès est : [Shift], [VARS] (Prgm), [F6], [F1] (Clr), [F2] (Grph).

Cls ~ Text « » ~ ClrGraph

Notre petit programme...

Que pourrions nous faire cette fois-ci ? Une idée ? Et bien moi j'en ai une, nous allons afficher à l'écran un symbole que nous allons faire bouger grâce aux flèches de direction. Afin de bien vous faire comprendre l'utilité des systèmes d'effacement, nous allons effectuer ce programme en deux temps : d'abord sans effacement afin que vous vous rendiez compte qu'il y a un problème et ensuite avec effacement.
Le symbole que nous allons afficher est le signe "multiplier" (x). Il faudra pouvoir diriger ce symbole grâce aux flèches de direction présentes en haut à droite du clavier de votre

calculatrice. Nous allons procéder comme précedemment, je vous donnerai des astuces mais ne les lisez que si vous n'arrivez pas à le faire seul. Bonne chance !
Notez que je ne vous donne au départ que très peu d'indications pour que vous appreniez par vous même à tout faire !

Sans effacement

astuce n°1 :

Code : Basic CASIO
```
Text A, B, "x"
```
Utilisez des variables comme paramètres de la fonction texte (et pas X et Y parce que ça pourrait planter ! Quelque chose du genre :

Code : Basic CASIO
```
Text A,B, "x"
```

astuce n°2 :
Utilisez la fonction Getkey pour changer les nombres retenus dans les variables.

astuce n°3 :
Voici un bout du code :

Code : Basic CASIO
```
Lbl 1
Text A, B, "x"
Getkey = 27 => B+1->B
Goto 1
```

N'essayez pas de caser ce morceau de code dans le votre, c'est jsute une indication. Il faudra changer certaines choses afin qu'il puisse marcher. N'essayez donc pas de fiare du copier-coller, faites marcher votre cervelle  .

Bon normalement vous devriez avoir trouvé. Si ce n'est pas le cas, effectuez la même démarche qu'au TP précédent.

Voici donc la solution (notez que vous pouvez avoir un code différent) :

Code : Basic CASIO
```
Cls
63 -> B
33 -> A
```

```
0 -> X
While 0<1
Text A, B, "x"
Getkey -> X
X=27 => B+1 -> B
X=38 => B-1 -> B
X=28 => A-1 -> A
X=37 => A+1 -> A
WhileEnd
```

Et bien parce que la fonction Text n'utilise pas la fenêtre d'affichage, elle se réfère aux pixles. Donc pourquoi s'encombrer de lignes de code inutiles ?

Pourquoi n'avons nous pas rélgé de fenêtre d'affichage ?

J'étais sûr que ceci allait vous poser problème... C'est une boucle infini (en effet 0 est toujours inférieur à 0 donc la condition est toujours respectée) car il faut toujours que la calculatrice soit prête à ce que nous appuyions sur une touche.

A quoi sert la boucle While ? Et pourquoi une telle condition ?

Les espaces que j'ai ajoutés dans mon code ne devront pas être recopiés sur votre CASIO. Je les ai mis pour rendre le code plus lisible.

Ca parait tout simple maintenant. Non ? Mais il se peut que vous ayez un problème... En effet lorsque votre symbole arrive au bord de l'écran et le dépasse un message d'erreur arrive ! Et oui, comment voulez vous que la calculatrice affiche un symbole dont les coordonnées ne sont pas comprises dans la fenêtre d'affichage. Et bien elle plante lamentablement. Si jamais vous avez une idée pour contourner ce bug, mettez la de côté, nous essaierons de le résoudre à la fin (car ce n'est pas notre princiapl souci).
Vous aurez remarqué que le symbole laisse une longue trainée derrière lui et au bout de quelques temps de déplacement on ne voit plus rien. C'est dommage, un si beau programme pour rien . Et bien faisons en sorte qu'il n'y ait à chaque fois qu'un symbole d'affiché (le dernier déplacé sinon ça n'a aucun intérêt).

