



Cours et exercices

Version 1.0

3ème \* Sciences Informatiques

# ALGORITHMIQUE & PROGRAMMATION

*Anis ELBAHI*

---

# 1. L'algorithmique

## 1.1 - L'algorithme

Lycée OTHMAN CHATTI M'SAKEN

2020 - 2021

Un **algorithme** est une suite (ordonnée) d'opérations (d'instructions / actions) permettant de résoudre un problème.

### Autrement dit :

*L'algorithme est la solution d'un problème informatique dans un langage naturel. Cette solution n'est pas compréhensible par l'ordinateur. Pour cela elle doit être traduite en **un langage de programmation**, ce qui donne **un programme**.*

Un **algorithme** se compose de trois grandes parties :

- les informations dont on a besoin au départ ;
- la succession d'instructions à appliquer ;
- la réponse que l'on obtient à l'arrivée.

Pour écrire un algorithme il faut respecter la convention suivante :

ALGORITHME *Nom*

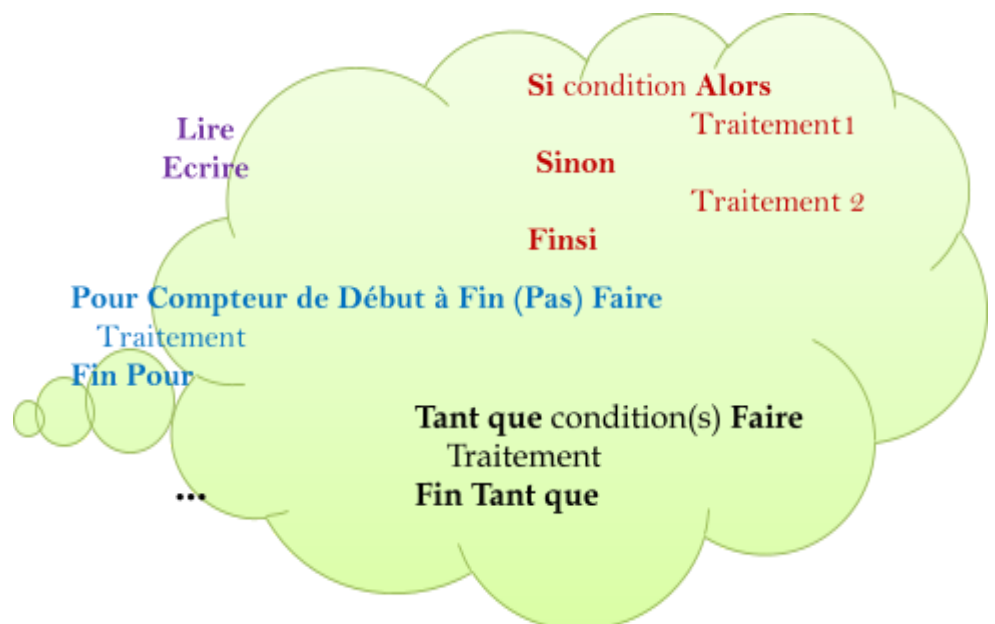
DEBUT

Traitements

FIN

Déclaration des objets

| Objet | Type / Nature |
|-------|---------------|
|       |               |



## 1.2 - La programmation

la programmation, appelée aussi codage, est l'ensemble des activités qui permettent l'écriture des programmes informatiques en utilisant un langage de programmation.

Exemples de langage de programmation : Pascal, C++, Python

## 2. Les opérations élémentaires simples

| Opération   | En algorithmique                    |
|-------------|-------------------------------------|
| Entrée      | Lire(objet)                         |
| Sortie      | Ecrire("Message",Objet, Expression) |
| Affectation | Objet $\leftarrow$ Expression       |

### Exercice 1 : (mon premier algorithme)

Rappelons que pour un rectangle :

- ☐ Le périmètre = (longueur + largeur) \* 2
- ☐ La surface = longueur \* largeur

- Donner l'algorithme nommé **Rectangle** qui permet de :
  - Lire la longueur (X) et la largeur (Y) d'un rectangle,
  - Calculer le périmètre (P) et la surface (S) du rectangle
  - Afficher les valeurs (S) et (P) trouvés
- Pour les valeurs X=2 et Y=3, quelles sont les valeurs trouvées de S et P

Solution :

#### ALGORITHME Rectangle

DEBUT

Ecrire ("donner longueur :")

Lire(X)

Ecrire ("donner largeur :")

Lire(Y)

$P(X+Y)*2$

$SX*Y$

Ecrire ("le périmètre = ", P)

Ecrire ("le surface = ", S)

FIN

S= 6 et P=10

### Exercice 2 : (Moyenne en programmation)

Donner l'algorithme nommé **Moyenne** qui permet de :

- Lire la note de contrôle 1 (NC1)
- Lire la note de contrôle 2 (NC2)
- Lire la note de synthèse (NS)
- Calculer et afficher la moyenne obtenue en programmation en appliquant la formule suivante :

$$MG = (NC1 + NC2 + NS * 2) / 4$$

**Solution :**

**ALGORITHME Moyenne**

**DEBUT**

**Ecrire** ("donner la note de contrôle 1 :")

**Lire**(NC1)

**Ecrire** ("donner la note de contrôle 2 :")

**Lire**(NC2)

**Ecrire** ("donner la note de synthèse :")

**Lire**(NS)

MG (NC1+NC2+NS\*2)/4

**Ecrire** ("la moyenne = ", MG)

**FIN**

## 3. Les structures de données

### 3.1 - Les variables

Une variable est « boîte » conçue pour contenir une information. Chaque variable doit avoir un nom pour qu'elle soit repérée par le programme. Il y a plusieurs types de variables :

- **numérique** : entier (int) , réel (float)
- **texte** : caractère, chaîne (string)
- **logique** : booléen (bool) ne peut prendre que deux valeurs : vrai ou faux



#### ATTENTION

- Affecter une valeur à une variable, c'est donner une valeur à cette variable.
- En Python il faut écrire False et non pas false, float et non pas Float, ...
- Il existe d'autres type de variables comme les tableaux, les matrices, les listes, les nombres complexes, ...

### 3.2 - Les opérateurs

Pour agir sur les variables, on peut utiliser des opérateurs qui dépendent du type de variables.  
Il existe 3 types d'opérateurs :

### Opérateurs arithmétiques

| Opération                     | En Algorithmique | En Python |
|-------------------------------|------------------|-----------|
| Addition                      | +                | +         |
| Soustraction                  | -                | -         |
| Multiplication                | *                | *         |
| Division réelle               | /                | /         |
| Division entière              | div              | //        |
| Rester de la division entière | mod              | %         |
| Exponentiation                | ^                | **        |

### Opérateurs de comparaison

| Opération             | En Algorithmique | En Python |
|-----------------------|------------------|-----------|
| Egal                  | =                | ==        |
| Différent             | ≠                | !=        |
| Strictement supérieur | >                | >         |
| Strictement inférieur | <                | <         |
| Supérieur ou égal     | ≥                | >=        |
| Inférieur ou égal     | ≤                | <=        |
| Appartient            | ∈                | in        |
| N'appartient pas      | ∉                | not in    |

### Opérateurs logiques

| Opération   | En Algorithmique | En Python |
|-------------|------------------|-----------|
| Négation    | non              | not       |
| Conjonction | et               | and       |
| Disjonction | ou               | or        |

### Exercice 3 : (application d'opérateurs)

Compléter le tableau suivant :



## ATTENTION

Il faut faire attention et bien distinguer l'instruction d'affectation = du symbole de comparaison ==.

### Exercice 4 : (fusionner deux nombres)

- Donner l'algorithme nommé fusion qui permet de saisir deux nombres **X** et **Y** de deux chiffres chacun puis de les fusionner dans un troisième nombre **R** comme le montre l'exemple suivant :

#### Exemple :

**X**=12 et **Y**=34 leur fusion donne **R**=1342.

Explication : l'entier **R** est obtenu en insérant l'entier **Y** entre les chiffres de **X**

#### Solution :

##### ALGORITHME fusion

##### DEBUT

**Ecrire** (" donner X "), lire(X)

**Ecrire** (" donner Y "), lire(Y)

a x div 10 , b x mod 10, c y div 10 , d y mod 10

R a\*1000 + c\*100 + d\*10 + b\*1

**Ecrire** (R)

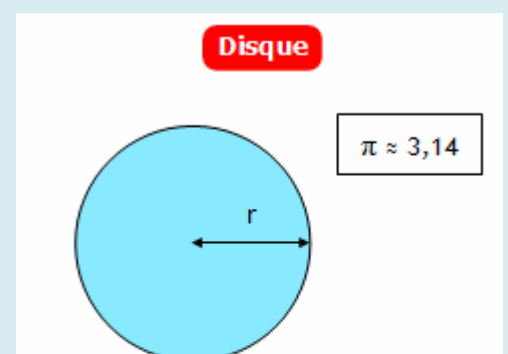
##### FIN

- Donner la traduction python de votre algorithme

### Exercice 5 : (périmètre et surface d'un disque)

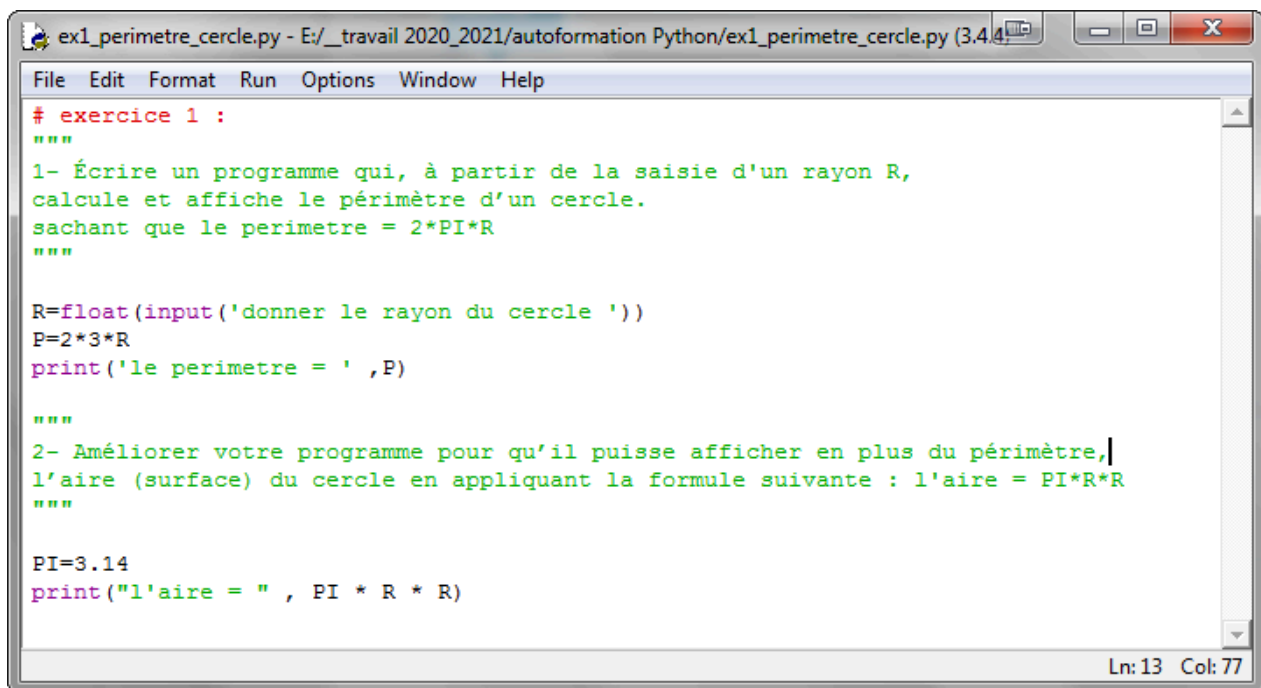
- Écrire un programme Python qui, à partir de la saisie d'un rayon, calcule et affiche **le périmètre** d'un cercle.

