COURS:

Attention sur une ligne de programme on ne peut pas commencer où l'on veut.

(ce qui est sur une ligne n'est pas flottant)

Si la ligne précédente se termine par : le retour ne se fait pas au même niveau.

En cas de doute mettez le curseur à droite de : puis appuyer sur la touche Entrée.

Le curseur se mettra là où il faut écrire.

Thème :

- for i in range (1,n+1): veut dire << pour tout entier i allant de 1 à n >>

  (Voir feuille précédente)

  Attention: i s'arrête à (n+1)- 1

- b=int(input("Entrez un entier : ")) Cela affiche Entrez un entier

  puis attend la saisie de cet entier

  si le nombre saisi n'est pas un entier

  b sera sa partie entière en raison de int

- *if Condition:*

  .....

Si ..... :

sous entendu alors .....

(le alors c-à-d then ne s'écrit pas en Python 2.7,

le : le remplace)

(Dans le cas de plusieurs si, on met elif pour chaque

si supplémentaire)

Une Condition comportant une égalité se met avec ==

Le symbole = est donc doublé

else: *cela veut dire sinon*

... .

Attention à la présence de

:

au bout d'une ligne

L'instruction qui suit est décalée (indentation)

(vers la droite par rapport au début de la ligne du dessus)

- ** *le double astérix signifie ^*

  par exemple: x^2 *s'écrit* x**2

- * *un seul astérix mis entre deux réels est la croix × de la multiplication*

  par exemple : x^{2p} *est codé* x**(2*p)

  *2**3 = 2 x 2 x 2 = 8*

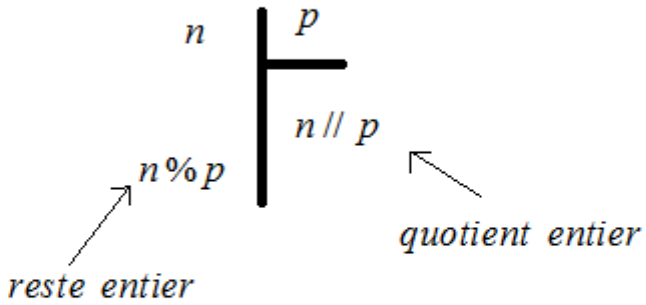
$$5**(2**3) = 5^{2^3}$$

$$5^{2^3} = 5^{2\times 2\times 2} = 5^8 = 390625$$

- *n%p* *donne le reste (entier) de la division de n par p*

Notations en Python2.7: Soit p est un entier naturel non nul .

Soit n un entier naturel.



n % p = 0 signifie que p est un diviseur de n

Par exemple : 7 // 3 donne 2

7% 3 donne 1

7 = 3 x 7 // 3 + 7 % 3

c-à-d 7 = 3 x 2 + 1

COURS:

Thème:

- from random import * *se met au dessus de* def nomprogramme():

  pour faire appel au module random

  s'il est installé

- while *Condition:* Tant que Condition :

  ....

- if *Condition:* Si Condition:

  ....

  elif *Condition:* *Si Condition:*

  ....

  else: Sinon:

  ....

- floor (*valeur*)

Donne la partie entière de la valeur

c-à-d "le plus grand entier relatif inférieur ou égal à la valeur"

En maths c'est E(valeur)

Cela nécessite de *faire appel au module math*

(Donc avant il faut mettre from math import)*

*Par exemple : from math import**

floor(5.4) = 5

floor(− 3.4) = − 4

Remarque: On peut aussi utiliser plus facilement

pour les réels positifs

int(valeur positive)

Pour les nombres réels positifs cela donne leur partie entière.

Mais attention cela ne marche pas pour les nombres réels négatifs.

(En fait cela donne la partie non décimale.)

- randint(0,100) Cela donne un entier entre 0 et 100 au hasard décidé par l'ordinateur (Cela nécessiteavant le menu from random import*)
- != Pour dire ≠ Par exemple: 6!=5 veut dire 6 ≠ 5
- *float(a)* *transforme a en un décimal*

  Par exemple: float(5) donne 5.0

  Si a est déjà un nombre décimal cela ne fait rien

  Par exemple: float(5.) donne 5.0

  float(a)/b permet d'avoir a / b comme nombre décimal .

  Par exemple: 5 / 2 donne 2

  5.0/ 2 donne 2.5

  5/ 2. donne 2.5

  float(5)/2 donne 2.5

float(5/2) = float(2) = 2.0

Si a et b ne sont pas déjà des nombres décimaux

a. / b comme a /b. donne aussi le résultat de la division

de a par b sous forme décimale

Parfois il faut avant faire appel au module decimal

Par exemple

float(11)/2 donne 5.5

Cela revient à 11.0 /2

L'ordinateur donne un décimal car 11.0 l'est

Cela peut parfois a*vant nécessiter d'écrire:*

*from decimal import**

- *random()* *donne un réel au hasard de l'intervalle [0;1[*

*Cela necessite avant from decimal import**

COURS:

Thème :

- if Condition1: *veut dire Si Condition1*

  ..Conséquence1... *alors Conséquence1*

  elif Condition2: *veut dire Si Condition2*

  ..Conséquence2 *alors Conséquence2*

  else: *veut dire Sinon*

  ..Conséquence3 *alors Conséquence3*

  (On peut utiliser plusieurs fois elif .... :)

- *float (x)* *Pour une écriture décimale de x*

  Cela nécessite un module complémentaire

- *raw_input(" texte ")* *Affiche le texte et attend la saisie d'une chaîne*

Exercice 1 : Somme des n premiers entiers

Ecrire une suite d'instructions qui saisit un entier n ≥ 1 et qui calcule la somme 1 + 2 +..... + n.

