

1. LES LABELS.

M.P.I. 7 LES BOUCLES.

1.1. L'INSTRUCTION.

Les labels sont des étiquettes que l'on place dans le programme. A n'importe quel moment, en rappelant l'étiquette, on retourne à l'endroit du programme où se trouve l'étiquette. Une même étiquette ne doit jamais être utilisée 2 fois dans un même programme. Rassurez-vous vous n'épuiserez pas le stock d'étiquettes fournies. Il y en a plus qu'il n'en faut: tous les nombres de 0 à 99 plus les 26 lettres et la lettre teta (vous pouvez utiliser les lettres comme label et comme variable cela n'a aucun rapport) quand il y en a plus, il y en a encore, vous pouvez mettre 2 lettres ensemble ou avec les chiffres, de A0 à teta teta !! Il faut juste ne pas mettre 2 fois la même étiquette.

Voilà la syntaxe :

: Le programme	
: Lbl 5	Etiquette 5
: Série d'instruction à l'intérieur de l'étiquette	
: Stop	Fin de Etiquette 5
: Poursuite du programme	
: Goto 5	Va à l'étiquette 5
: Poursuite du programme	

Explications :

1.2. PREMIERE APPLICATION: LE MENU - DONNER PLUS DE CHOIX.

On se rappelle l'instruction **Input**. Nous devons utiliser l'instruction **Input** avec l'instruction **If Then Else** pour simplement demander une question qui ne peut avoir pour réponse que la commande Oui ou Non.

L'instruction Menu permet de faire la même chose mais surtout proposer plus que deux choix. Il suffit de regarder les menus proposés par la calculatrice. L'inconvénient de notre instruction Menu, c'est qu'elle ne propose pas plus de 7 choix possibles.

Pour illustrer notre propos, nous allons créer un programme, appelé «Menu», qui a pour objectif, de proposer divers choix de réponses, connaissant la valeur de A et Z.

```
: ClrHome
: Fix 0
: Normal
: Input "Entrer la valeur de Z",Z
: Input "Entrer la valeur de A",A
: Clrhome

: Menu ("Atome", "Protons", 1, "Nucléons", 2,
"Electrons", 3, "Neutrons", 4)

: Lbl 1
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Protons")
: Stop

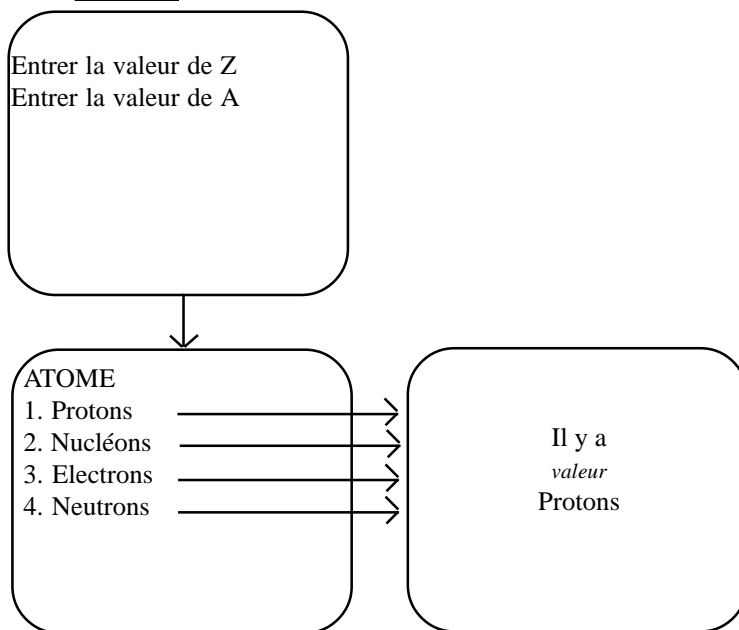
: Lbl 2
: Output (3, 6, "Il y a")
: Output (4, 8, A)
: Output (5, 6, "Nucléons")
: Stop

: Lbl 3
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Electrons")
: Stop

: Lbl 4
: A - Z -> X
: Output (3, 6, "Il y a")
: Output (4, 8, X)
: Output (5, 6, "Neutrons")
: Stop

: Stop
```

A l'écran



Remarque.

Le titre du menu ne peut contenir que 16 caractères. Quand on utilise l'instruction **Input**, on peut l'employer de manière à ce que la question posée par l'instruction **Input** semble plus longue que 16 caractères. De sorte que l'on doit parfois réduire au strict minimum la question posée par le menu. Dans le cas présent il aurait fallu poser la question «Quelques caractéristiques de l'atome», mais la question «Atome» est en elle-même compréhensible.