- Améliorer votre programme pour qu'il puisse afficher en plus du périmètre, **l'aire** (surface) du cercle en appliquant la formule suivante :



$$\text{aire} = \pi \times r^2$$

### Solution :



```
ex1_perimetre_cercle.py - E:/_travail 2020_2021/autoformation Python/ex1_perimetre_cercle.py (3.4.4)
File Edit Format Run Options Window Help
# exercice 1 :
"""
1- Écrire un programme qui, à partir de la saisie d'un rayon R,
calcule et affiche le périmètre d'un cercle.
sachant que le perimetre = 2*PI*R
"""

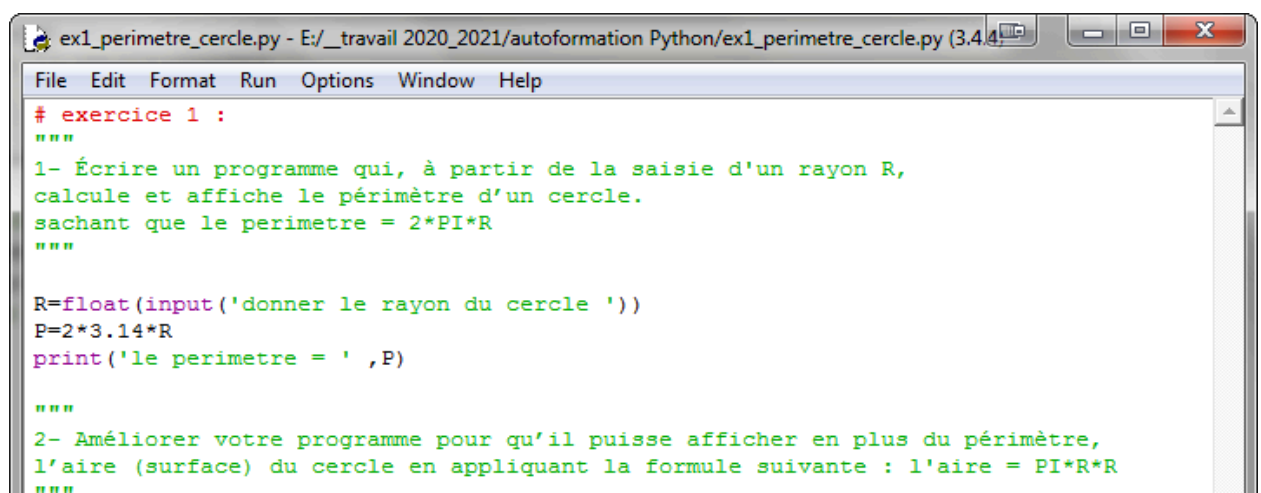
R=float(input('donner le rayon du cercle '))
P=2*3*R
print('le perimetre = ' , P)

"""
2- Améliorer votre programme pour qu'il puisse afficher en plus du périmètre,
l'aire (surface) du cercle en appliquant la formule suivante : l'aire = PI*R*R
"""

PI=3.14
print("l'aire = " , PI * R * R)

Ln: 13 Col: 77
```

Ou bien on importe la bibliothèque math



```
ex1_perimetre_cercle.py - E:/_travail 2020_2021/autoformation Python/ex1_perimetre_cercle.py (3.4.4)
File Edit Format Run Options Window Help
# exercice 1 :
"""
1- Écrire un programme qui, à partir de la saisie d'un rayon R,
calcule et affiche le périmètre d'un cercle.
sachant que le perimetre = 2*PI*R
"""

R=float(input('donner le rayon du cercle '))
P=2*3.14*R
print('le perimetre = ' , P)

"""
2- Améliorer votre programme pour qu'il puisse afficher en plus du périmètre,
l'aire (surface) du cercle en appliquant la formule suivante : l'aire = PI*R*R
"""
```

## 3.3 - Types de variables en Python

### 3.3.1 Les nombres

Il y a trois types numériques en Python :

- Le type entier `int` permet de représenter n'importe quel nombre entier, peu importe sa taille.
- Le type flottant `float` permet de représenter des nombres comportant une partie décimale (comme 3.14 ou  $2.1e-23$ ), compris entre  $10^{-324}$  et  $10^{308}$ .
- Le type complexe `complex` permet de représenter des nombres complexes, où le nombre imaginaire se note `j` (par exemple  $3.1+5.2j$ )

### 3.3.2 Les booléens

Les variables de type booléen peuvent prendre deux valeurs : **False**, **True**.

### 3.3.3 Les chaînes de caractères

#### • Présentation

Une chaîne de caractères est une séquence de caractères entre guillemets (simples ou doubles). En Python, les éléments d'une chaîne sont indexés à partir de 0 et non de 1.

**Exemple :**

```
>>> s = 'Anis ELBAHI'  
>>> print(s[0])
```

Ce bloc affiche le caractère **A**



**ATTENTION**

Une chaîne de caractères est un objet **immuable**, c'est-à-dire ses caractères ne peuvent pas être modifiés par une affectation et sa longueur est fixe.



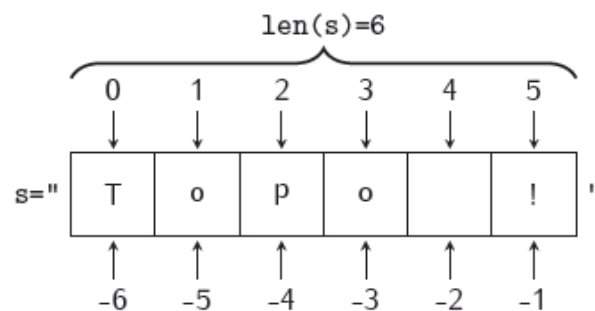
- **Manipulation d'une chaîne**

On peut extraire une sous-chaîne en déclarant l'indice de **début (inclus)** et l'indice de **fin (exclu)**, séparés par deux points : `s[i:j]`, ou encore une sous-chaîne en déclarant l'indice de début (inclus), l'indice de fin (exclu) et le pas, séparés par des deux-points : `s[i:j:k]`.

Cette opération est connue sous le nom de *slicing* (en anglais).

### Exemple :

Soit la chaîne de caractères : "Topo !"



```
>>> s='Topo !'
>>> s[2:4]
'po'
>>> s[2:]
'po !'
>>> s[:2]
'To'
>>> s[:]
'Topo !'
>>> s[2:5]
'po '
```

```
>>> s[2:9]
'po !'
>>> s[1:6:2]
'oo!'
>>> s[-4:-2]
'po'
>>> s[-1]
'!'
>>> s[-1:-7:-1]
'! opoT'
```

- **Autres opérations et méthodes associées aux chaînes de caractères**

| Opération / méthode  | Rôle                           |
|----------------------|--------------------------------|
| <code>len(s)</code>  | Donne la longueur d'une chaîne |
| <code>s1 + s2</code> | Concatène les chaîne s1 et s2  |

|                |                                                                |
|----------------|----------------------------------------------------------------|
| "x" in s       | Donne True si la chaine contient l'élément "x"                 |
| s.count("x")   | Donne le nombre d'occurrence de l'élément "x" dans s           |
| s.index("x")   | Donne l'indice de la première position de l'élément "x" dans s |
| str(n)         | <b>Transforme le nombre n en une chaine</b>                    |
| s.lower()      | Remplace les majuscules par des minuscules                     |
| s.upper()      | Remplace les minuscules par des majuscules                     |
| s.capitalize() | Transforme le premier caractère de la chaine en majuscule      |
| s.title()      | Transforme le premier caractère de chaque mot en majuscule     |

### Exercice 6 : (Changement de casse)

Soit la chaine suivante :

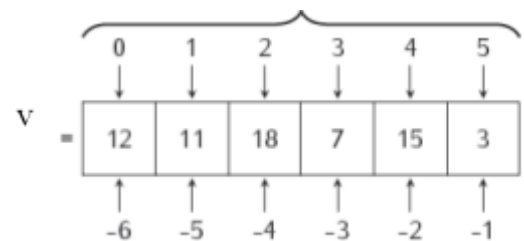
s='Programmation python'

compléter le tableau suivant par la valeur de x relative à chaque instruction :

Une liste est une suite d'objets, rangés dans un certain ordre. Une liste peut contenir des objets de types différents les uns des autres.

### Exemple :

```
t=[10, -11,205,'Anis', True, 2.5]
```



- Manipulation d'une liste (vecteur)**

Soit la liste v suivante, examiner les instructions suivantes et vérifier quelles sont correctes:

```
>>> v=[12, 11,18,7,15,3] #initialisation d'une liste v
>>> v #affichage de tous les elements de v
[12, 11, 18, 7, 15, 3]
>>> v[2:5] #affichage des elements à partir de l'indice2 à l'indice4
[18, 7, 15]
>>>
```

```

>>> v[2:]      #affichage de l'indice2 à la fin
[18, 7, 15, 3]
>>> v[:2]      #affichage dès le debut jusqu'a l'indice1
[12, 11]
>>> v[:]       #affichage de la totalité
[12, 11, 18, 7, 15, 3]
>>> v[1:5:2]    #affichage de l'indice1 à l'indice4 par pas de 2
[11, 7]
>>> v[-2:-6:-1] #affichage de l'indice-2 à l'indice-5 par pas de -1
[15, 7, 18, 11]
>>> v[-2:-6:1]  #affichage de l'indice-2 à l'indice-5 par pas de 1
[]
>>> |

>>> len(v)      #donne le nombre d'elements de v
6
>>> max(v)      #donne le maximum de v
18
>>> min(v)      #donne le minimum de v
3
>>> -10 in v    # donne vrai si -10 se trouve dans v
False
>>> -10 not in v #donne vrai si -10 ne se trouve pas dans v
True
>>> |

```

### ATTENTION

Si on tente d'extraire un élément avec un index dépassant la taille de la liste, Python renvoi un message d'erreur.

Attention : les instructions suivantes modifient la liste !

|               |                                                                           |
|---------------|---------------------------------------------------------------------------|
| a.append(x)   | ajoute l'élément x en fin de la liste a                                   |
| a.extend(L)   | ajoute les éléments de la liste L en fin de la liste a, équivaut à a + L  |
| a.insert(i,x) | ajoute l'élément x en position i de la liste a, équivaut à a[i:i]=x       |
| a.remove(x)   | supprime la première occurrence de l'élément x dans la liste a            |
| a.pop([i])    | supprime l'élément d'indice i dans la liste a et le renvoie               |
| a.index(x)    | renvoie l'indice de la première occurrence de l'élément x dans la liste a |
| a.count(x)    | renvoie le nombre d'occurrence de l'élément x dans la liste a             |
| a.sort()      | modifie la liste a en la triant                                           |
| a.reverse()   | modifie la liste a en inversant les éléments                              |

### Exercice 7 : (fonction min, max et sum sur les listes )

- Créer le tableau T suivant :

|    |   |   |     |   |
|----|---|---|-----|---|
| 10 | 5 | 2 | 199 | 4 |
|----|---|---|-----|---|

- Afficher l'élément T[3]
- Afficher le maximum du tableau
- Afficher le minimum du tableau
- Afficher la somme des éléments du tableau.

### • Manipulation d'une liste de liste (Matrice)

Les matrices peuvent être représentées comme des listes imbriquées : chaque ligne est un élément d'une liste.

```
>>> a=[ [11,12,13], [21,22,23], [31,32,33]]
>>> a
[[11, 12, 13], [21, 22, 23], [31, 32, 33]]
>>> print (a[1][2]) #afficher l'element de la deuxieme ligne troisieme colonne
23

>>> print(len(a)) #nombre de lignes
3
>>> print(len(a[0])) #nombre de colonne de la ligne1
3
```

#### Exercice 8 : (Matrice) [\[Solution\]](#)

- Soit la matrice M suivante, On demande de créer et remplir la matrice comme indiqué.

#création et remplissage de la matrice

```
M=[
[10, 2, -2 ],
[1, 2, 3],
[-15, 102, 23],
[77, 49, -9]
]
```

- Donner à chaque fois, l'instruction python permettant de :

```
ex1'_matrice1.py - E:/_travail 2020_2021/autoformation Python/ex1'_matrice1.py (3.4.4)
File Edit Format Run Options Window Help
#création et remplissage de la matrice
M=[
[10, 2, -2 ],
[1, 2, 3],
[-15, 102, 23],
[77, 49, -9]
]

#afficher la valeur 102
print(M[2][1])

#Afficher le nombre de lignes de la matrice
print('le nombre de lignes de M = ',len(M))

#Afficher le nombre de colonnes de la matrice
print('le nombre de colonnes de M = ',len(M[0]))

#Modifier la valeur qui se trouve dans M[0,2] par 6
M[0][2] =6

#Modifier la première valeur de la deuxième colonne par sa valeur absolue.
M[0][1] = abs(M[0][1])

#Afficher le contenu de la matrice M
print(M)

#Afficher la somme de 2eme colonne
print(M[0][1] + M[1][1] + M[2][1] + M[3][1])

Ln: 11 Col: 0
```

### 3.3.5 Les tuples

Pour simplifier, les tuples sont des listes qui ne peuvent pas être modifiées.

Exemple :

```
t=(12, 10, 18, 7, 15, 3, "elbahi", 1.5)
```



#### ATTENTION

Si on essaye de modifier les éléments d'un tuple on a un message d'erreur.

Toute fonction ou méthode qui s'applique à une liste sans la modifier peut être utilisée pour un tuple.

## 3.4 - La fonction range

La fonction range crée un itérateur (compteur) .

