CHOISIR LE TYPE DE BOUCLE

Il existe trois types de boucles généralement employés en algorithmique:

les boucles "Pour", les boucles "Répéter … Jusqu'à " et les boucles " Tant que" [1].

Ces trois types peuvent être traduits dans la plupart des langages de programmation et sur les calculatrices. La boucle "Tant que" pourrait suffire dans tous les cas, mais, selon les situations, on peut simplifier les algorithmes en employant l'un ou l'autre des autres types.

En effet les boucles "Tant que" nécessitent obligatoirement une initialisation préalable, et leur condition d'arrêt s'exprime par une négation: Tant que la condition **n**'est **pas** réalisée, on répète les instructions concernées. Autrement dit, la boucle "Tant que" est un peu plus délicate à exploiter que les deux autres, elle peut donner lieu à plus d'erreurs ou d'oublis.

Quelques critères pour choisir la boucle la mieux adaptée:

- La boucle "Pour " s'emploie quand on peut exprimer à l'avance le nombre de répétitions:

Par exemple : Affichage des n premier carrés des entiers naturels non nuls:

```
Saisir n
Pour i variant de 1 à n
    Afficher n^2
Fin du pour
```

Avantages : l'initialisation et l'incrémentation de i sont intégrées dans l'écriture de la première ligne de la boucle: "Pour i variant de 1 à n". On a donc deux lignes de moins à écrire dans le programme, avec autant de risques d'erreurs ou d'oublis en moins. Pour s'en convaincre, on peut traduire l'exemple ci-dessus avec "un tant que". Pour ce type de boucle on parle aussi de boucle « définie ».

- La boucle "Répéter" s'emploie quand on ne peut pas exprimer à l'avance le nombre de répétitions à effectuer dans la boucle, mais on veut au moins un passage dans la boucle.
  En effet dans les boucles du type "Répéter", la condition d'arrêt est examinée après le passage dans la boucle
  Ce type de boucle fait partie des boucles dites« indéfinies », tout comme les boucles Tant Que.

Par exemple pour simuler une caisse de magasin qui calcule la somme S à payer: on arrête l'addition si le prix x rentré est 0.
Ici, on va donc rentrer au moins un prix.

[1] On peut aussi obtenir des boucles avec des instructions du type LBL et GOTO, à proscrire, car difficiles à contrôler dès que les programmes s'allongent un peu.

```
S prend la valeur 0
Répéter
    Saisir  x
    S prend la valeur S + x
Jusqu'à ce que x=0
```

Avantage de "Répéter": formulation très intuitive, condition d'arrêt simple à formuler (pas de négation), l'initialisation peut parfois être intégrée, comme dans l'exemple ci-dessus. Cette boucle est à privilégier par rapport au "Tant que", si l'on a le choix entre les deux (par exemple dans l'algorithme de dichotomie).
On peut transcrire l'algorithme ci-dessus avec une boucle "Tant que", et comprendre tout de suite le surcroît de complication.
Curieusement: ce type de boucle ne semble pas exister en Python (à vérifier), alors qu'il existe dans la plupart des autres langages, soit sous forme Repeat … Until, Do… While[2], Loop…, ou autres.
Ce type de boucle fait partie des boucles « indéfinies ».

.

- La boucle "Tant que" s'emploie quand <u>on ne peut pas exprimer à l'avance le nombre de répétitions</u> à effectuer dans la boucle, <u>ce passage dans la boucle n'étant pas obligatoire.</u>
  En effet, pour les boucles du type "Tant que", la condition d'arrêt est examinée <u>avant</u> de rentrer dans la boucle.

Par exemple : L'utilisateur choisit un nombre entier n, l'algorithme donne le plus petit entier supérieur ou égal à n qui soit divisible par 7:

```
Saisir n
Tant que n  n'est pas divisible par 7
        n prend la valeur n+1
 Fin du Tant que
 Afficher n
```

Ici, si n est lui-même divisible par 7, la boucle n'effectuera aucun traitement, et l'on passera directement à l'affichage de n.

Avantage de "Tant que": tous les types de boucles peuvent s'exprimer uniquement avec "Tant que", mais quand ce n'est pas bien adapté, cela complique un peu l'algorithme, et cela augmente les risques d'erreurs.

[2] Il existe une variante de la boucle "Répéter Jusqu'à , qui est "Répéter … Tant que, on la trouve par exemple sur les calculatrices Casio: Do….. Lp While, Lp étant l'appréciation de Loop (boucle en français). Bien penser dans ce cas que la condition d'arrêt est une négation.