Avec effacement

Et bien à vous de trouver. Ca ne devrait pas être très dur si vous avez tout bien compris.

Indication :

Il n'y a qu'une fonction à rajouter.

Il est fort probable (si vous avez réussi) que lorsque vous éxécutez votre programme, le symbole clignote en permanence (normal puisqu'il est à chaque fois effacé puis réaffiché). Ne

vous en souciez pas pour l'instant, nous y réfléchirons à la fin (chaque chose en son temps).

Et bien voici la solution au cas où vous n'auriez pas trouvé...

La fonction à rajouter est la fonction Cls. Si vous n'avez pas trouvé, essayez de la placer dans le code afin que ça marche et surtout essayez de comprendre pourquoi.
Voici donc maintenant le code complet :

Code : Basic CASIO
```
Cls
63 -> B
33 -> A
0 -> X
While 0<1
Cls
Text A, B, "x"
Getkey -> X
X=27 => B+1 -> B
X=38 => B-1 -> B
X=28 => A-1 -> A
X=37 => A+1 -> A
WhileEnd
```

Il y a d'autres emplacements possibles pour la fonction Cls.

Si jamais vous n'avez pas réussi ces deux exercices seuls, relisez le chapitre sur les graphismes (et eventuellement celui sur le Getkey) et retentez. Si vous comprenez le code c'est déjà une bonne chose mais ce n'est pas tout.

Quelques améliorations

Vous avez remarqué que notre programme a deux problèmes :
- le clignotement du symbole
- le bug lors de la sortie de l'écran du symbole.

Et bien qu'attendons nous ? Résolvons les !
Nous allons commencer par le deuxième problème car c'est le plus facile. Nous allons procéder de la même manière que d'habitude : vous essayez de vous débrouiller seul (sans que je ne vous dise rien) et je vous mets des atuces au cas où vous ne trouvez pas (dans ce cas ce ne serait pas anormal).

Indication n°1 :

Quel est le problème en fait ? C'est que les paramètres pris en compte par la fonction Text ne sont plus valides. Pourquoi ? Parce que A est devenu supérieur à 63 ou inférieur à 1 ou parce que B est devenu supérieur à 127 ou inférieur à 1.

Indication n°2 :
Deux solutions s'offrent donc à vous :
- soit vous faites réapparaître le symbole de l'autre côté de l'écran (une fois qu'il est sorti d'un côté)
- soit vous faites en sorte que la fonction ne réponde plus lorsque le symbole sort de l'écran.
A vous de choisir.

Voici la solution :

Code : Basic CASIO
```
Cls
63 -> B
33 -> A
0 -> X
While 0<1
Cls
Text A, B, "x"
Getkey -> X
X=27 => B+1 -> B
X=38 => B-1 -> B
X=28 => A-1 -> A
X=37 => A+1 -> A
A=128 => 1->A
A=0 => 127->A
B=64 => 1->B
B=0 => 63->B
WhileEnd
```

J'ai utilisé la solution où lorsque le symbole sort d'un côté de l'écran il réapparait de l'autre (que je trouve plus correcte). Si vous avez trouvé un code où l'autre marche, c'est bon !

Pourquoi le symbole ne s'affiche pas lorsque nous allons trop à droite ou trop en bas de l'écran ?

Normal, nous avons oublié de prendre en compte la largeur et la hauteur de notre symbole (3 pixels sur 3). Et bien résolvons ce problème en remplaçant dans le morceau de code que nous venons de rajouter (les 4 lignes de condition) les 127 par des des 124 (car 127-3=124) et les 63 par des 60 (car 63-3=60). Et voilà ça devrait marcher à la perfection maintenant ...