Exemple :

- **range(n)** renvoi un itérateur parcourant 0, 1, 2 ... , n - 1 ;

- **range(n,m)** renvoi un itérateur parcourant  $n, n+1, n+2, \dots, m-1$  ;
- **range(n,m,p)** renvoi un itérateur parcourant  $n, n+p, n+2p, \dots, m-1$ .

```
>>> A=range(5)
>>> A
range(0, 5)
>>> L=list(A)
>>> L
[0, 1, 2, 3, 4]
```

Pour afficher le range A, il faut créer une liste

```
>>> list(range(3,8))
[3, 4, 5, 6, 7]
```

```
>>> list(range(0,20,5))
[0, 5, 10, 15]
```

### Exercice 9 : (Devine le résultat)

Cocher la bonne réponse (vrai ou faux) pour chaque instruction

## 4. Les structures conditionnelles :

Les structures conditionnelles permettent de faire un choix parmi plusieurs,  
On peut distinguer 3 formes différentes :



Forme simple réduite

Forme simple complète

Forme généralisée

|                                                                              |                                                                                                                                    |                                                                                                                                                                                                          |
|------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> if condition_1:     instruction_1.1     instruction_1.2     ... </pre> | <pre> if condition_1:     instruction_1.1     instruction_1.2     ... else:     instruction_n.1     instruction_n.2     ... </pre> | <pre> if condition_1:     instruction_1.1     instruction_1.2     ... elif condition_2:     instruction_2.1     instruction_2.2     ... ... else:     instruction_n.1     instruction_n.2     ... </pre> |
|------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## 4.1 - Quand utiliser les structures conditionnelles ?

Une structure de contrôle conditionnelle peut être utilisée si on est obligé à tester une condition pour exécuter un traitement.

Pour certains cas, on associe à chaque traitement une condition particulière, si la condition est vraie le traitement sera exécuté et dans le cas contraire le traitement sera ignoré.

## 4.2 - Indentation

En Python (contrairement aux autres langages) c'est l'indentation (les espaces en début de chaque ligne) qui détermine les blocs d'instructions (structures conditionnelles, boucles, etc.).

### Exercice 10 : (Parité de x)

Donner l'algorithme d'un programme nommé **Parité** qui permet de :

- Saisir un entier x
- Vérifier et afficher si x est pair ou impair

**Solution :**

```

ALGORITHME Parité
DEBUT
  Ecrire ("donner x" )
  Lire(x)
  Si (x mod 2 = 0) alors
    Msg "pair"
  Sinon
    Msg "impair"
  FinSi
  Ecrire (x , " est " , Msg)
FIN

Déclaration des objets

```

### Exercice 11 : (Admis / Redouble)

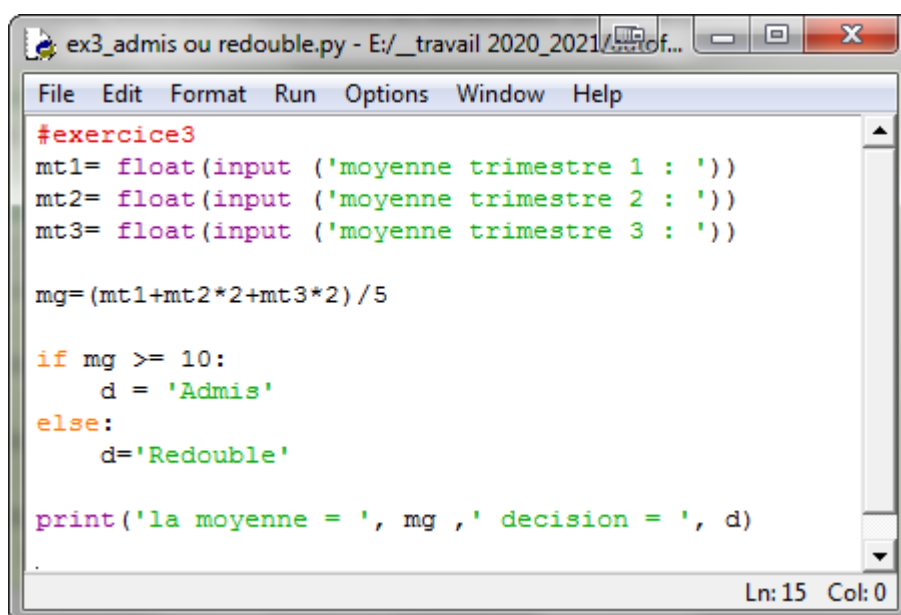
Pour savoir si un élève est **admis** ou **redoublant** il faut calculer sa moyenne générale de fin d'année en utilisant la formule suivante :

$$MG = (MT1 * 1 + MT2 * 2 + MT3 * 2) / 5$$

Avec :

- MG : **Moyenne Générale**
- MT1 : **Moyenne Trimestre 1**

Solution :



```
#exercice3
mt1= float(input ('moyenne trimestre 1 : '))
mt2= float(input ('moyenne trimestre 2 : '))
mt3= float(input ('moyenne trimestre 3 : '))

mg=(mt1+mt2*2+mt3*2) /5

if mg >= 10:
    d = 'Admis'
else:
    d='Redouble'

print('la moyenne = ', mg , ' decision = ', d)
```

Ln: 15 Col: 0

### Exercice 12 : (Equation de second degré)

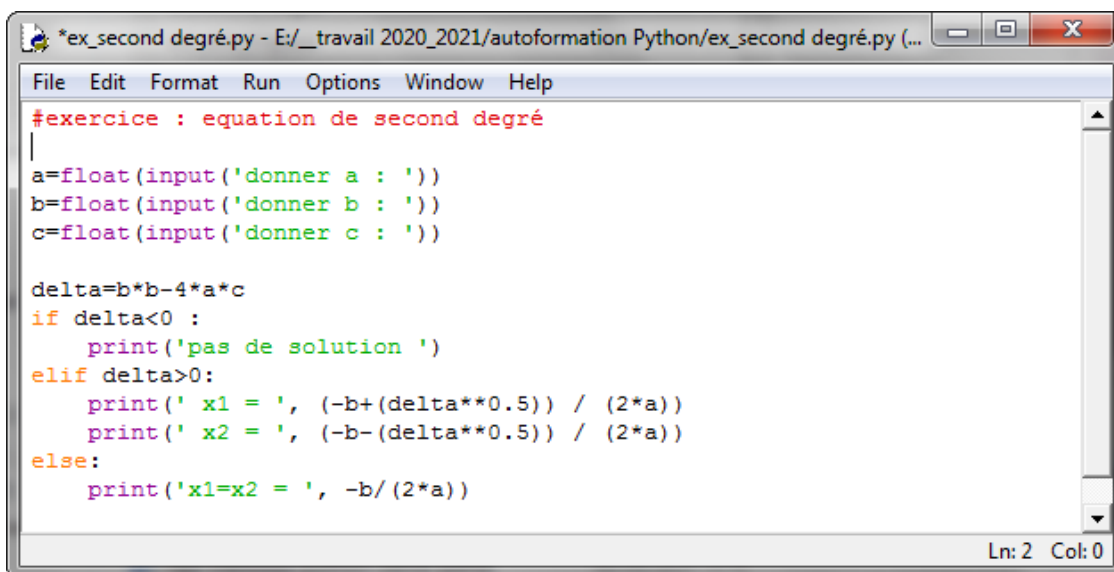
On désire faire l'algorithme puis le programme Python qui permettent de résoudre une équation de second degré après avoir saisi ces coefficients a, b et c. Sachant qu'une équation de second degré s'écrit sous la forme  $ax^2+bx+c=0$ .

Pour résoudre telle équation, il faut suivre la démarche suivante :

**NB :** Pour calculer la racine carrée s'un nombre k on utilisera la propriété . Donc au lieu d'écrire



Solution :

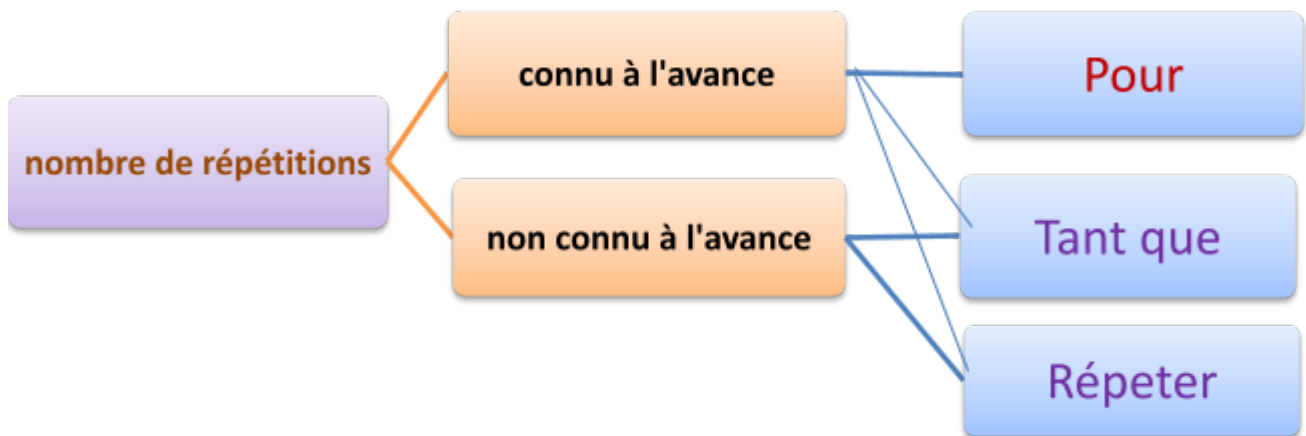


```
*ex_second degré.py - E:/__travail 2020_2021/autoformation Python/ex_second degré.py (...)  
File Edit Format Run Options Window Help  
#exercice : equation de second degré  
|  
a=float(input('donner a : '))  
b=float(input('donner b : '))  
c=float(input('donner c : '))  
  
delta=b*b-4*a*c  
if delta<0 :  
    print('pas de solution ')  
elif delta>0:  
    print(' x1 = ', (-b+(delta**0.5)) / (2*a))  
    print(' x2 = ', (-b-(delta**0.5)) / (2*a))  
else:  
    print('x1=x2 = ', -b/(2*a))  
  
Ln: 2 Col: 0
```

## 5. Les structures itératives

Les structures itératives ou structures répétitives ou boucles permettent de répéter un traitement un certain nombre de fois.

répétitions inconditionnelles



répétitions conditionnelles

|                     |                                                                         |                                                                                |
|---------------------|-------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| En<br>Algorithmique | Tant que condition(s) Faire<br>Traitement<br>Fin Tant que               | Pour Compteur de Début à Fin (Pas) Faire<br>Traitement<br>Fin Pour             |
|                     | La boucle Tant que                                                      | La boucle for                                                                  |
|                     | La boucle while                                                         | La boucle for                                                                  |
| En Python           | <pre>while condition:     instruction_1     instruction_2     ...</pre> | <pre>for target in sequence:     instruction_1     instruction_2     ...</pre> |

## 5.1 - La boucle while

La boucle **tant que ... faire** s'exécute tant que la condition est **vraie**.

### ATTENTION

Si la condition est toujours vraie et ne devient jamais fausse, le bloc d'instruction est répété indéfiniment (à l'infini) et le programme ne se termine pas. On se trouve dans une boucle « infinie ».

#### Exemple 1 :

Afficher les entiers de 1 à 15 en utilisant la boucle while.

```

*while_1_15.py - C:/elbahi_anis/while...
File Edit Format Run Options Window Help
# afficher de 1 à 15 (while)

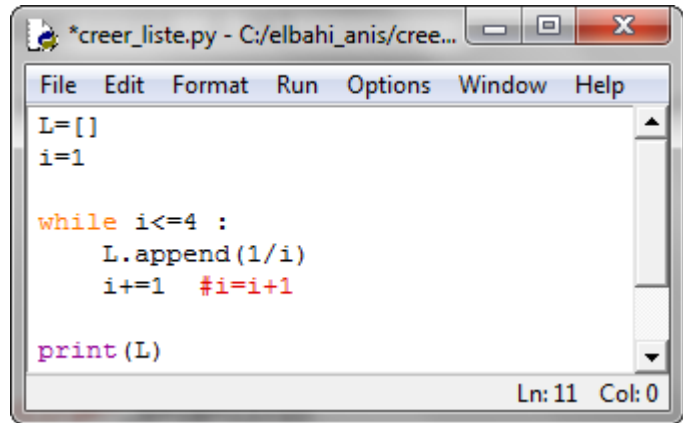
x=1
while x<=15:
    print(x)
    x=x+1
  
```

Ln: 9 Col: 0

### Exemple 2 :

En utilisant la boucle while, créer et afficher la liste suivante :

$$\left[1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}\right]$$



```
File Edit Format Run Options Window Help
L=[]
i=1

while i<=4 :
    L.append(1/i)
    i+=1 #i=i+1

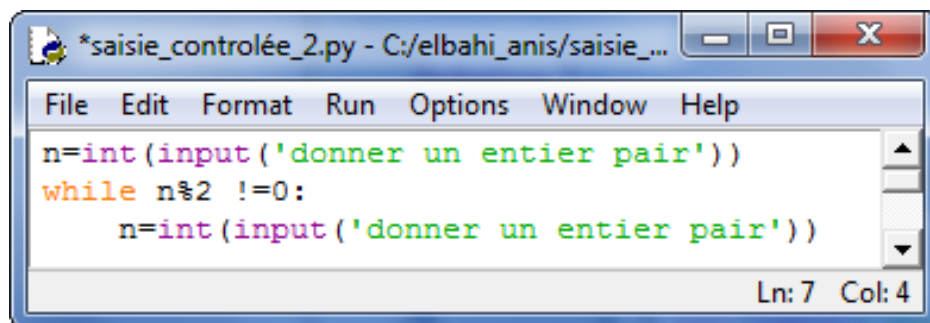
print(L)
```

Ln: 11 Col: 0

### Exemple 3 : (saisie contrôlée)

On demande de saisir un entier n pair.