Il est à noter que les boucles Lbl 1 à Lbl 4 se terminent par l'instruction **Stop**, ce qui n'est pas le cas pour la dernière étiquette Lbl 5. C'est un des cas où l'instruction **Stop** est utilisée, lorsqu'on se trouve au milieu du programme. Si cette instruction **Stop** n'est pas utilisée avant d'entamer une nouvelle étiquette Lbl 2, le programme affichera toutes les réponses alors que l'objectif était d'afficher uniquement une seule réponse.

On n'a pas besoin de l'instruction **Stop** à la fin de l'étiquette Lbl 3 car c'est de toute façon la fin du programme et qu'il doit de toute façon s'arrêter.

1.3. DEUXIEME APPLICATION: LE RENVOI A UNE RECHERCHE D'UNE INSTRUCTION.

On peut, dans un programme, répéter plusieurs fois la même instruction. Pour éviter d'avoir à répéter les instructions, il suffit de renvoyer le programme, à la recherche de ces instructions, que l'on aura regroupées ailleurs dans le programme.

Dans notre programme, nous avons une «petite lourdeur». En effet, dès que nous avons fait le choix de faire afficher le nombre de protons, par exemple, dès que cette opération s'est affichée, si nous désirons qu'il s'affiche une autre valeur (le nombre de nucléons, ou d'électrons...), il nous faut alors passer par l'écran qui redemande les valeurs de A et de Z, alors qu'elles n'ont pas changé. Nous aimerions passer outre cet écran et revenir directement à l'écran menu.

```

: ClrHome
: Fix 0
: Normal
: Input "Entrez la valeur de Z",Z
: Input "Entrez la valeur de A",A
: Lbl M
: Clrhome

: Menu ("Atome", "Protons", 1, "Nucléons", 2,
"Electrons", 3, "Neutrons", 4)

: Lbl 1
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Protons")
: Pause
: Goto M
: Stop

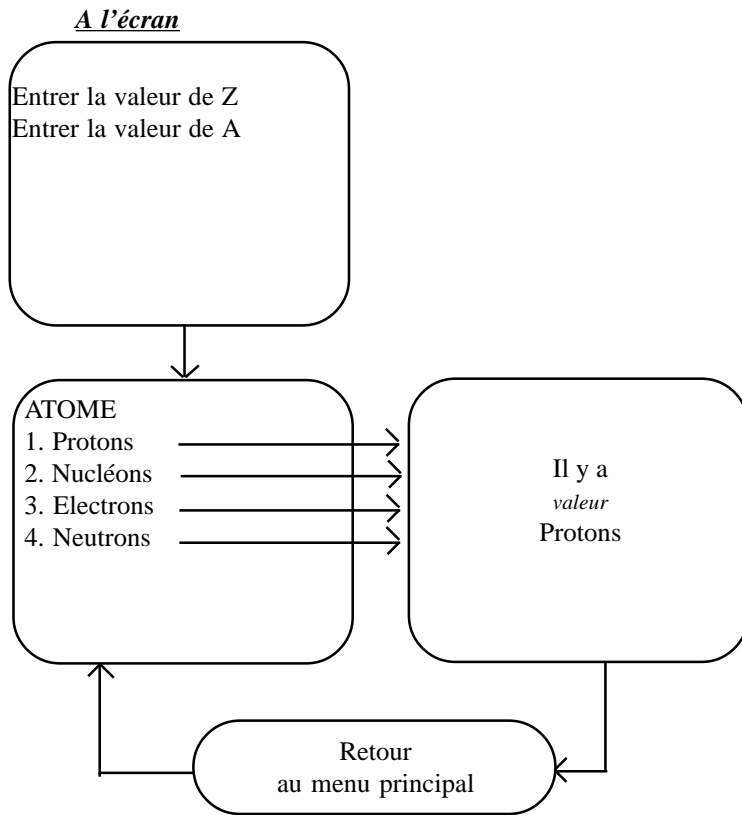
: Lbl 2
: Output (3, 6, "Il y a")
: Output (4, 8, A)
: Output (5, 6, "Nucléons")
: Pause
: Goto M
: Stop

: Lbl 3
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Electrons")
: Pause
: Goto M
: Stop

: Lbl 4
: A - Z -> X
: Output (3, 6, "Il y a")
: Output (4, 8, X)
: Output (5, 6, "Neutrons")
: Pause
: Goto M
: Stop

: Stop

```



Remarque.