Il resterait le problème du clignotement à résoudre mais vous n'avez pas (encore) assez de compétences pour le résoudre. Nous pourrions le faire mais ce ne serait pas une bonne idée de ma part de vous montrer comment faire car ça vous ferait utiliser un code qui ne serait pas du

tout optimisé et vous prendriez de mauvaises habitudes... Donc nous nous arrêtons là pour ce TP.

On ne pourrait pas l'enrichir un peu ?

Ah oui, j'allais oublier. Si vous vous sentiez d'attaque vous pourriez créer un autre symbole fixe sur l'écran que vous devriez aller "manger" avec votre premier symbole (une sorte de "jeu du serpent"). Je vais vous donner quelques indications.
Comme vous ne savez pas encore utiliser la fonction qui tire un nombre au sort (car la position de notre symbole fixe devra être aléatoire), je vais vous donner les deux lignes de code dont vous aurez besoin (en effet, une pour l'abscisse et une autre pour l'ordonnée).
Les voici donc :

Code : Basic CASIO
```
Int 60Ran#+1 -> D
Int 124Ran#+1 -> C
```

Pour les petits curieux, je ne vous expliquerai pas le fonctionnement de cette focntion pour ne pas troubler votre esprit de jenue prorammeur. Nous la verrons dans quelques temps (en effet elle est assez complexe).
Et maintenant afin que vous ne fassiez pas un programme qui n'est que bidouillage, je vais vous indiquer une nouvelle fonction qui sera très utile et qui est très simple : And.
Voici son chemin d'accès : [OPTN], [F6], [F6], [F4] (logic) , [F1] (And). Elle vous permettra de lier deux éléments d'une condition. Un exemple de code pour que vous compreniez (il n'a rien à voir avec l'exercice en cours) :

Code : Basic CASIO
```
A=3 And B=4 => 5->C
```

Explications : Il faut que les deux conditions soient respectées pour que C prenne la valeur 5. Avouez tout de même que son utilisation est très simple. Nous l'étudierons plus précisément (ainsi que ses copines Or et Not) dans quelques chapitres.

Et bien vous êtes parti ! Je ne vous donne aucune autre indication ! Je ne vous donne pas non plus de solution : si vous voulez le faire, vous devez le faire tout seul ! Lorsque vous programmerez, je ne serai pas là pour vous guider et vous donner les solutions. Il faut prendre les bonnes habitudes dès maintenant. Si jamais vous bloquiez, posez la question dans les commentaires, je répondrai.
Bonne chance !

Les cercles et les courbes

Nous allons maintenant voir comment tracer des formes un peu plus complexes grâce à certaines nouvelles fonctions.

Les cercles

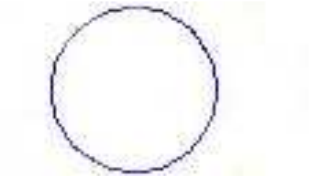
Cette partie est encore assez facile. En effet, une fois de plus, il existe une fonction effectuant tout le travail à votre place : la fonction Circle. Voici la syntaxe de cette dernière :
Circle <abscisse>,<ordonnée>,<rayon du cercle>

Notons que tous les paramètres sont définis par rapport à la configuration de la fenêtre d'affichage (ViewWindow) et non en fonction des pixels (comme les fonction Text, Pxl …).
Le chemin d'accès de cette fonction est : [Shift], [F4] (Sketch), [F6], [F3] (Crcl).
Voici un petit exemple pour mieux comprendre (nous travaillons toujours avec le réglage que nous avons défini dans le premier chapitre de cette partie).

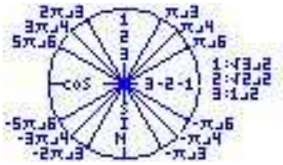
Code : Basic CASIO
```
Cls
Circle 64, 32, 25
```

La calculatrice va donc nous afficher un cercle de rayon 25 pixels (car nous sommes dans une fenêtre d'affichage où chaque point est associé à un pixel) au centre de l'écran.