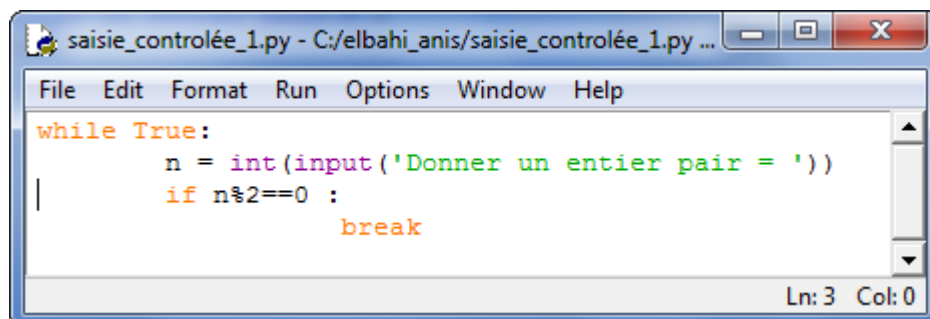
#### Solution 1 :



```
File Edit Format Run Options Window Help
n=int(input('donner un entier pair'))
while n%2 !=0:
    n=int(input('donner un entier pair'))
```

Ln: 7 Col: 4

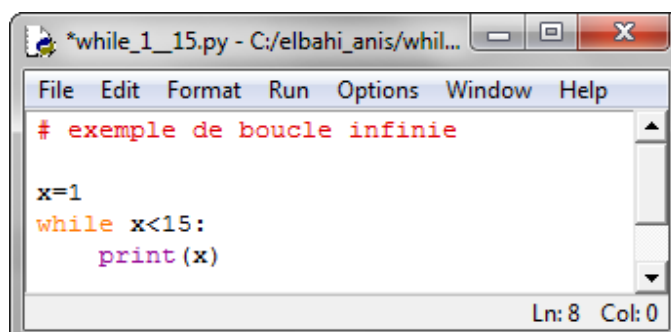
#### Solution 2 :



```
File Edit Format Run Options Window Help
while True:
    n = int(input('Donner un entier pair = '))
    if n%2==0 :
        break
```

Ln: 3 Col: 0

### Exemple de boucle infinie :



```
File Edit Format Run Options Window Help
# exemple de boucle infinie

x=1
while x<15:
    print(x)
```

Ln: 8 Col: 0

### Exercice 13 : (tournage à la main)

Soit le code Python suivant :

```
n=5
i, s = 0 , 0
while i<n :
    i=i+1
    s=s+i
print(s)
```

- Faire le tournage à la main du code précédant et donner le résultat à afficher ?

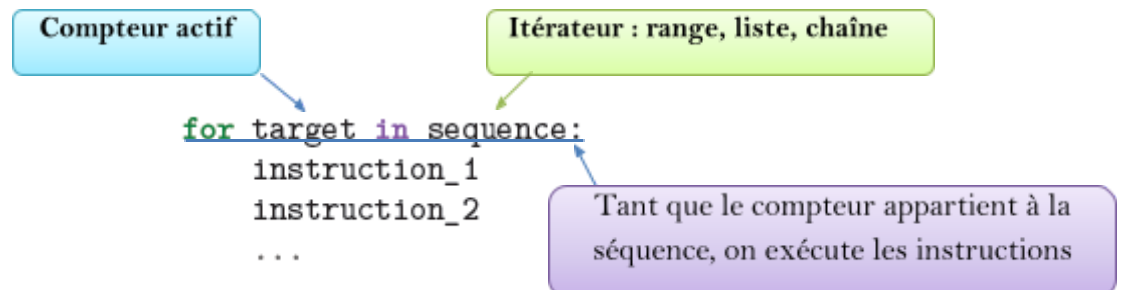
Résultat = 15

- Donner le rôle du code précédant : calculer et afficher la somme des n premiers entiers

**Remarque :** la somme des n premiers entiers =

## 5.2 - La boucle for

La boucle **pour ... faire** est utilisée lorsqu'on souhaite répéter un bloc d'instructions un nombre déterminé de fois.



Exemples :

```
#afficher les elements d'une liste
L=["Anis", "Ines", "Anas"]
for i in L :
    print(i)
```

Anis  
Ines  
Anas

```
#afficher les elements d'une chaine
ch="Anis ELBAHI"
for c in ch :
    print(c)
```

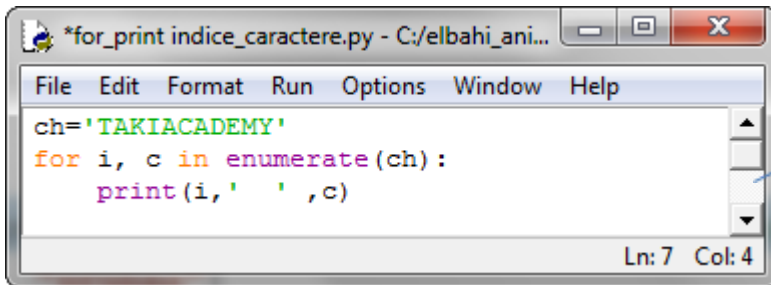
A  
n  
i  
s  
  
E  
L  
B  
A  
H  
I

```
#afficher les elements d'une range
for j in range(5) :
    print(j)
```

0  
1  
2  
3  
4

### Remarque :

Pour accéder en même temps à la position et au contenu on utilisera `enumerate(liste)`.



```
0 T
1 A
2 K
3 I
4 A
5 C
6 A
7 D
8 E
9 M
10 Y
```

### Exercice 14 : (créer une liste alphabétique)

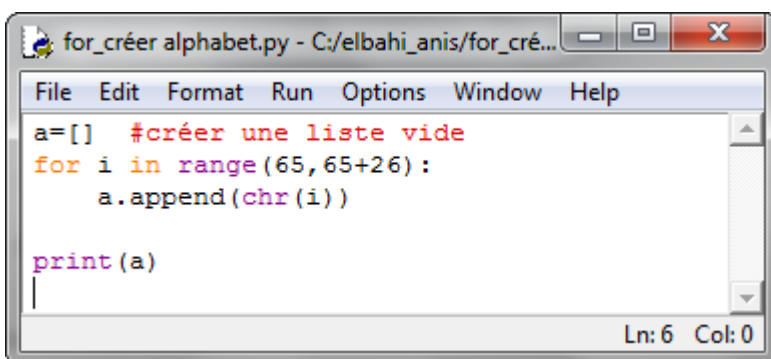
Donner un code Python qui permet de créer et afficher une liste formée par les lettres alphabétiques majuscules en utilisant la boucle for:

#### Solution :

```
a=[] # créer une liste vide
for i in range(65,65+26): # le compteur i fait le parcours de 65 à 91
    a.append(chr(i)) # à chaque itération on ajoute à la fin de la liste le caractère suivant

print(a) # afficher la liste obtenue
```

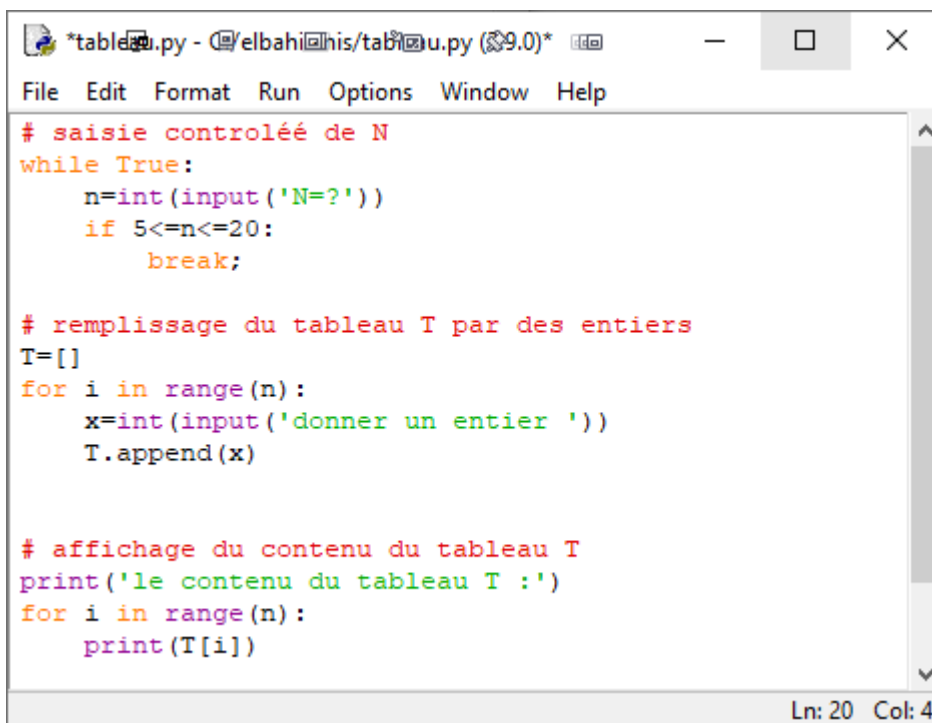
le résultat affiché :



### Exercice 15 : (remplir et afficher un tableau)

Donner un code Python qui permet de :

- Saisir  $5 \leq N \leq 20$
- Remplir un tableau T (liste) par N entiers
- Afficher le contenu du tableau T



```
*tableau.py - @/elbahi@his/tableau.py (Python 3.9.0)*
File Edit Format Run Options Window Help
# saisie contrôlée de N
while True:
    n=int(input('N=?'))
    if 5<=n<=20:
        break;

# remplissage du tableau T par des entiers
T=[]
for i in range(n):
    x=int(input('donner un entier '))
    T.append(x)

# affichage du contenu du tableau T
print('le contenu du tableau T :')
for i in range(n):
    print(T[i])

Ln: 20 Col: 4
```

### Exercice 16 : (Manipulation d'un tableau)

Donner un code Python qui permet de :

- Saisir la taille du tableau  $N$  :  $4 \leq N < 15$
- Remplir un tableau  $T$  (liste) par  $N$  entiers positifs de 3 chiffres chacun.
- Afficher tous les entiers d'Armstrong qui se trouve dans le tableau  $T$ .

**NB :** un entier est dit d'Armstrong s'il est égal à la somme des cubes de ses trois chiffres

Exemple d'entiers d'Armstrong :  $153 = 1^3 + 5^3 + 3^3$

```
*tableau.py - C:/elbahi/his/tableau.py (3.9.0)*
File Edit Format Run Options Window Help

# saisie contrôlée de N
while True:
    n=int(input('N=?'))
    if 4<=n<15:
        break;

# remplissage du tableau T par des entiers
T=[]
for i in range(n):
    while True:
        x=int(input('donner x positifs de 3 chiffres '))
        if x>=100 and x<=999:
            break
    T.append(x)

# affichage des entiers d'armstrong :
print("les entiers d'Armstrong : ")
for i in range(n):
    c=T[i]//100
    d=(T[i]%100)//10
    u=T[i]%10
    if c*c*c+d*d*d+u*u*u==T[i]:
        print(T[i])

Ln: 26 Col: 0
```

### Exercice 17 : (saisie contrôlée + chaîne palindrome)

Faire le programme Python qui permet de :

- Saisir une chaîne de caractères non vide.
- Vérifier et afficher si la chaîne saisie est palindrome ou non.

**NB :**

Une chaîne est dite palindrome s'il elle peut être lue de droite à gauche comme de gauche à droite.