On a rajouté dans chaque Lbl, les instructions: : Pause
: Goto 8

Afin de permettre à l'utilisateur le temps de lire la réponse. En touchant une touche du clavier, l'utilisateur relance le programme qui va revenir au menu principal.

Il a fallu ajouter avant l'instruction Menu, une étiquette Lbl 8 qui est le point de retour du programme.

Nous avons fait attention à ne pas reprendre même étiquette déjà utilisée notamment dans le menu.

Conclusion.

Le programme fonctionne, mais nous sommes à nouveau devant un petit problème: nous observons bien le retour à l'écran menu, mais nous n'avons pas le temps de lire la valeur qui s'affiche. A peine affichée, la valeur s'efface et l'on retourne à l'écran menu. Il nous faut donc prévoir une pause dans le programme, qui laisserait au joueur le temps de lire l'écran, avant de retourner automatiquement à l'écran menu.

La calculatrice a une instruction Pause, mais l'utilisateur se doit d'appuyer sur la touche Enter pour lancer la suite des instructions du programme... ce qu'il n'est pas censé savoir.... Ce que nous désirons, c'est que le programme lui-même laisse quelques secondes à l'utilisateur pour lire l'écran.

Un autre moyen est de créer une boucle d'attente, une boucle d'instruction qui prendrait un certain temps à la calculatrice, de quoi "l'occuper", avant qu'elle n'enchaîne sur la suite du programme, à savoir le retour à l'écran menu. Une boucle d'instruction "inutile" pour notre programme, mais utile pour laisser le temps de lire la réponse.

L'instruction FOR va nous y aider.

2. L'INSTRUCTION FOR.

Une commande très utile qui peut être utilisée pour des opérations très diverses les unes des autres.

La commande For permet un retour de boucle jusqu'à une certaine valeur défini dans sa ligne de programme. Cette instruction est utilisée aussi bien pour compter ou effectuer une même action plus d'une fois. Que faire en effet, si on veut, à l'intérieur du programme, afficher le même texte sur huit lignes différentes ? Nous pourrions utiliser l'instruction Output huit fois... mais cela «consomme» de la mémoire, ce qui «coûte» cher !! Nous pouvons par contre utiliser l'instruction For. La commande For utilise toujours une variable.

2.1. SANS L'INSTRUCTION FOR.

```
: 0 -> X
: ClrHome
: Lbl 1
: X + 1 -> X
: Output (X,2, «Coucou»)
: If X < 8: Goto 1
: Stop
```

Explications :

Mémoire 0 dans la variable X
Efface l'écran principal
Début de l'étiquette 1
Ajoute 1 à la variable X et réenregistre cette nouvelle valeur dans X
Affiche «Coucou» à la ligne X, colonne 2
Si X < 8, alors retourne à l'étiquette 1
Fin du programme.

Ce programme n'utilise pas l'instruction For, mais indique ce que nous devrions faire, si nous ne possédions pas cette instruction For, car cette instruction fait exactement la même chose.

2.2. AVEC L'INSTRUCTION FOR.

Avec la commande For voilà ce que ça donne:

```
: ClrHome
: For (X,1,8)
: Output (X,2, «Coucou»)
: End
: Stop
```

Explications :

Efface l'écran principal
X va varier de la valeur 1 à 8
Affiche «Coucou» à la ligne X, colonne 2
Fin de l'instruction For
Fin du programme.

Quelques remarques.

Quand X varie de 1 à 8, la calculatrice va afficher "Coucou" sur chaque ligne. Au départ X à la valeur 1, de sorte que la première ligne où "Coucou" s'affiche est la 1^{ère} ligne de l'écran. Puis X prend la valeur 2, de sorte que la phrase s'affiche sur la seconde ligne, et ainsi de suite....

L'instruction Stop à la fin du programme n'est pas nécessaire. Le programme, en son absence, se serait malgré tout arrêté après avoir effectué complètement l'instruction For..

Quand on lance le programme, on a l'impression que toutes les lignes s'affichent d'un seul coup, mais en fait ce n'est pas ainsi que cela se fait... La calculatrice affiche chaque ligne l'une après l'autre... mais comme le programme est assez court, la calculatrice effectue cette instruction très rapidement... trop rapidement pour observer l'écriture successive ligne après ligne.