Voici une image qui va vous montrer ce que vous devriez avoir à l'écran :



Et puis voici ce que vous devriez pouvoir afficher avec un peu de code en plus :



Avouez que là il y a une utilité (si vous êtes dans une classe "inférieure" à la première vous ne comprendrez probablement rien à ces notations) . Ca devient intéressant !

Les courbes (et les droites)

Si vous avez un niveau en mathématiques inférieur à celui de la 3ème, je vous déconseille de lire ce chapitre car vous n'y comprendriez pas grand chose. En outre, il faudrait même avoir un niveau supérieur à la seconde générale. Il existe une fonction : Graph Y=, qui permet de tracer des droites et des courbes en fonction de leur équation …

Les droites

Si vous avez un niveau de fin de collège alors, vous pouvez lire cette sous-partie. En effet, vous devez savoir, qu'à n'importe quel nombre X, on peut avoir par une équation une image de ce nombre. En 3ème, on ne voit que des exemples basiques de droites. Il est vrai que nous avons déjà une fonction permettant de tracer une droite (même deux) mais dans le but de faire un cours complet, je vous dit tout ! (on ne sait jamais, ça pourrait vous servir).
La syntaxe de cette fonction pour tracer une droite sera donc :
Graph Y= ax+b

Notons que a et b sont deux réels car Y= ax+b est l'équation d'une droite.
Le chemin d'accès de cette fonction est : [Shift], [F4] (Sketch), [F5] (GRPH), [F1] (Y=).
Comme exemple, traçons la droite d'équation Y= 3x+5

Code : Basic CASIO
```
Cls
Graph Y= 3x+5
```

Vous noterez sûrement que cette fonction dépend du ViewWindow. Je vous l'accorde, dans ce cas, cette fonction ne vous sera guère utile.

Les courbes

Mais si vous êtes dans l'enseignement secondaire, cette fonction aura pour vous une utilité. En effet, de cette manière, nous ne pouvons pas tracer que des droites mais aussi des courbes …
De la même manière, il vous suffit d'entrer à la suite de la fonction Graph Y=, l'équation de la courbe.
Graph Y=<équation de la courbe>

Pour bien comprendre, nous allons tracer la courbe de la fonction carré :

Code : Basic CASIO
```
Cls
Graph Y=x²
```

Et voilà, le tour est joué … Dans notre repère, ceci n'est pas idéal, mais il suffit de choisir d'autres fonctions ou simplement de choisir des fonctions associées (niveau 1ère générale, toutes sections) à celles que l'on veut afficher.

Les domaines de définition

Si vous ne souhaitez voir affichée sur votre écran qu'une partie de la courbe (ou de la droite), il existe un moyen de définir le domaine …
Pour ceci, nous utilisons les crochets :
Graph Y=<équation de la droite>,[<Xmin>,<Xmax>]

Ces derniers se trouvent sur votre clavier : [Shift], [+] ([ ) ou [-] ( ]).

Prenons un exemple : tracer la courbe de la courbe d'équation Y=√(x-20)+30 sur [20,40]

Code : Basic CASIO
```
Cls
Graph Y= √(x-20)+30,[20 ,40]
```

Note : j'ai choisi cette équation en fonction du repère qui à chaque pixel associe des coordonnées donc prenez celui-là. Sinon, vous pourriez ne rien avoir d'afficher à l'écran.

Et bien voilà, pour ceux qui ne possèdent pas la couleur sur leur calculatrice, ce chapitre s'arrête ici : vous avez acquis assez de connaissances sur les graphismes pour l'instant. Nous verrons tout de même de nouvelles techniques et fonctions plus compliquées sur les graphismes dans le chapitre suivant.

La couleur

On fait le tri

En effet, si vous ne possédez pas de calculatrice intégrant la couleur (bleu, vert et orange), ne prenez pas la peine de lire cette partie, elle ne serait utile qu'à votre culture générale dans le domaine CASIO …
Les principaux modèles concernés sont les Graph 65 et les Graph 80. Il est par ailleurs possible que d'autres modèles de chez CASIO intègrent cette fonction (je ne connais pas tout).
Je vous préviens tout de suite, ne vous attendez pas à quelque chose d'extraordinaire : ce ne sont que quelques nuances, rien de plus mais ça permet d'avoir quelque chose de plus esthétique et de plus lisible.