**Exemple :**

RadaR, ETE, elle, AzizA, ...

```
ex-palindrome.py - C:/Users/TakiAcademy/Desktop/ex 3si palindrom
File Edit Format Run Options Window Help

#saisie controlee d'une chaine
while True:
    ch=input('donner une chaine non vide ')
    if len(ch)!=0:
        break

#verification de la chaine palindrome ou non
test=True
```



Si la chaîne doit être formée uniquement par des lettres alphabétiques majuscules.

```
*ex-palindrome.py - C:/Users/TakiAcademy/Desktop/ex 3si palindrome/ex-palindrome.py (3.9.0)*
File Edit Format Run Options Window Help
#saisie controlee d'une chaine non vide formee par des alphabets majuscules
while True:
    ch=input('donner une chaine non vide ')
    nb=0
    for i in range(len(ch)):
        if 'A'<=ch[i]<='Z':
            nb=nb+1
    if len(ch)!=0 and len(ch)==nb:
        break

#verification de la chaine palindrome ou non
test=True
for i in range(len(ch)//2):
    if ch[i]!= ch[len(ch)-i-1]:
        test=False

#affichage de resultat
if test:
    print(ch,' est palindrome ')
else:
    print(ch,'non palindrome ')

exit()
```

### Exercice 18 : (saisie contrôlée + entier premier)

Faire le programme Python qui permet de :

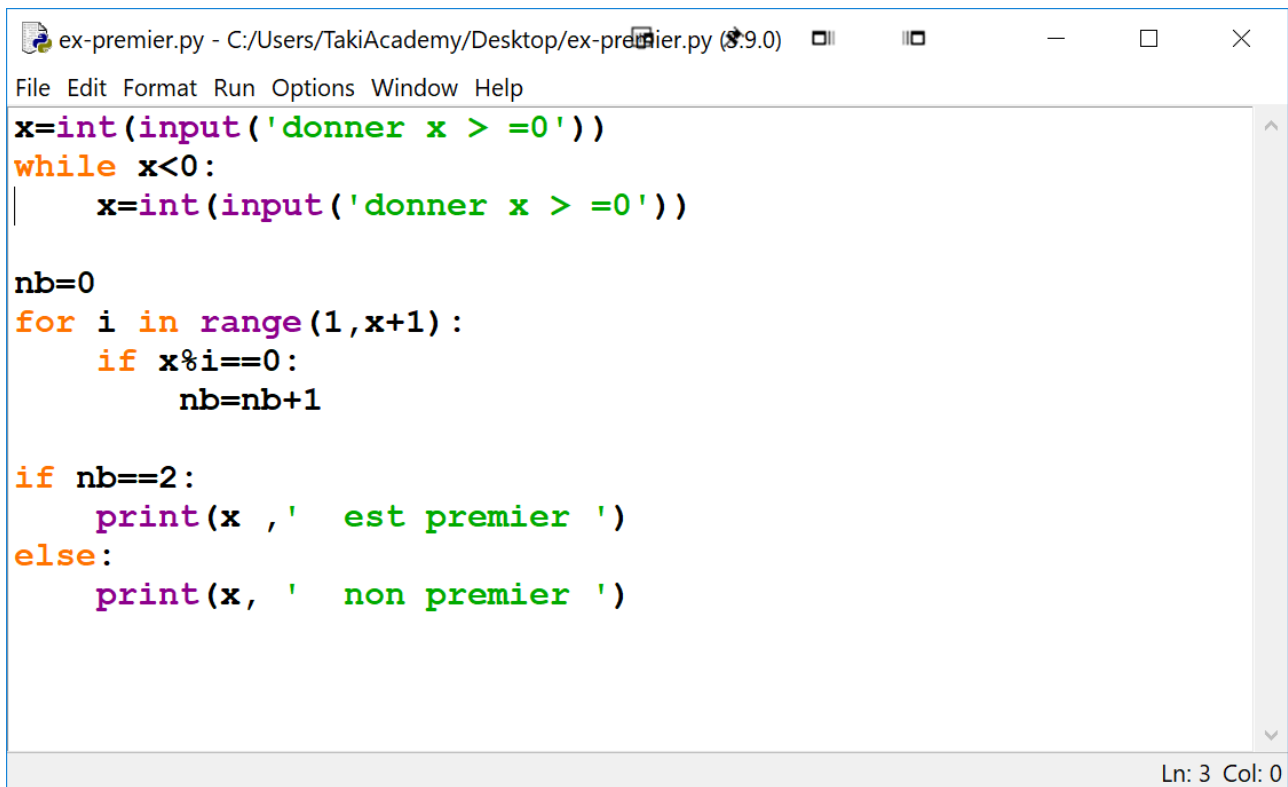
- Saisir un entier positif.
- Vérifier et afficher si l'entier est premier ou non.

**NB :**

Un entier premier n'a que deux diviseurs 1 et lui-même.

**Exemple :**

2, 3, 5, 7, 11, 13, 17, 23,



```
ex-premier.py - C:/Users/TakiAcademy/Desktop/ex-premier.py (3.9.0)
File Edit Format Run Options Window Help
x=int(input('donner x > =0'))
while x<0:
    x=int(input('donner x > =0'))

nb=0
for i in range(1,x+1):
    if x%i==0:
        nb=nb+1

if nb==2:
    print(x, ' est premier ')
else:
    print(x, ' non premier ')

Ln: 3 Col: 0
```

## 6. Les sous programmes

En programmation, il est préférable de décomposer le programme en sous programmes indépendants et de difficultés moindres appelés modules.

### 6.1 - Les fonctions

#### 6.1.1 Définition d'une fonction

En algorithmique une fonction est déclarée en utilisant la syntaxe suivante :

**Fonction** Nom\_fonction (pf<sub>1</sub>: type<sub>1</sub>, pf<sub>2</sub>: type<sub>2</sub>, ... , pf<sub>n</sub>: type<sub>n</sub>) : Type\_résultat

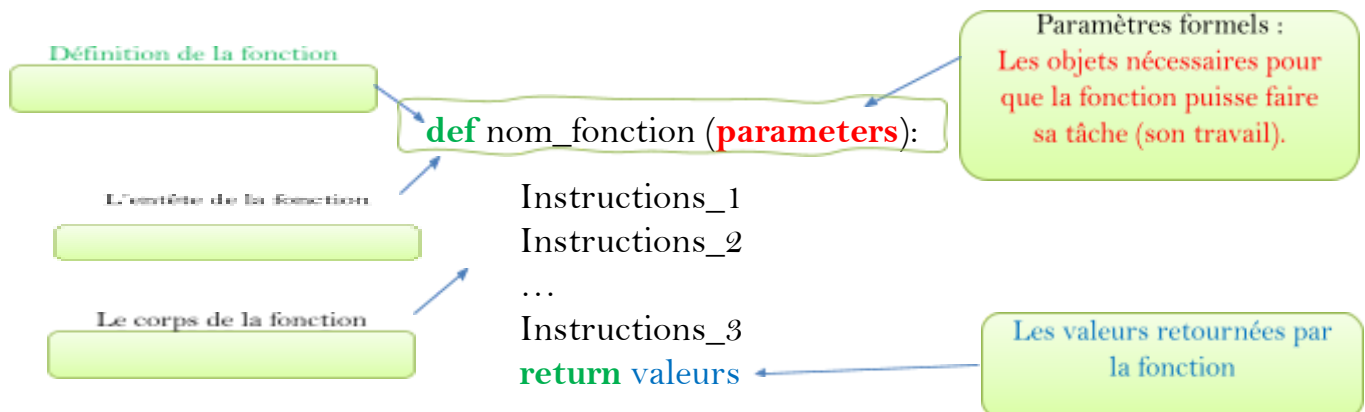
**DEBUT**

Traitement

**Retourner** Résultat

**FIN**

Pour définir une fonction en langage Python, on peut utiliser la syntaxe suivante :



### Remarques :

- 1- Pour exécuter une fonction, il faut **l'appeler**.
- 2- Toutes les instructions qui suivent l'instruction **return** constitue un code mort c'est-à-dire non exécutable par la fonction.
- 3- Une fonction peut ne pas avoir de paramètres au moment de la définition, dans ce cas il ne faut pas oublier les deux parenthèses ().
- 4- Une fonction peut renvoyer une ou plusieurs valeurs de types différents dans la même instruction return.

### **Exemple 1 :** (définition d'une fonction)

Donner le code d'une fonction nommée double qui prend en paramètre un entier x pour renvoyer le double de sa valeur.

```
def double(x):  
    return(x*2)
```

Ne peut confondre la fonction **print** et l'instruction **return**

### **Exemple 2 :** (appel d'une fonction)

```
print(double(2)) #la valeur 4 sera affichée
```

### **Exemple 3 :** (fonction non paramétrée)

```
def test():  
    return(0*5, 'computer') #la fonction retourne deux valeurs 10 et computer
```

### **Exercice 19 :** (calculer la factorielle)

La factorielle d'un entier n noté **n!** est égal à **1\*2\*... n**

- Calculer : **4!** = 1\*2\*3\*4 et **5!** = 1\*2\*3\*4\*5 et **0!** = 1
- Donner l'algorithme d'une fonction nommée **fact** qui prend en paramètre un entier n afin de calculer et renvoyer sa factorielle.
- Traduire l'algorithme de la fonction en Python

**fact : nom de la fonction**

**n : paramètre formel de la fonction**

```
def fact(n):  
    f=1  
    for i in range(1,n+1):  
        f=f*i  
    return f
```

**Solution :**

### **Exercice 20 :** (bêtisier)

Voici un bêtisier qui génère à chaque fois une erreur, donner la cause de l'erreur dans chaque cas.

Une variable définie à l'intérieure d'une fonction ne sera pas visible depuis l'extérieur. On dit que la variable est locale à la fonction.

#### **Exemple 1 :** (variable locale)

```
def f():
    x=2
    return (x*2)

print(f()) #affiche 4
print(x) #genere une erreur : x not found
```

Le `print(x)` génère une erreur car `x` est une variable locale à la fonction et elle n'est pas visible à l'extérieure de la fonction `f`.

Si une variable est déjà définie à l'extérieure d'une fonction, si on modifie sa valeur durant l'exécution de la fonction, sa valeur d'origine ne sera pas modifiée c'est-à-dire, tout se passe comme si la variable était masquée momentanément.

### Exemple 2 : (variable extérieure à la fonction)

Soit le code suivant :

```
x=6
m=2
def fonction(y):
    x = 7
    print(x)
    return y*x

print(x)
print(fonction(2))
print(x)
```

quel est le résultat affiché par le programme précédent ?

```
6
7
14
6
```

## 6.2 - Modules

Un module est une collection de fonctions. Il y a différents types de modules :

- Ceux qui sont inclus dans la version de Python comme **random** ou **math**
- Ceux que l'on peut rajouter comme **numpy** ou **matplotlib**
- Ceux que l'on peut faire soi-même (il s'agit dans les cas simples d'un fichier Python contenant un ensemble de fonctions).

### 6.2.1 Importation des modules

Pour utiliser des fonctions faisant partie d'un module, il faut avant tout les importer.

La syntaxe générale est :

```
import ModuleName
```

Exemple :

```
>>> import time
>>> import random
>>> import math
```

Les fonctions s'utilisent sous la forme :

**ModuleName.FunctionName(parameters)**

**Exemple :**

```
>>> print(math.factorial(5))
120
```

```
>>> print(math.sqrt(16))
4.0
```

Remarquons que la commande qui permet de calculer est précédée du module duquel elle vient.



### ATTENTION

- On peut utiliser un alias pour le nom du module, on écrira alors :

**import ModuleName as Alias**

Les fonctions s'utilisent alors sous la forme :

**Alias.FunctionName(parameters)**

**Exemple :**

```
>>> import math as mt
>>> print(mt.sqrt(25))
5.0
```

- Il est également possible d'importer seulement quelques fonctions d'un module :

**from ModuleName import fonction1, fonction2.**

Dans ce cas les fonctions peuvent être utilisées directement par **FunctionName(parameters)**.

