

TP3 Python : les boucles « Tant que »...



La syntaxe de la boucle « Tant que » :

```
while condition :  
    instruction1  
    instruction2  
    etc.  
suite du programme
```

Ne pas oublier les tabulations !

Les **conditions** s'écrivent : *variable* *symbole* *variable ou nombre ou texte ou calcul*

où *symbole* est un des symboles suivants :

< > <= >= == != in



Exercice 1 : la punition

Bart doit régulièrement recopier un texte un certain nombre de fois.

Le texte change chaque semaine (oui, il est collé toutes les semaines...) et le nombre de fois qu'il doit le recopier change aussi.

Écrivez un programme qui demanderait à Bart la phrase à recopier, le nombre de fois qu'il doit la recopier et qui fait la punition à sa place...

Testez ce programme bien sûr.



Exercice 2 : code secret

Samuel a oublié son code pin. Il a droit à 5 essais, après quoi son téléphone sera bloqué !

Le code oublié est 1234 (dur à oublier pourtant...).

Écrivez et testez un programme qui demandera à Samuel son code et bloquera son téléphone au bout de cinq erreurs (votre programme affichera simplement « Téléphone bloqué » dans ce cas).



Exercice 3 : à la recherche d'une formule magique

Certains racontent que le jeune Carl Friedrich Gauss avait un jour été puni et devait calculer la somme de tous les entiers de 1 à 100 pendant sa récréation.

Le jeune homme, un peu plus doué en maths que la moyenne, a trouvé une formule et terminé le calcul rapidement, ce qui lui a permis de profiter de sa récréation.

1°) Écrivez un programme qui calcule la somme des entiers de 1 à n où n est choisi par l'utilisateur.

2°) Testez ce programme avec différentes valeurs de n et essayez de trouver la formule qu'aurait (re)trouvé Gauss.



Le hasard avec Python

Il est souvent utile de créer des nombres aléatoires, que ce soit pour des simulations informatiques ou pour des jeux. Python ne sait pas faire cela à la base, on doit donc ajouter à Python de nouvelles fonctions.

Il y a pour cela deux instructions possibles, à taper au début du programme :

soit :

```
>>> from random import random
```

qui veut dire : dans la bibliothèque random, aller chercher la fonction random.

soit :

```
>>> import random
```

qui veut dire : aller chercher toutes les fonctions de la bibliothèque random.

Vous pouvez ensuite utiliser la fonction random() ainsi :

```
>>> random.random()
```

(utilisez la flèche vers le haut du clavier pour relancer cette commande)

Si vous avez importé la bibliothèque complète, vous aurez aussi accès à la fonction randint(), qui choisit un nombre entier au hasard, par exemple pour simuler un lancer de dé, tapez :

```
>>> random.randint(1,6)
```



Exercice 4 : un jeu

Écrivez et testez un programme qui :

- choisit un nombre entier au hasard ;
- demande au joueur de le trouver ;
- le joueur a le droit à autant d'essai qu'il le souhaite ;
- la machine dit à chaque essai si le nombre mystère est plus petit ou plus grand que le nombre choisi par le joueur.

On peut à la fin attribuer un score au joueur en fonction du nombre d'essai qu'il a réalisé.



Exercice 5 : problème du duc de Toscane

Le duc Cosme II de Médicis, protecteur de Galilée lui a un jour posé le problème suivant :

« J'ai remarqué que quand je lance trois dés, j'obtiens plus souvent la somme 10 que la somme 9, alors qu'il y a six façons d'obtenir l'un ou l'autre !».

1°) Quelle sont les six façons d'obtenir un 9 ? les six façons d'obtenir un 10 ?

2°) Écrivez un programme qui permet de vérifier l'affirmation du duc de Toscane.