Mais si vous possédez une Graph 35(+), une astuce a été découverte afin que vous puissiez jouer avec les teintes.

Comment ça ?

Je ne l'ai jamais essayé mais j'en ai souvent entendu parler (je ne possède pas de Graph 35(+)). Tout ce que je sais, c'est que grâce à cette astuce, vous pourrez avoir différents niveaux de bleu (bleu clair, bleu normal (celui de la CASIO) et bleu foncé). Je crois que le rendu final n'est pas mal (pour une calculatrice). Mais l'avantage est surtout que ces teintes s'utilisent de la même manière que les couleurs sur les autres CASIO (de manière aussi simple). Vous trouverez toutes les explications nécessaires ici. Je vous mets tout de même en garde devant cette manipulation car elle est assez risquée. Mais sachez que si jamais il y a un problème, vous pourriez toujours retrouver les paramètres d'origine en faisant un RESET à l'arrière de

votre calcualtrice. Et bien voilà si vous effectuez cette manipulation, vous pourrez aussi suivre ce chapitre. Heureux ?

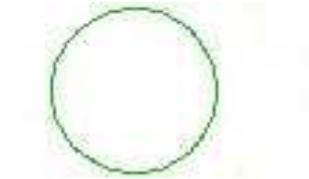
Comment faire ?

Et bien il existe des fonctions pour chaque couleur (c'est aussi simple que ça). Les trois fonctions sont Blue, Orange et Green. Je ne vous donne pas leurs correspondances, je pense (et j'espère) que vous les avez comprises. Par contre, je vais vous donner la syntaxe, bien que vous pourriez la deviner :
<Fonction couleur> <fonction graphique> <paramètres de la fonction graphique>

Prenons un petit exemple afin de vous éclaircir. Nous allons tracer un cercle vert de rayon 25 au milieu de l'écran.

Code : Basic CASIO
```
Cls
Green Circle 64,32,25
```

Facile n'est-ce pas ? Voici ce que vous devriez obtenir :



Pour le orange nous allons faire quelque chose d'un peu plus évolué : nous allons colorer le fond de l'écran en orange. Pour ce faire nous allons utiliser les fonctions Horizontal ou Vertical que vous connaissez déjà.
Je vous laisse faire ce petit programme (très court : quelques lignes) mais faites le avec votre tête (réfléchissez). Ne le faites surtout pas de manière "débile". Je vous fais confiance. Voici ce que vous devrez obtenir :

Solutions :

Code : Basic CASIO
```
ViewWindow 1, 127, 1, 1, 63, 1
Cls
For 1->X To 63
```

Orange Horizontal X
Next

ou

Code : Basic CASIO
```
ViewWindow 1, 127, 1, 1, 63, 1
Cls
For 1->X To 127
Orange Vertical X
Next
```

Ce n'était pas bien dur ? Entrainez vous à refaire certains exercices que nous avons fait sur les graphismes en ajoutant des couleurs : cela vous permettra de bien tout assimiler.

Attention

Comme ces fonctions n’existent pas sur les autres modèles de calculatrices graphiques, ces fonctions ne seront pas analysées par la calculatrice (celle qui ne possède pas les couleurs ou les niveaux de bleu) et la fonction couleur sera remplacée dans le code par un @ auquel la calculatrice ne fera pas attention. Le bleu, l’orange et le vert seront donc affichés en bleu (car l’unique couleur des autres Graph xx est le bleu). Faites donc bien attention !!!! Ce serait dommage que vos graphismes ne ressemblent plus à rien une fois transférés sur une autre Graph xx. Pour éviter ce problème, je vous conseille, soit de créer des programmes directement compatibles, soit de créer deux versions de vos programmes (une couleur et une monochrome).
Je vais vous donner un exemple grâce à un programme que j'ai réalisé l'année dernière : "A Prendre ou à Laisser".