**Exemple :**

```
>>> from math import sqrt, factorial, floor, trunc
>>> print(sqrt(16))
4.0
>>> print(trunc(12.75))
12
>>> print(floor(194.32))
194
```

## 6.2.2 Exemples de modules

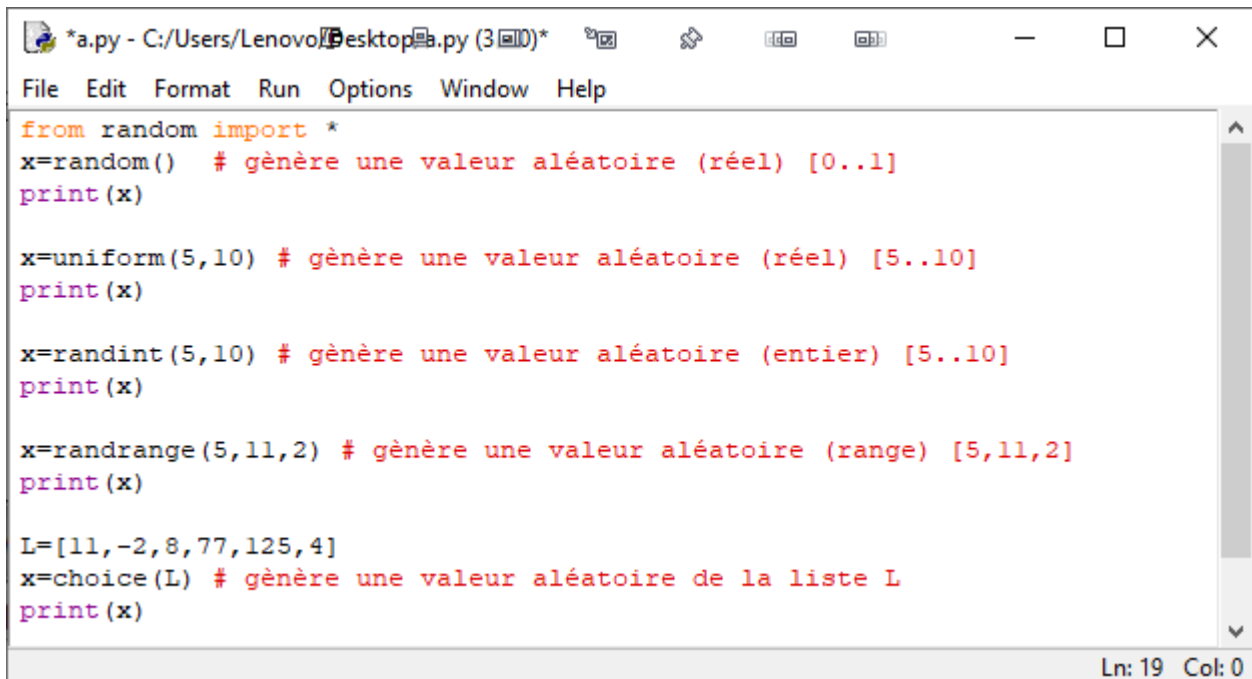
Python offre par défaut une bibliothèque de plus de deux cents modules qui évite d'avoir à réinventer la roue dès que l'on souhaite écrire un programme. Ces modules couvrent des domaines très divers : mathématiques (fonctions mathématiques usuelles, calculs sur les réels, sur les complexes, combinatoire, matrices, manipulation d'images. . .), administration système, programmation réseau, manipulation de fichiers, etc.

### 1- Le module random :

Ce module propose diverses fonctions permettant de générer des nombres aléatoires (au hasard) :

|                                      |                                                                           |
|--------------------------------------|---------------------------------------------------------------------------|
| <code>random.randrange(p,n,h)</code> | choisit un éléments aléatoirement dans la liste <code>range(p,n,h)</code> |
| <code>random.randint(a,b)</code>     | choisit un <i>entier</i> aléatoirement dans l'intervalle $[a;b]$          |
| <code>random.choice(seq)</code>      | choisit un éléments aléatoirement dans la liste <code>seq</code>          |
| <code>random.random()</code>         | renvoie un <i>décimal</i> aléatoire dans $[0;1[$                          |
| <code>random.uniform(a,b)</code>     | choisit un <i>décimal</i> aléatoire dans $[a;b]$                          |

Tester le code suivant :



```
*a.py - C:/Users/Lenovo/Desktop/a.py (3.10.0)*
File Edit Format Run Options Window Help
from random import *
x=random() # génère une valeur aléatoire (réel) [0..1]
print(x)

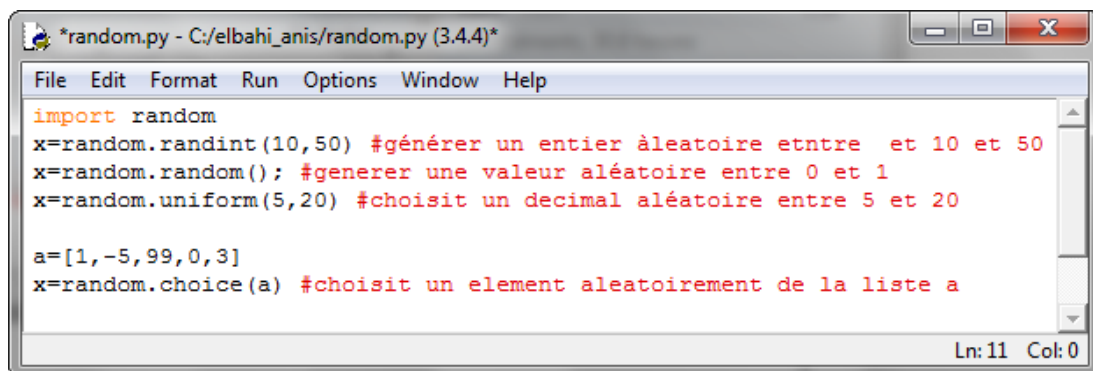
x=uniform(5,10) # génère une valeur aléatoire (réel) [5..10]
print(x)

x=randint(5,10) # génère une valeur aléatoire (entier) [5..10]
print(x)

x=randrange(5,11,2) # génère une valeur aléatoire (range) [5,11,2]
print(x)

L=[11,-2,8,77,125,4]
x=choice(L) # génère une valeur aléatoire de la liste L
print(x)

Ln: 19 Col: 0
```



```
*random.py - C:/elbahi_anis/random.py (3.4.4)*
File Edit Format Run Options Window Help
import random
x=random.randint(10,50) #générer un entier aleatoire etntre et 10 et 50
x=random.random(); #generer une valeur aléatoire entre 0 et 1
x=random.uniform(5,20) #choisit un decimal aléatoire entre 5 et 20

a=[1,-5,99,0,3]
x=random.choice(a) #choisit un element aleatoirement de la liste a

Ln: 11 Col: 0
```

### Exercice 21 : (Manipulation d'un tableau avec les sous programmes)

On desire faire un programme Python qui permet de faire les tâches suivantes :

- **Saisir** un entier  $n$  (avec  $2 < n < 10$ ).
- **Remplir** un tableau  $t$  par  $n$  entiers.
- **Afficher** le tableau  $t$  après le remplissage.
- **Calculer** et afficher la **moyenne** du tableau  $t$ .
- **Chercher** et afficher le **maximum** du tableau  $t$ .

NB: il faut décomposer votre programme en modules

```

#saisie contrôlée de n  2<n<10
def saisir():
    n=int(input("donner n :"))
    while not (n>2 and n<10):
        n=int(input("donner n :"))
    return n

#remplissage du tableau
def remplir(n):
    t=[]
    for i in range(n):
        x=int(input("donner l'element : "+str(i)+" "))
        t.append(x)
    return(t)

#affichage du tableau
def afficher(t,n):
    for i in range(n):
        print(t[i], end = ' ')
    print()

#calcul de la moyenne du tableau
def moyenne(t,n):
    s=0
    for i in range(n):
        s=s+t[i]
    return (s/n)

#cherche du maximum
def maximum(t,n):
    mx=t[0]
    for i in range(1,n):
        if t[i]>mx:
            mx=t[i]
    return mx

#### --- le programme principal --- ####
n=saisir()
t=remplir(n)
afficher(t,n)
m=moyenne(t,n)
print('la moyenne du tableau :', m)
mx=maximum(t,n)
print('le max du tableau :', mx)

```



## 7. Algorithmes d'arithmétiques

L'arithmétique est une branche des mathématiques qui étudie les relations entre les nombres. Du grec « arithmétiké » qui signifie « l'art des nombres », et elle se définit aussi par « la science des nombres ». Le test de primalité, le test de parité, l'étude des nombres parfaits, le calcul de factoriel, le calcul de PGCD, le calcul de PPCM, ... sont des exemples d'études arithmétiques.

### 7.1 Nombres premiers

Un nombre est dit premier s'il n'a que deux diviseurs 1 et lui-même.

#### Exercice 22 : (Nombre premier)

- Donner 5 exemples de nombre premiers : 2, 3, 5, 7, 11, 13
- Donner l'algorithme d'une fonction booléenne qui permet de vérifier si un entier x passé en paramètre est premier ou non.

**Fonction Premier(x :entier) : entier**

**DEBUT**

nb ← 0

Pour i de 1 à x faire

    Si  $x \bmod i = 0$  alors

        nb ← nb + 1

    Fin Si

Fin Pour

Si nb=2 alors

    Premier ← vrai

Sinon

    Premier ← Faux

Fin si

**FIN**

TDOL :

| Objet  | Type / Nature |
|--------|---------------|
| nb , i | entier        |

### 7.2 Calcul du factoriel

La factorielle d'un entier positif N noté N! est calculé comme suit :  $N! = 1 * 2 * 3 * \dots * N$

### Exercice 23 : (factorielle)

1- Donner l'algorithme d'une fonction somme qui calcule la somme SN suivante avec N un entier positif saisi au clavier et passé en paramètre à la fonction somme.

$$SN = 1 + 2 + 3 + \dots + N$$

2- calculer 5 ! puis 8 !

3- Donner l'algorithme d'une fonction fact qui calcule la factorielle d'un entier naturel N

## 7.3 Calculer la puissance de deux entiers positifs

### Exercice 24 : (Puissance $x^y$ )

1- Donner l'algorithme d'une fonction qui prend en paramètre deux entiers x et y positifs pour calculer et renvoyer la valeur de x à la puissance y ( $x^y$ )

2- Traduire l'algorithme de la fonction en Python

### Exercice 25 : (somme avancée)

On désire faire un programme qui permet de faire les tâches suivantes :

- Saisir un entier N avec ( $4 \leq N \leq 15$ ) et un réel  $x > 0$
- Calculer et afficher la somme S suivante :

#### Travail à Faire :

- Faire l'algorithme du programme principal.
- Faire les algorithmes des modules envisagés

## 7.4 Nombres parfait

Un nombre est dit parfait s'il est égal à la somme de ses diviseurs autres que lui-même.

Par exemple 6 est un nombre parfait car les diviseurs de 6 sont 1,2,3 et 6 et puisque  $6 = 1 + 2 + 3$  donc 6 est un nombre parfait.

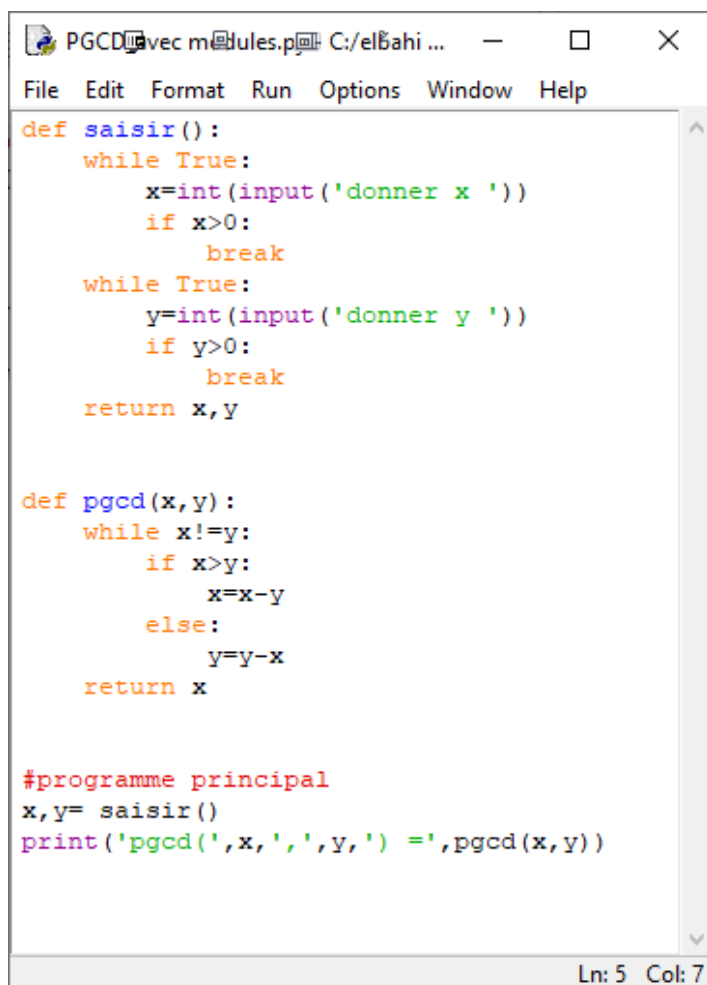
### Exercice 26 : (Nombre parfait)

Donner l'algorithme d'une fonction nommée parfait qui vérifie si un nombre positif x est parfait ou non.