Sauter une ligne sur deux ?

```
: ClrHome
: For (X,1,8,2)
: Output (X,2, «Coucou»)
: End
: Stop
```

Explications :

Efface l'écran principal
X va varier de la valeur 1 à 8 **avec un incrément de 2.**
Affiche «Coucou» à la ligne X, colonne 2
Fin de l'étiquette
Fin du programme.

Le programme précédent est très similaire à la première approche de l'instruction For... mais une petite différence (de taille) est à noter.

Quand on lance le programme, on notera que la phrase "Coucou" s'affiche une ligne sur 2..... Le "2" indique à la calculatrice de "sauter" une ligne sur 2 entre les lignes 1 et 8. Si on veut, on aurait pu prendre un incrément de 3.

Dans le premier programme, nous aurions pu ajouter un "1" après le 8 de l'instruction For et nous aurions eu le même résultat visuel. Ce "1" ne présente donc aucun intérêt lorsqu'on désire faire un incrément de 1.

Effectuer une décroissance ?

```

: ClrHome
: For (X,6,1,-1)
: Disp X
: End
: Stop
    
```

Explications :

Efface l'écran principal
 X va varier de la valeur 6 à 1 **avec un incrément de -1**.
 Affiche X
 Fin de l'étiquette
 Fin du programme.

Il affiche donc simplement les valeurs de X par décroissance de 6 à 1. A noter que nous devons placer l'indication "-1" pour que cela marche. Autant dans le second programme on avait aucun intérêt de préciser "1", car c'est implicite, autant "-1" n'est pas implicite.

2.3. APPLICATION: TEMPS DE PAUSE.

Nous disposons maintenant de cette instruction FOR. Pour créer une pause «artificielle» il suffit de demander à la calculatrice d'opérer une série d'instruction «inutile» qui ne s'affichera pas sur l'écran, mais pour lesquelles la calculatrice devra prendre un certain temps à les effectuer avant de pouvoir poursuivre les instructions suivantes du programme.

```

: For (X,1,500)
: End
    
```

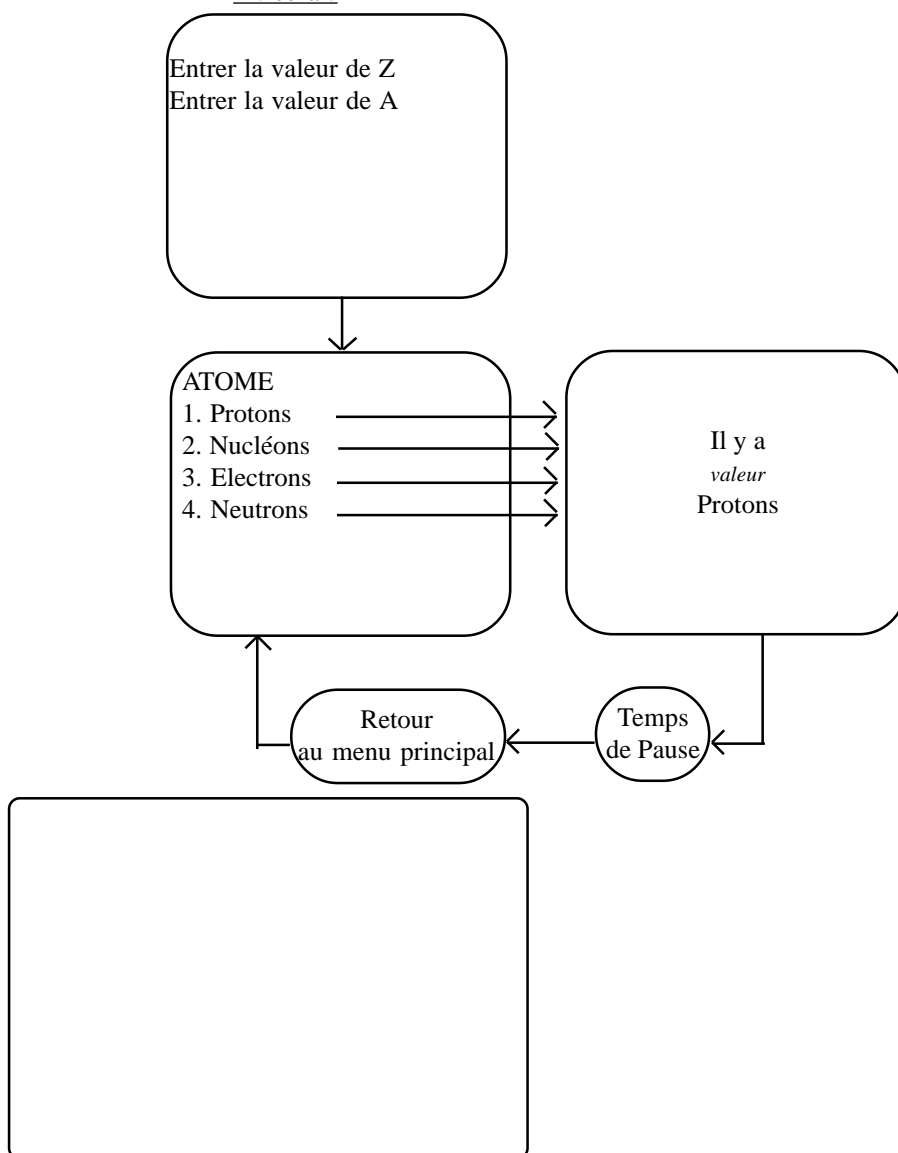
Explications :