Voici une image de la version couleur de mon programme :

Et voici ce qu'elle donnerait sur une calculatrice n'intégrant pas les couleurs :

Par Ilae pour planete-casio.fr

Vous ne voyez aucun problème ? Et bien pourtant il y en a un. Comparez avec cette image qui est celle que j'ai refaite pour les possésseurs de calculatrice n'intégrant pas la couleur :

Vous voyez maintenant ?
Note : le problème ne vient pas des chiffres effacés à gauche de l'écran ou du numéro de boîte différent.

Le problème est le mot "Boite" qui n'est pas affiché dans l'image qui était à la base prévue pour une calculatrice couleur.
Mais la différence est minime ?
Certes mais un programmeur doit être très minucieux dans son travail. Donc on fait bien !

Il existe encore une astuce par rapport aux couleurs mais nous la verrons dans le chapitre prochain … (je ne veux pas vous embrouiller).

Maintenant que vous avez de bonnes bases en graphismes et en programmation (pure), vous devriez pouvoir commencer à faire de jolis petits programmes. Essayez car ça vous permettra de ne pas oublier tout ce que vous avez appris jusqu'à maintenant et de l'appliquer sur un exemple concret. Bonne chance !

Partie 3 : Quelques techniques plus avancées

Certes vous êtes maintenant capables de créer de vrais programmes mais il vous manque encore certaines choses. En effet, vous vous rendrez vite compte que certaines choses vous manquent (fonctions mathématiques par exemples). Mais dans cette partie, vous apprendrez aussi de nouvelles techniques plus compliquées qui vous permettront d'accélérer ce que vous pouvez faire avec ce que vous savez aujourd'hui. Et sur une calculatrice, ces histoires de vitesse ne sont pas négligeables.

Les fonctions mathématiques

Voici une des rares parties de ce chapitre que vous arriverez à assimiler facilement car par la suite, elles se compliqueront …
Si vous avez essayé de créé des petits programmes, vous avez dû vous rendre compte qu'il vous manquait certaines choses : en effet ces fonctions sont très pratiques et quasiment indispensables. Nous allons donc étudier les principales :

La valeur absolue

Si jamais vous avez eu besoin de celle là, il est vrai que grâce à votre ingéniosité et aux conditions, vous avez pu la remplacer. Mais puisqu'elle existe, pourquoi ne pas s'en servir ?
Tout d'abord, il est possible que certains d'entre vous ne connaissent pas cette fonction (on l'utilise en $1^{ère}$ mais je crois qu'elle est évoquée en $3^{ème}$). Je vais tout de même vous expliquer rapidement son fonctionnement : c'est très simple.

Soit X.
Si X>0 alors la valeur absolue de X est (X).
Si X<0 alors la valeur absolue de X est (-X).

En fait, pour n'importe quel nombre, la valeur absolue de ce nombre, est ce nombre sans son signe.
La valeur absolue de X est notée : |X|
Quelques exemples :
|-5| = 5
|7| = 7
|15000| = 15000
|-3000000| = 3000000

|0| = 0

Cela peut vous paraître débile et inutile mais vous vous trompez …
Passons maintenant à la partie programmation … (et oui quand même)
La fonction valeur absolue est Abs et a pour chemin d'accès : [Optn], [F6], [F4] (num), [F1] (Abs).
Voici donc la syntaxe (qui est évidente) :
Abs <nombre>

Il est vrai que si nous ne prenons que des nombres ça n'a aucun intérêt mais je vous rappelle que les variables, les listes et les matrices contiennent des nombres. Voici donc des exemples pour comprendre :

