NOTIONS FONDAMENTALES – 3 - BOUCLES

RESUME DE COURS - EXERCICES

Version sept 2018

5. ++ Boucles

boucle tant que (while)

On veut afficher tableau de conversion des fahrenheits en celsius pour des fahrenheits allant de 0 à 300 de 20 en 20.

```
Programme listeFahr
    fahr = 0
    Tant que fahr < = 300
        celsius = 5 * (fahr – 32) / 9
        afficher (fahr, celsius)
        fahr = fahr + 20
    Fin tant que
Fin
```

Cinquième exemple – boucle pour (for)

On veut afficher les 10 premiers entiers et leurs carrés en partant de 1

```
Programme lesCarrés
    Pour i de 1 à 10
        afficher (i, i*i)
    Fin pour
Fin
```

Transformation d'une boucle for en boucle tant que, Transformation d'une boucle tant que en boucle for

Les débranchements : Break, Continue, Return

Le goto : débranchement non-structuré : INTERDIT !!!

Simulation

	Fahr	Fahr <=300	Celsius
Fahr ← 0	0		
		Vrai	
Celsius ← 5 * (fahr – 32) / 9			-17,6

Fahr ← fahr + 20	20		
		Vrai	
Celsius ← 5 * (fahr – 32) / 9			-6,7
Fahr ← fahr + 20	40		
		Vrai	
Celsius ← 5 * (fahr – 32) / 9			4,4
	Etc.		
Fahr ← fahr + 20	280		
		Vrai	
Celsius ← 5 * (fahr – 32) / 9			137,8
Fahr ← fahr + 20	300		
		Vrai	
Celsius ← 5 * (fahr – 32) / 9			148,9
Fahr ← fahr + 20	320		
		faux	

Python de base

```
# Programme listeFahr - boucle while
fahr =0
while fahr <= 300:
    celsius = 5*(fahr-32)/9
    print("%3d fahr = %6.2f celsius" % (fahr, celsius))
    fahr =fahr + 20
```

```
# Programme listeFahr - boucle for
for fahr in range(0,301,20):
    celsius = 5*(fahr-32)/9
    print("%3d fahr = %6.2f celsius" % (fahr, celsius))
```

```
# Programme lesCarrés - boucle for
for i in range(1, 11):
    print("%2d au carré = %3d" % (i,i*i))
```

```
# Programme lesCarrés - boucle while
i=0
while i <=10:
    print("%2d au carré = %3d" % (i,i*i))
    i+=1
```

EXERCICES – SERIE 3 – BOUCLE

Méthode de base d'analyse algorithmique

La méthode de base pour écrire un algorithme suit les 4 étapes suivantes :

1. Comprendre le problème : bien lire le sujet et bien comprendre ce qu'il y a à faire.
2. Lister ce dont on a besoin pour résoudre le problème (les données) et ce qu'on va produire (les résultats) : préciser les Entrées et les Sorties.
3. Trouver un principe de résolution : se donner les grandes lignes, en français, de la méthode de résolution. Eventuellement, se donner des procédures ou des fonctions (des actions générales).
4. Ecrire l'algorithme en détail.

1-bases

Exercice 1 : table de multiplication

Écrire un programme qui affiche la table de multiplication par 7.

Faire 4 versions : avec une boucle à compteur, avec une boucle tant que, avec une boucle répéter, avec une boucle sans fin.

Exercice 2 : somme des n premiers nombres

Écrire un programme qui calcule la somme des n premiers nombres (on utilisera une boucle).

Écrire une variante qui calcule la somme des n premiers nombres pairs.

Simuler la variante pour n = 10.

Exercice 3 : recommencer le programme

Réécrire le programme précédent en effaçant l'écran au début et en donnant la possibilité à l'utilisateur de recommencer le programme ou de quitter le programme. Pour effacer l'écran, on utilise la fonction cls() ou clear(). En python, on écrit : import os ; res=os.system(« cls ») ; // cls sous windows, clear sous lynux

Exercice 4 : nombre premier

Ecrire un programme pour déterminer si un nombre est premier.