X va varier de la valeur 1 à 500 «inutilement», mais cette variation de X va prendre un «certain temps» à la calculatrice, un temps qui sera exploité par le joueur pour lire l'écran qui s'affiche.
 Fin de l'étiquette

```

: ClrHome
: Fix 0
: Normal
: Input "Entrer la valeur de Z",Z
: Input "Entrer la valeur de A",A
: Lbl M
: Clrhome
: Menu ("Atome", "Protons", 1, "Nucléons", 2,
"Electrons", 3, "Neutrons", 4)
: Lbl 1
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Protons")
: Goto J
: Stop
: Lbl 2
: Output (3, 6, "Il y a")
: Output (4, 8, A)
: Output (5, 6, "Nucléons")
: Goto J
: Stop
: Lbl 3
: Output (3, 6, "Il y a")
: Output (4, 8, Z)
: Output (5, 6, "Electrons")
: Goto J
: Stop
: Lbl 4
: A - Z -> X
: Output (3, 6, "Il y a")
: Output (4, 8, X)
: Output (5, 6, "Neutrons")
: Pause: Goto J
: Stop
: Lbl J
: For (X,1,500)
: End
: Goto M
: Stop
: Stop
    
```

A l'écran



Remarque.

Nous avons volontairement changé le Goto M de chaque Lbl du menu, pour renvoyer à une étiquette instruction Lbl J en fin de programme, puisque cette opération est répétée dans chaque Lbl du menu. Alors pourquoi le réécrire à chaque fois, puisqu'il suffit de renvoyer à cette étiquette, qui elle même renvoie à Lbl M.

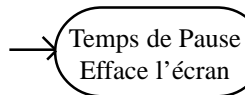
3. POUR CONCLURE QUELQUES PROGRAMMES QUI UTILISENT L'INSTRUCTION FOR.

3.1. LE PROGRAMME PRESENTATION.

En guise d'application finale de cette instruction For, nous allons créer un programme Présentation.

```
: ClrHome
: For (X,1,16)
: Output (2,X, «->»)
: For (D,1,100)
: End
: End
: Output (3,3, «Programme»)
: Output (4,3, «Réalisé»)
: Output (5,3, «Par Eric»)
: For (X,16,1,-1)
: Output (6,X, «->»)
: For (D,1,100)
: End
: End
: For (D,1,500)
: End
: ClrHome
: Prgm Question
: Stop
```

A l'écran



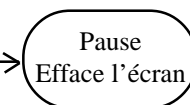
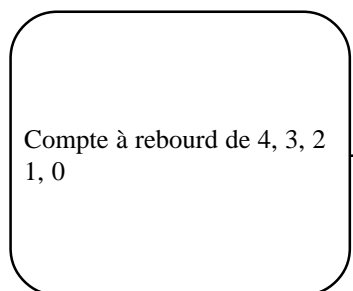
```
ATOME
1. Protons
2. Nucléons
3. Electrons
4. Neutrons
```

Les pointillés s'affichent lentement de la gauche à droite pour 1^{ère} ligne et de droite à gauche pour la 2^{nde} ligne.

Si on désire plus de temps de pause, il suffit d'augmenter le second nombre. Si on désire moins de temps de pause, il suffit de diminuer ce même nombre.

3.2. LE PROGRAMME FIN.

Pour terminer notre «jeu», nous pouvons faire une page de fin.



```
: Fix 0
: Normal
: For (X, 4, 0, -1)
: Output (4, 8, X)
: For (D, 1, 350)
: End
: End
: ClrHome
: Output (4, 4, "Bye Bye")
: For (D, 1, 350)
: End
: Stop
```