Code : Basic CASIO
```
Abs –5
Abs G -> G
Abs Mat B[5,2] -> Mat B[5,2]
Abs List 1[2] -> List 1[2]
```

La première ligne comporte l'exemple le plus débile que j'ai trouvé car j'espère que vous savez calculer une valeur absolue tout seul et que vous n'allez pas utiliser une ligne de code pour mettre ceci (vous apprendrez par le suite, qu'elles sont précieuses).
La deuxième ligne nous montre la valeur absolue d'une variable qui est réassignée à cette même variable.
La troisième ligne nous montre la valeur absolue du nombre de la case 5,2 de la matrice B qui est de nouveau assigné à cette case.
La quatrième ligne nous montre la valeur absolue du nombre de la 2ème ligne de la liste 1 qui est de nouveau assigné à cette case.
J'espère que ceci ne vous pose aucun problème.

Partie entière/partie fractionnaire

Voici quelques fonctions qui vous seront très utiles par la suite et qui ont l'avantage d'être très simples d'utilisation. Vous allez le comprendre par vous même.

Int

Comme indiqué dans le titre, cette fonction permet d'obtenir la partie entière d'un nombre. La syntaxe est très simple, je pense que vous pouvez la deviner mais je vous la donne quand même :
Int <nombre>

Grâce à cette fonction, nous aurons donc la partie entière du nombre suivant cette fonction. Le chemin d'accès de cette dernière est le suivant : [Optn], [F6], [F4] (num), [F2] (Int).
Voici un petit exemple pour bien comprendre :

Code : Basic CASIO
9,8 -> A
Int A -> B
B¤

J'espère que vous avez compris : nous assignons la valeur 9,8 à la variable A puis nous assignons la partie entière de cette dernière à la variable B. Enfin nous affichons la valeur de B qui sera de 9.

Intg

La fonction Intg est une fonction mathématique qui arrondi un nombre par défaut (en dessous) à 1 près.
Voici la syntaxe :
Intg <nombre>

Et voici son chemin d'accès : [Optn], [F6], [F4] (num), [F5] (Intg).
Vous noterez que cette fonction n'a un intérêt que dans le négatif car comme la fonction Int tronque, elle arrondi par défaut dans le positif.

Frac

Cette fonction est quasiment la même que la fonction Int. En effet la syntaxe et le principe sont les même. Elle permet d'obtenir la partie fractionnaire d'un nombre. Je vous redonne donc la syntaxe (même si il vous suffirait de regarder au-dessus) :
Frac <nombre>

Je ne vous explique pas le fonctionnement, c'est le même que précédemment. Voici le chemin d'accès : [Optn], [F6], [F4] (num), [F3] (Frac).
Voici donc un petit exemple pour bien comprendre :

Code : Basic CASIO
9,8 -> A
Frac A -> B
B¤

J'espère que vous avez compris : nous assignons la valeur 9,8 à la variable A puis nous assignons la partie fractionnaire de cette dernière à la variable B. Enfin nous affichons la valeur de B qui sera de 0,8.

Trigonométrie

Si vous avez un niveau inférieur à la 4ème, la lecture de cette partie ne vous sera pas obligatoire. Si vous êtes en 4ème, seul la partie sur le cosinus nécessitera votre attention. Si

vous avez un niveau supérieur à la 3ème, toutes les parties devraient vous intéresser. Toujours est il que si vous n'avez pas un niveau assez élevé, l'utilisation de ces fonctions ne vous sera que très restreinte car vous ne leur connaîtrez pas d'autres utilités que dans les triangles rectangles, voire les lois de Descartes (en physique).
Les trois fonctions ont exactement la même syntaxe et s'utilisent exactement de la même manière mais je vais vous faire un cours complet qui vous paraîtra sûrement redondant mais ce sera plus simple de lecture ainsi.

Cosinus

Cette fonction se trouve sur le clavier en dessous des flèches de direction et est nommée « cos ». Voici sa syntaxe :
cos <nombre>

Je ne donne pas d'exemple pour l'instant car il faudrait que je vous indique quelque chose avant (que j'annoncerai à la fin).