## 7.5 – PGCD : Plus Grand Commun Diviseur

### Exercice 27 : (PGCD)

- Calculer le PGCD (Plus Grand Commun Diviseur) pour les couples suivants en utilisant la méthode de différence qui consiste à soustraire le plus petit du plus grand jusqu'à ce que les deux valeurs soient égaux.
  - ☐ X=15 et Y=27
  - ☐ X=120 et Y=50
- On désire faire un programme qui permet de :
  - Saisir deux entier x et y strictement positifs



```
def saisir():
    while True:
        x=int(input('donner x '))
        if x>0:
            break
    while True:
        y=int(input('donner y '))
        if y>0:
            break
    return x,y

def pgcd(x,y):
    while x!=y:
        if x>y:
            x=x-y
        else:
            y=y-x
    return x

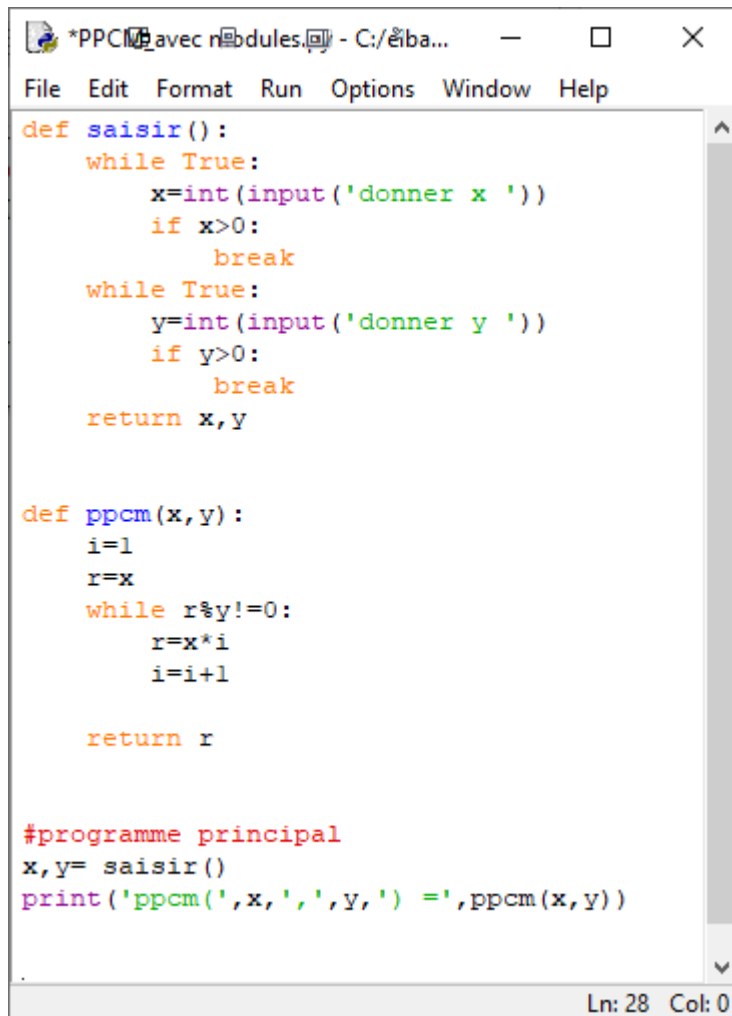
#programme principal
x,y= saisir()
print('pgcd(' ,x, ',' , y, ') =' ,pgcd(x,y))
```

Ln: 5 Col: 7

## 7.6 – PPCM : Plus Petit Commun Multiple

### Exercice 28 : (PPCM)

- Pour les couples suivants calculer les PPCM
$$\text{PPCM}(5,6) = 30$$
$$\text{PPCM}(120,50) = 600$$
$$\text{PPCM}(50,120) = 600$$
- Donner l'algorithme d'une fonction qui calcul le PPCM de deux entiers x et y.



```
def saisir():
    while True:
        x=int(input('donner x '))
        if x>0:
            break
    while True:
        y=int(input('donner y '))
        if y>0:
            break
    return x,y

def ppcm(x,y):
    i=1
    r=x
    while r%y!=0:
        r=x*i
        i=i+1

    return r

#programme principal
x,y= saisir()
print('ppcm(',x,',',y,') =',ppcm(x,y))
```

Ln: 28 Col: 0

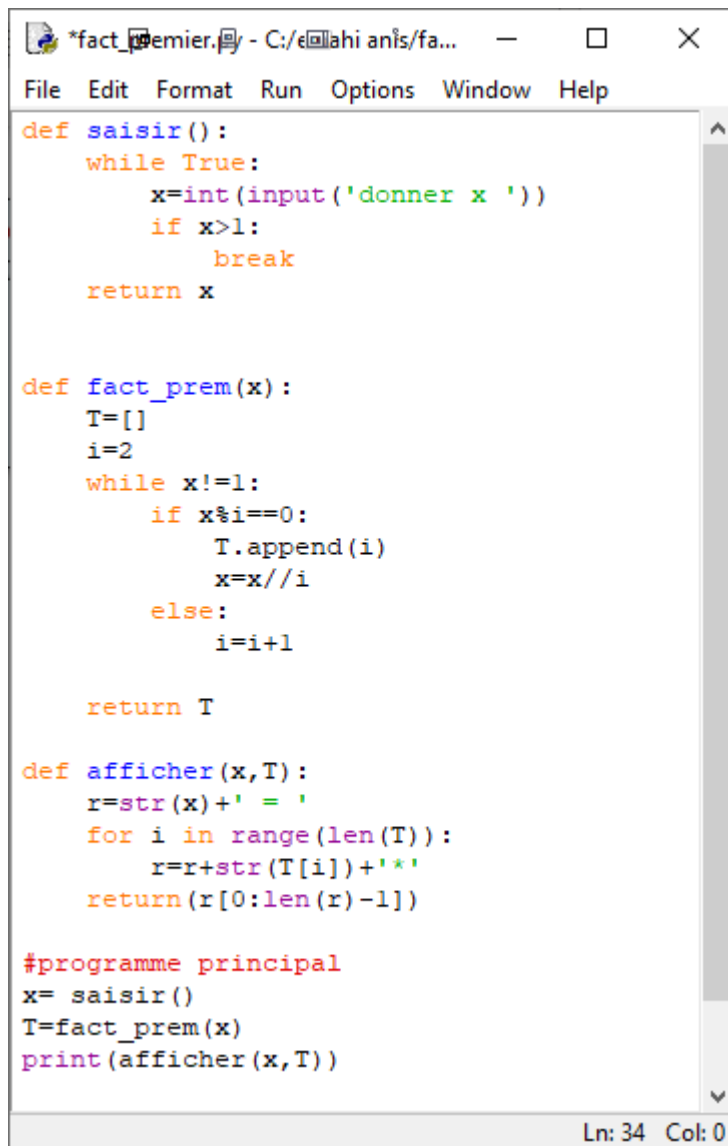
## 7.7 – Décomposition en facteurs premiers

### Exercice 29 : (décomposition en facteurs premiers)

- Décomposer 600 sous forme de produit de facteurs premiers :  $600 = 2*2*2*3*5*5$
- Donner le programme python qui permet de :
  - Saisir un entier  $x > 1$
  - Afficher la décomposition en facteurs premiers de l'entier  $x$  comme le montre l'exemple :

Exemple :

Pour  $x=630$  le programme doit afficher :  $630 = 2*3*3*5*7$



```
def saisir():
    while True:
        x=int(input('donner x '))
        if x>1:
            break
    return x

def fact_prem(x):
    T=[]
    i=2
    while x!=1:
        if x%i==0:
            T.append(i)
            x=x//i
        else:
            i=i+1
    return T

def afficher(x,T):
    r=str(x)+' = '
    for i in range(len(T)):
        r=r+str(T[i])+'*'
    return(r[0:len(r)-1])

#programme principal
x= saisir()
T=fact_prem(x)
print(afficher(x,T))
```

Ln: 34 Col: 0

## 8. Algorithmes de tri

Un algorithme de tri est une suite d'instructions servant à ordonner (ranger) une séquence d'éléments de façon croissante ou décroissante.

### Exemples :

- Séquence triée : 12 – 15 – 88 – 121 – 122 – 714 – 901 – 2510
- Séquence non triée : 15.5 – 19 – 10 – 201 – 3 – 155

### 8.1 Tri à Bulles :

Ordre croissant

#### Principe :

Cette méthode consiste à faire remonter le plus grand élément du tableau en comparant les éléments successifs.

- 1- On compare le premier **pair d'éléments**.
- 2- Si  $T[1] > T[2]$  alors on permute  $T[1]$  et  $T[2]$ , aller au pair suivant et répéter les étapes 1 et 2 jusqu'à comparer le dernier pair  $T[N]$  et  $T[N-1]$ . A la fin de ce premier parcours on aura passé le plus grand élément du tableau vers sa place finale qui est le Nième élément du tableau.
- 3- On recommence cette opération en parcourant de 1 à N-1 puis de 1 à N-2 et ainsi de suite.
- 4- On arrête le traitement si on arrive au dernier élément du tableau ou le tableau devient trié.

#### Algorithme de tri à bulles :

PROCEDURE Tri\_bulles (N : entier , @ T :TAB)

DEBUT

Répéter

  échange  $\square$  Faux

    Pour i de 0 à N-2 faire

      Si  $T[i] > T[i+1]$  alors

        aux  $\square$   $T[i]$

$T[i] \square T[i+1]$

$T[i+1] \square$  aux

        échange  $\square$  Vari

      Finsi

    Fin pour

    N  $\square$  N-1

  Jusqu'à (N =0) ou (échange = Faux)

FIN

TDOL :

| Objet   | Type / Nature |
|---------|---------------|
| aux, i  | Entier        |
| échange | Booléen       |



### Exercice 30 : (tri à bulles)

On désire faire un programme nommé **classement** qui permet de remplir un tableau T par N entiers quelconques (avec  $3 < N < 10$ ) puis de trier ses éléments par ordre croissant en utilisant le tri à bulles et d'afficher le tableau T avant et après le tri.

#### Travail demandé :

- Donner l'algorithme du programme principal
- Donner les algorithmes des modules envisagés.

```

def tri_bulles(t,N):
    while (N!=0) :
        echange=False
        for i in range(N-1):
            if t[i]>t[i+1]:
                aux=t[i]
                t[i]=t[i+1]
                t[i+1]=aux
                echange=True
        N=N-1
        if echange==False :
            break
    return t

def saisir():
    while True:
        N=int(input('donner N'))
        if (3<N<10):
            break
    return N

def remplir(N):
    t=[]
    for i in range(N):
        x=int(input('T[' + str(i) + ']'))
        t.append(x)
    return t

def afficher(t,N):
    for i in range(N):
        print(t[i], end=' ')
    print()

#programme principal
N=saisir()
t=remplir(N)
print('avant le tri')
afficher(t,N)
t=tri_bulles(t,N)
print('apres le tri')
afficher(t,N)

```

## 8.2 Tri par sélection :

Ordre croissant

### Principe :

Cette méthode consiste à :

- 1- Trouver l'indice « position » (posmin) du plus petit élément du tableau.
- 2- **Placer le plus petit élément ( $T[posmin]$ ) à sa position finale** (la première position)
- 3- Rechercher l'indice du second plus petit élément
- 4- Le placer à sa position finale (deuxième position)
- 5- Répéter le traitement précédent (3 et 4) jusqu'à ce que le tableau soit trié.

### Algorithme de tri par sélection :

PROCEDURE Tri\_select (N : entier , @ T : TAB)

DEBUT

Pour i de 0 à N-2 faire

pm  $\leftarrow$  FN Pmin (T,N,i)

si (pm  $\neq$  i) alors

aux  $\leftarrow T[i]$

$T[i] \leftarrow T[pm]$

$T[pm] \leftarrow$  aux

Fin si

Fin Pour

FIN

pm : est la position du minimum

Fonction Pmin (T : TAB , N, d : entier) : entier

DEBUT

pm  $\leftarrow$  d

Pour j de d+1 à N-2 faire

Si  $T[j] < T[pm]$  alors

pm  $\leftarrow$  j

Finsi

Fin pour

Retourner(pm)

FIN

#### Rôle de la fonction Pmin :

la fonction cherche la position du plus petit élément dans la partie du tableau de la case d à la case N

### Exercice 31 : (tri par sélection)

On désire faire un programme qui permet de remplir un tableau T par N entiers quelconques (avec  $3 < N < 10$ ) puis de trier ses éléments par ordre croissant en utilisant le tri par selection et d'afficher le tableau T avant et après le tri.

#### Travail demandé :

- Donner le program Python qui permet de résoudre ce problème.



## 9. Les enregistrements :

### 1- Présentation du problème

Les types prédéfinis que nous avons vus (entier, réel, chaîne de caractères, booléen, ...) sont insuffisants pour traiter des données plus complexes. Par exemple si on veut représenter un élève qui est caractérisé son code, son nom, son prénom, son genre et sa moyenne générale, etc. On voudrait qu'une seule variable conserve et donc donne accès à toutes ces informations. En algorithmique, on définirait alors un **type enregistrement** **Eleve** regroupant ces informations.