Rappel : un nombre premier n'a pas de diviseurs autre que 1 et lui-même

2-avancés

Exercice 1 : nombre parfait

Un nombre est dit parfait s'il est égal à la somme de ses diviseurs excepté lui-même. Par exemple, 6 est parfait, 6=1+2+3. 28 est parfait : 28=1+2+4+7+14

Ecrire un programme qui permet d'afficher tous les nombres parfaits plus petit qu'une valeur N.

Exercice 2 : triplet pythagoricien

Un triplé d'entiers naturels (x, y, z) est dit pythagoricien si et seulement si on a : $x^2 + y^2 = z^2$.

Ecrire un programme qui, pour une valeur de n donnée, affiche tous les triplets pythagoriciens tels que $x^2 + y^2 <= n$.

Exercice 3 : triangle pascal

Ecrire un programme qui permet d'afficher le triangle de Pascal le niveau étant fourni au départ.

Triangle de pascal :

N0 : 1
N1 : 1 1
N2 : 1 2 1
N3 : 1 3 3 1
N4 : 1 4 6 4 1
Etc.

Principe par l'exemple : pour N4 :

- Le premier vaut toujours 1,
- 4 = 3 + 1 (les valeurs sont prises en N3 au-dessus et au-dessus à gauche)
- 6 = 3 + 3
- 4 = 1 + 3
- Le dernier vaut toujours 1

Exercice 4 : pyramide d'étoiles

Ecrire un programme qui permet d'afficher une pyramide d'étoiles : en fonction du nombre de niveaux fournis.

Exemple pour 4 niveaux :

```
        *
      * * *
    * * * * *
  * * * * * * *
```

Exercice 5 : arbre d'étoiles

Modifier le programme précédent de manière à différencier la partie centrale, puis un arbre

```
          |
        * | *
      * * | * *
    * * * | * * *
  * * * * | * * * *
          |
        - | -
```

Exercice 3 :

Écrire un programme qui soit capable d'afficher un damier de 7 colonnes et de 6 lignes (l'aire de jeu du futur « puissance 4 » qu'il faudra implémenter ultérieurement.

```
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
| | | | | | | |
+-+-+-+-+-+-+-+
```

Exercice 6 : plongeur

Un plongeur avec bouteille quand il remonte à la surface doit marquer des arrêts pour décompresser : ce sont les paliers de décompression.

Le principe des paliers est que le plongeur doit faire un palier de 100 secondes à 30 mètres. De 30 mètres mètres à 0, il y a un palier tous les 3 mètres, le temps diminuant de 10 secondes à chaque fois. Au-dessus de 30 mètres, ses paliers sont augmentés de 25 secondes tous les 10 mètres, (ainsi par exemple : 150 secondes à 50 m).

On souhaite connaître, pour une profondeur donnée, le temps de remontée minimum, sachant que la vitesse de remontée est fixée à 1 m/s.

3-branchés maths

Exercice 1 : X puissance n

Écrire un programme qui calcule X puissance n, avec n entier relatif. On traitera tous les cas particuliers.

Exercice 2 : factorielle

Ecrire un programme qui permet de calculer factorielle N (N !).

Rappels : 0 ! =1, N ! = 1*2*3*…*N

Exercice 3 : suite

On désire calculer le terme d'ordre n de la suite suivante définie par :

S(0)=1, S(n)=*2F(n-1)+1

Écrire un programme qui permet de calculer F(n).

Exercice 4 : suite Fibonacci

On désire calculer le terme d'ordre n de la suite de Fibonacci définie par :

F(0)=0, F(1)=1, F(n)=F(n-1)+F(n-2).

Écrire un programme qui permet de calculer F(n).

Exercice 5 : racine de Newton

On désire calculer la racine carrée par la méthode de Newton.

Racine(A) est donnée par le calcul de la suite suivante :

S(0)=0, S(n+1) = 0,5 * (S(n) + A / S(n))

Quand deux valeurs successives de la suite sont identiques, on a trouvé la racine.

Écrire un programme qui permet de calculer racine(A) avec cette méthode.

Exercice 6 : sin(x)

Ecrire un programme qui permet de calculer le développement limité de sin x.

$\sin(x) = x - x^3 / 3! + x^5 / 5 ! - x^7 / 7 !$ etc.

La condition d'arrêt du calcul sera que le terme $x^n/n !$ soit plus petit qu'une constante EPSILON donnée. (Epsilon est la plus petite valeur possible pour un réel).