Sinus

Cette fonction se trouve sur le clavier en dessous des flèches de direction et est nommée « sin ». Voici sa syntaxe :
sin <nombre>

Tangente

Cette fonction se trouve sur le clavier en dessous des flèches de direction et est nommée « tan ». Voici sa syntaxe :
tan <nombre>

Attention

Il existe plusieurs façons de mesurer un angle donc la calculatrice vous propose un réglage. En effet, vous pouvez mesurer en degrés, en radians (niveau supérieur à la 1ère Générale) et en grade (rarement utilisé). Vous comprenez bien que le cosinus d'un angle exprimé en degré (dans votre tête) alors que votre calculatrice est programmée en degrés vous donnera un résultat non-cohérent avec notre énoncé. Pour ce faire, il faut régler tout ça :
[Shift], [Menu] (Setup), [F1] (Angl) puis choisissez la mesure d'angle dont vous avez besoin : [F1] (Deg) pour les degrés, [F2] (Rad) pour les radians ou [F3] (Gra) pour les grades.
Aucune syntaxe n'est requise : il vous suffit de mettre le type de mesure puis d'aller à la ligne.
Dans d'autres menus, pour changer ces paramètres, il vous suffit de faire [Shift], [Menu] (Setup) puis de vous diriger grâce aux flèches de direction afin de trouver la rubrique Angle et de choisir la mesure dont vous avez besoin.

Inverses

Afin de retrouver les mesures des angles, il vous suffira d'appuyer sur [Shift], juste avant d'effectuer la pression sur la touche trigonométrique.

Vous aurez noté que cela s'utilise de la même manière que dans le menu destiné à faire des calculs.

Les Pictures

Nous revenons un peu dans les graphismes … Vous aurez sûrement remarqué que l'affichage des graphismes sur les Graph xx est assez long (sauf sur la Graph 85 (SD)). Il existe donc un système de capture d'écran qui vous permet de les rétablir par la suite.
Ce chapitre risque d'être assez complexe pour vous utilisateurs car les manière de pratiquer dépendent des modèles. Je possède la Graph 65 et la Graph 85 donc pourrais vous décrire précisément le mode opératoire de ces dernières (ainsi que de la 35 car elle est similaire à la 65). Je vais donc le décrire cas par cas pour être précis.
Mais tout d'abord, je vais vous expliquer plus précisément le fonctionnement (je ne compte pas le répéter dans chaque partie). Il faudra copier une image dans la mémoire puis vous pourrez la coller plus tard sur un écran vierge (ou déjà occupé par d'autres graphismes). Ce modèle est assez similaire à celui des copier-coller sous Windows.

Graph 65 et 35 (+)

Sur ces modèles, vous avez le droit d'avoir en mémoire six images d'un coup.
Je vais vous donner le chemin d'accès de la fonction Copier puis celui de la fonction Coller.
Copier : [OPTN], [F6], [F6], [F2] (Pict), [F1] (Sto) puis choisissez le numéro de sauvegarde (de 1 à 6) grâce aux touches du clavier.
Coller : [OPTN], [F6], [F6], [F2] (Pict), [F2] (Rcl) puis choisissez le numéro d'où vous avez sauvegardé l'image (de 1 à 6) grâce aux touches du clavier.
Note : dans d'autres menus (où l'écran graphique est utilisé), il se peut que vous deviez utiliser les touches [F1], [F2], [F3], [F4], [F5] et [F6] afin de choisir la picture à coller ou à copier.

Il n'y a pas vraiment de syntaxe : il vous suffira de presser la touche correspondante puis d'aller à la ligne. Selon si vous copiez ou collez, la calculatrice affichera respectivement StoPict ou Rcl Pict

Graph 85 (SD)

Sur ces modèles, vous avez le droit d'avoir en mémoire vingt images d'un coup.