On pourra le faire de plusieurs façons.

- Soit à l'aide d'une boucle.

On a : S = 0

Puis S = S+1 Donc S = 0+1

Puis S =S+2 Donc S=0+1+2

etc .....

Enfin S= S+n Donc S=0+1+2+ ... + n

Attention: Pour un matheux S=S+1 est absurde

Cela signifie en informatique que on remplace S par S+1

En maths on mettrait: S ← S + 1

- Soit, directement, à l'aide de la formule mathématique:

1 + 2 +..... + n =[(1 + n) / 2]× n

Exercice 2 : Carrés successifs

Ecrire une suite d'instructions qui saisit deux nombres entiers n et p, puis qui fournit les valeurs de:

n^2 , $(n^2)^2$, $((n^2)^2)^2$, ... etc

c-à-d

n^2 , $n^{2\times 2}$, $n^{2\times 2\times 2}$, ... , n^{2^p}

Exercice 3 : Diviseurs d'un entier.

Ecrire une suite d'instructions qui saisit un nombre entier n et fournit tous ses diviseurs.

(on pourra utiliser l'instruction n%p qui donne le reste (entier) de la division de n par p).

REPONSE:

Il y a au moins trois façon de procéder:

1. Avec une boucle

   for i in range(1,n+1):

2. Avec la formule qui donne la somme des n premiers termes

   d'une suite arithmétique de premier terme 1 et de raison 1

3. Avec un" tant que"

   while i <n+1:

•Première méthode:

On peut envisager:

```
def test():
    n=input("jusqu'à quel rang voulez-vous calculer ? ")
    s=0
    for i in range(1,n+1):

    # attention range(1,n+1) donne i =1,2,3,...,n
       s=s+i
       print("1+...+"),(i),"=",(s)

    # Avec Python 2.5 ou 2.7 on peut mettre    print "1+...+",i,"=",s   sans parenthèses.

    # En alignant le p de print avec le f de for on obtient la dernière étape
```

On obtient par exemple:

```
>>> test()
jusqu'à quel rang voulez-vous calculer ? 10
1+...+ 1 = 1
1+...+ 2 = 3
1+...+ 3 = 6
1+...+ 4 = 10
1+...+ 5 = 15
1+...+ 6 = 21
1+...+ 7 = 28
1+...+ 8 = 36
1+...+ 9 = 45
1+...+ 10 = 55
>>>
```

REPONSE:

Première recherche:

Obtenir la liste des carrés des entiers de proche en proche.

le"carré de n", puis le carré du"carré de n" , ainsi de suite...

```
def test():
    n=input("Entrez un nombre entier n: n =  ")

    p=input("Entrez le nombre de carrés successifs à calculer : p =  ")
   for x in range(1,p+1):

  #attention for x in range(1,p+1) donne  x =1,3,...,p
       n=n**2
       print " "
      print(n)
```

Pour le faire tourner .

```
>>> test()
Entrez un nombre entier n: n =  3
Entrez le nombre de carrés successifs à calculer : p =  5

9
```

REMARQUE: Autre interprétation

Pour avoir la SOMME des termes de la forme 2^(2k) de k =1 à k =p

Voici un algorithme:

```
def somcarre():
    n=input("Entrer un entier naturel non nul  ")
    p=input("Entrer un entier naturel non nul ")
    S=0
    for k in range(1,p+1):
        S=S+(n**(2*k))
    print (" la somme"),n,"^(2*1)+ ...+ ",n,"^(2*",p,")"," =",S
```

Par exemple:

```
>>> somcarre()
Entrer un entier naturel non nul  2
Entrer un entier naturel non nul  3
 la somme 2 ^(2*1)+ ...+  2 ^(2* 3 )  = 84
>>>
```

REPONSE:

```
def diw(x):
    n=input("Entrez un nombre entier naturel non nul : ")
    for x in range(1,n+1):
    if n%x==0:

    # n%x  est le reste  de la division de n par x
    # Si le reste  de la division de n par x est 0 alors x divise n

      print( "il y a comme diviseur: " ),(x)

     # On peut suprimer les parenthèses en disant,   print "il y a comme diviseur: " ,x
    # Attention après deux points se décaler de trois blanc à la ligne
```

Pour le faire tourner:

```
>>> diw(1)
Entrez un nombre entier naturel non nul : 15
il y a comme diviseur:  1
il y a comme diviseur:  3
il y a comme diviseur:  5
il y a comme diviseur:  15
>>>
```