### 2- Definition d'un enregistrement

*Un enregistrement* { Est un type de données défini par l'utilisateur qui permet le regroupement d'un ensemble de champs décrivant un objet de monde réel. }

Exemple d'enregistrement :

| ELEVE |                  |           |
|-------|------------------|-----------|
| Champ | Signification    | Type      |
| C     | Code             | Entier    |
| N     | Nom              | Chaîne    |
| P     | Prénom           | Chaîne    |
| G     | Genre            | Caractère |
| MG    | Moyenne Générale | Réel      |

| Champs | Valeurs |
|--------|---------|
| C      | 1023    |
| N      | ELBAHI  |
| P      | Anas    |
| G      | H       |
| MG     | 18.75   |

### Exercice 32 : (exemples d'enregistrements)

Dans le tableau suivant, donner trois autres exemples d'objets de monde réel ainsi que certaines de leurs caractéristiques comme le montre l'exemple suivant :

*Exemple :*

### 3- Déclaration d'un enregistrement:

#### Algorithmique :

#### Déclaration algorithmique d'un type enregistrement

| En algorithmique                                                                                                                                                                                                                                                   | Exemple |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|----------------------------------|----------------|----------------|-----|----------------|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|-------------------------------|------------|---------------|---------------|-----------|------------------|
| Tableau de Déclaration des Nouveaux Types                                                                                                                                                                                                                          | TDNT    |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| <table><tr><th>TYPES</th></tr><tr><td>Nom_type = <b>Enregistrement</b></td></tr><tr><td>    Champ1 : type1</td></tr><tr><td>    Champ2 : type2</td></tr><tr><td>    ...</td></tr><tr><td>    Champn : typen</td></tr><tr><td><b>Fin</b> Nom_type</td></tr></table> | TYPES   | Nom_type = <b>Enregistrement</b> | Champ1 : type1 | Champ2 : type2 | ... | Champn : typen | <b>Fin</b> Nom_type | <table><tr><th>TYPES</th></tr><tr><td>eleve = <b>Enregistrement</b></td></tr><tr><td>    C : entier</td></tr><tr><td>    N, P : chaine</td></tr><tr><td>    G : caractère</td></tr><tr><td>    MG : réel</td></tr><tr><td><b>Fin</b> eleve</td></tr></table> | TYPES | eleve = <b>Enregistrement</b> | C : entier | N, P : chaine | G : caractère | MG : réel | <b>Fin</b> eleve |
| TYPES                                                                                                                                                                                                                                                              |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| Nom_type = <b>Enregistrement</b>                                                                                                                                                                                                                                   |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| Champ1 : type1                                                                                                                                                                                                                                                     |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| Champ2 : type2                                                                                                                                                                                                                                                     |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| ...                                                                                                                                                                                                                                                                |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| Champn : typen                                                                                                                                                                                                                                                     |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| <b>Fin</b> Nom_type                                                                                                                                                                                                                                                |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| TYPES                                                                                                                                                                                                                                                              |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| eleve = <b>Enregistrement</b>                                                                                                                                                                                                                                      |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| C : entier                                                                                                                                                                                                                                                         |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| N, P : chaine                                                                                                                                                                                                                                                      |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| G : caractère                                                                                                                                                                                                                                                      |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| MG : réel                                                                                                                                                                                                                                                          |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |
| <b>Fin</b> eleve                                                                                                                                                                                                                                                   |         |                                  |                |                |     |                |                     |                                                                                                                                                                                                                                                              |       |                               |            |               |               |           |                  |

#### Déclaration algorithmique d'une variable de type enregistrement

| En algorithmique                                                                                                                                                                                | Exemple       |               |           |          |                                                                                                                                                        |       |               |   |       |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|---------------|-----------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------|-------|---------------|---|-------|
| Tableau de Déclaration des Objets<br><table border="1"> <thead> <tr> <th>Objet</th><th>Type / nature</th></tr> </thead> <tbody> <tr> <td>nom_objet</td><td>Nom_type</td></tr> </tbody> </table> | Objet         | Type / nature | nom_objet | Nom_type | TDO<br><table border="1"> <thead> <tr> <th>Objet</th><th>Type / nature</th></tr> </thead> <tbody> <tr> <td>E</td><td>eleve</td></tr> </tbody> </table> | Objet | Type / nature | E | eleve |
| Objet                                                                                                                                                                                           | Type / nature |               |           |          |                                                                                                                                                        |       |               |   |       |
| nom_objet                                                                                                                                                                                       | Nom_type      |               |           |          |                                                                                                                                                        |       |               |   |       |
| Objet                                                                                                                                                                                           | Type / nature |               |           |          |                                                                                                                                                        |       |               |   |       |
| E                                                                                                                                                                                               | eleve         |               |           |          |                                                                                                                                                        |       |               |   |       |

#### Python :

Le type enregistrement n'existe pas vraiment en Python, mais plusieurs solutions sont possibles : les tuples, les dictionnaires ou les classes.

#### **Exercice 33 : (Enregistrement en Python)**

Soit l'enregistrement eleve suivant :

TDNT

A chaque fois, déclarer l'enregistrement **eleve** en Python sous forme de tuple, class, dictionnaire et remplir ses champs par les valeurs suivantes :

### Solution 1 : (Enregistrement sous forme d'un tuple)

Un **tuple** est un type qui regroupe un ensemble d'éléments de types éventuellement différents. Chaque élément est accessible grâce à son nom.

```
#enregistrement sous forme de tuple
#definition d'un enregistremenet
#et remplissage de ses champs
eleve=(789,'ELBAHI','Anis','H', 19.30)
c,n,p,g,mg = eleve

#afficher des champs
print('le code = ', c)
print('le nom et prenom = ', n , ' ' ,p)
print('la moyenne = ', mg)

#-> un tuple est immutable cad on ne peut pas modifier ces champs
#-> chaque modification revient a creer un nouveau tuple

mg=mg+1 #pour ajouter 1 a la moyenne generale
eleve=(c,n,p,g,mg)

#pour afficher tous les champs
print(eleve)
|
```

### Solution 2 : (Enregistrement sous forme d'un dictionnaire)

Un **dictionnaire** est une structure de données dite associative, car elle permet de stocker une valeur en lui associant une clé.

```

#enregistrement sous forme de dictionary
#definition d'un enregistrement
#et remplissage de ses champs
eleve = dict(c=789 , n='ELBAHI' , p='Anis' , g='H' , mg=19.30)

eleve["c"] =123 #modifier la valeur d'un champ
eleve['mg']=18.99

#affihcer quelques champs
print('le code = ', eleve["c"])
print('le nom et prenom = ', eleve["n"], ' ', eleve["p"])
print('la moyenne = ', eleve["mg"])

#afficher tous les champs
print(eleve)

```

### Solution 3 : (Enregistrement sous forme d'une classe)

```

#enregistrement sous forme de class
#pour definir un enregistrement :
class eleve:
    |   c=0
        n=""
        p=""
        g=""
        mg=0

#pour creer une variable nommee e de type eleve
e=eleve()

#remplissage des champs de la variable e
e.c=789
e.n='ELBAHI'
e.p='Anis'
e.g='H'
e.mg=19.30

#changer la valeur d'un champ
e.c=779

#affihcer les valeurs des champs
print(e.c, ' ', e.p, ' ', e.n, ' ', e.mg)

```

### Remarque :

- 1- Si on veut définir une classe sans champs, on doit écrire
 

```

class eleve :
    pass

```

2- Dans ce cours nous privilégierons les classes pour définir les enregistrements.

### Exercice 34 : (Enregistrement en Python)

Soit l'enregistrement **Patient** suivant formé de 5 champs, on vous demande de :

- Déclarer en algorithmique l'enregistrement **Patient**
- Déclarer en algorithmique deux variables **P1** et **P2** de type **patient**.
- Remplir les champs des patients P1 et P2 par des valeurs de votre choix
- Afficher pour chaque patient son Code, Nom, Prénom et son IMC (Indice de masse corporelle) sachant que :  

$$IMC = M/T^2$$
 (P= poids en Kilogrammes et T=taille en metres)
- Traduire les question 1,2,3 et 4 en python

Solution :

1-

Déclaration algorithmique d'un type enregistrement

| Exemple                                                                                                                                                                                                                                                                                                                                          |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|-------|--|--------------------------|--|------------|--|--------------|--|------------|--|---------------|--|-------------|--|
| TDNT                                                                                                                                                                                                                                                                                                                                             |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| <table border="1"> <tr> <th colspan="2">TYPES</th></tr> <tr> <td colspan="2">patient = Enregistrement</td></tr> <tr> <td colspan="2">C : entier</td></tr> <tr> <td colspan="2">P,N : chaîne</td></tr> <tr> <td colspan="2">T,M : réel</td></tr> <tr> <td colspan="2">G : caractère</td></tr> <tr> <td colspan="2">Fin patient</td></tr> </table> |  | TYPES |  | patient = Enregistrement |  | C : entier |  | P,N : chaîne |  | T,M : réel |  | G : caractère |  | Fin patient |  |
| TYPES                                                                                                                                                                                                                                                                                                                                            |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| patient = Enregistrement                                                                                                                                                                                                                                                                                                                         |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| C : entier                                                                                                                                                                                                                                                                                                                                       |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| P,N : chaîne                                                                                                                                                                                                                                                                                                                                     |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| T,M : réel                                                                                                                                                                                                                                                                                                                                       |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| G : caractère                                                                                                                                                                                                                                                                                                                                    |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |
| Fin patient                                                                                                                                                                                                                                                                                                                                      |  |       |  |                          |  |            |  |              |  |            |  |               |  |             |  |

2 -

Déclaration algorithmique d'une variable de type enregistrement

| Exemple |               |
|---------|---------------|
| TDO     |               |
| Objet   | Type / nature |
| P1, P2  | patient       |

4-

Ecrire ("patient 1 :")

Ecrire (P1.C)

Ecrire (P1.N)

Ecrire (P1.P)

Ecrire (P1.M/(P1.T\*P1.T))

Ecrire ("patient 2 :", P2.C, " ", P2.N, " ", P2.P, " ", P2.M/(P2.T\*P2.T))

---

```

#declaration de l'enregistrement Patient
class patient:
    C,P,N,T,M,G= 0, '', '', 0,0, ''

#declaration des variables de type patient
P1=patient()
P2=patient()

#remplissage des champs du patient 1
P1.C=123
P1.P='Khaled'
P1.N='BEN AHMAD'
P1.T=1.71
P1.M=83.6
P1.G='H'

#remplissage des champs du patient 2
P2.C=456
P2.P='samia'
P2.N='benzarti'
P2.T=1.65
P2.M=71
P2.G='F'

#affichage des champs du patient 1
print('Patient 1 ')
print(P1.C)
print(P1.N)
print(P1.P)|
print(P1.M / (P1.T*P1.T))

#affichage des champs du patient 2
print('Patient 2 ')
print(P2.C, ' ', P2.N, ' ', P2.P, ' ', P2.M / (P2.T*P2.T))

```

## 10. Les vecteurs d'enregistrements :

### 1- Présentation du problème

Pour stocker les informations d'un élève, on a défini un nouveau objet de type enregistrement nommé `eleve` formé par un ensemble de champs de types différents, chaque champ contient une donnée.

Maintenant si on désire stocker les informations de plusieurs élèves que faut-il faire ?

Réponse :

La meilleure solution est de créer une structure pouvant regrouper plusieurs objets de type enregistrements `eleve`. Cette structure s'appelle vecteur d'enregistrements.

### 2- Definition d'un vecteur d'enregistrements

*Un vecteur d'enregistrements* { Est un ensemble fini d'éléments chacun d'eux est de type enregistrement }

Exemple :

Soit l'enregistrement **ELEVE** suivant formé de 4 champs :

|               |  |
|---------------|--|
| Code          |  |
| Nom et prénom |  |
| Genre         |  |
| Moyenne       |  |

Le tableau `T` suivant est un vecteur formé par 4 enregistrements `ELEVE`.

*C'est le code de l'élève qui se trouve dans la case d'indice 2 du tableau `T` : `T[2].code`*

T

|               |  |
|---------------|--|
| Code          |  |
| Nom et prénom |  |
| Genre         |  |
| Moyenne       |  |

0

|               |  |
|---------------|--|
| Code          |  |
| Nom et prénom |  |
| Genre         |  |
| Moyenne       |  |

1

|               |  |
|---------------|--|
| Code          |  |
| Nom et prénom |  |
| Genre         |  |
| Moyenne       |  |

2

|               |  |
|---------------|--|
| Code          |  |
| Nom et prénom |  |
| Genre         |  |
| Moyenne       |  |

3

### Exercice 35 : (Vecteurs d'enregistrements)

Soit l'enregistrement `eleve` suivant :

On désire faire un programme qui permet de remplir un tableau `T` par `N` élèves (avec  $3 \leq N \leq 39$ ) puis de calculer et d'afficher le nombre des admis, des redoublants ainsi que la moyenne de la classe.

Travail à faire :

- Faire l'algorithme du programme principal.
- Faire les algorithmes des modules envisagés.



