

# Le langage informatique Python 3

Nom :

Prénom :

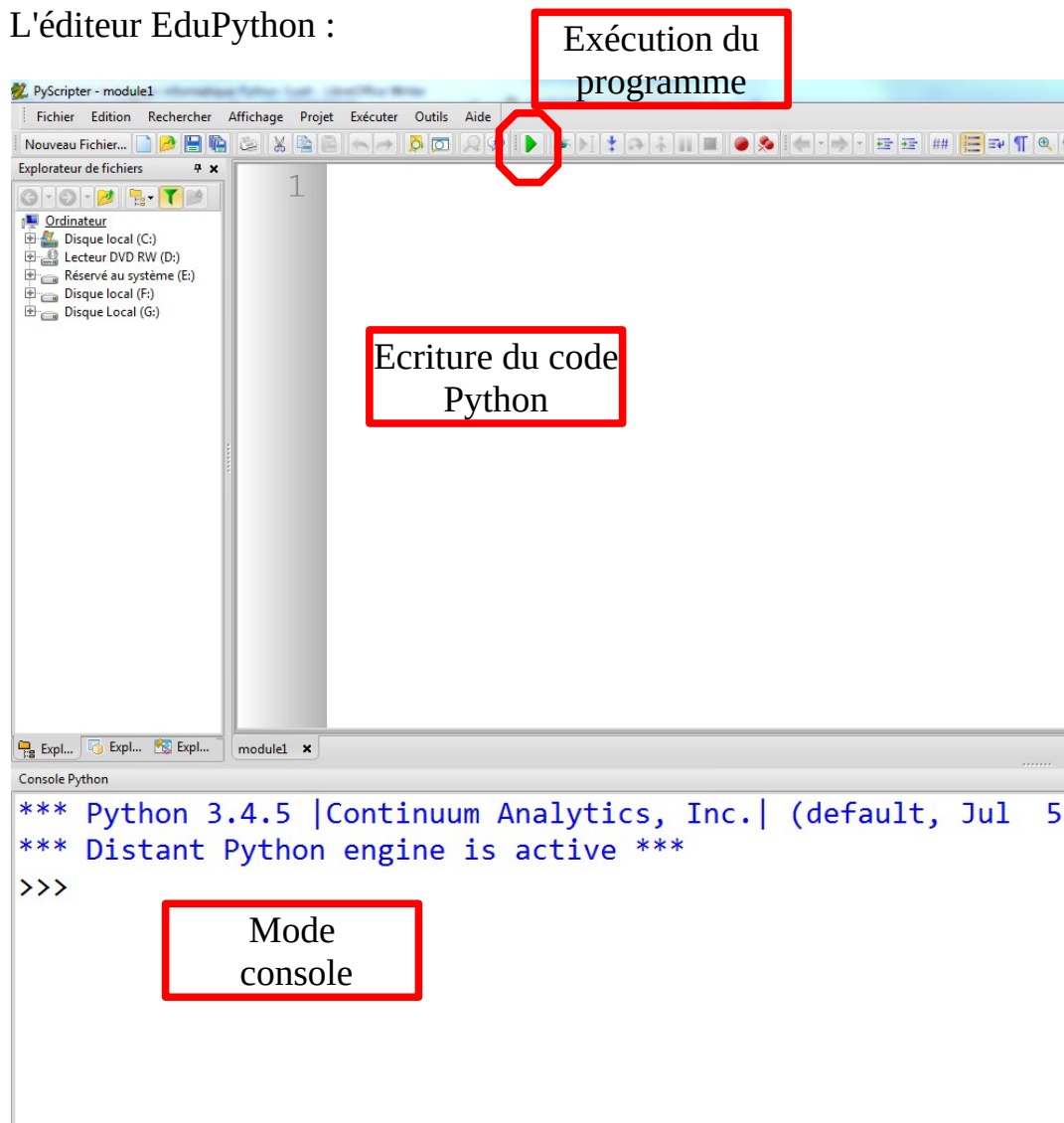
Classe :

Créer dans votre espace de travail NSI un répertoire Python. Ouvrir un nouveau document LibreOfficeWriter de nom NomPrenomPython3.odt et enregistrer le dans ce nouveau répertoire.

Pour apprendre ce langage informatique qui nous permet de donner des instructions à un ordinateur, nous allons suivre dans un premier temps quelques chapitres du livre "Apprendre à programmer avec Python 3" de Gérard Swinnen. Vous trouverez ce livre au format pdf dans votre espace de travail ( Répertoire Python ).

Pour programmer et exécuter les programmes en Python, il nous faut un éditeur Python. Notre choix s'est porté sur EduPython ( <https://edupython.tuxfamily.org/> ). Edupython est disponible dans le menu "Windows".

L'éditeur EduPython :



## **A ) Premiers pas avec Python :**

### **A-1 ) Calculer avec Python :**

Lire les pages 31 et 32 ( /473 ) et exécuter en mode console les 8 calculs demandés et faites un copier / coller des calculs et résultats. Ecrire un résumé des informations lues.

### **A-2 ) Données et variables :**

Lire la partie correspondante de la page 33 /473. Ecrire un résumé des informations lues.

### **A-3 ) Noms de variables et mots réservés :**

Lire la partie correspondante de la page 34 /473. Ecrire un résumé des informations lues.

### **A-4 ) Affectation ou assignation :**

Lire la partie correspondante des pages 34 et 35 ( /473 ). Exécuter en mode console les 3 affectations demandées faites un copier / coller des opérations effectuées. Ecrire un résumé des informations lues.

### **A-5 ) Afficher la valeur d'une variable :**

Lire la partie correspondante des pages 35 et 36 ( /473 ). Exécuter les opérations demandées en mode console. Faites un copier / coller des opérations effectuées. Ecrire un résumé des informations lues.

### **A-6 ) Typage des variables :**

Lire la partie correspondante des pages 36 et 37 ( /473 ).  
En mode console, exécuter les instructions suivantes et copier / coller les résultats obtenus.

```
>>> type(n)
>>> type(msg)
>>> type(pi)
```

Ecrire un résumé des informations lues.

#### A-7 ) Affectations multiples :

Lire la partie correspondante de la page 37 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues. Faire les exercices 2.1 et 2.2. Copier / coller les résultats obtenus.

#### A-8 ) Opérateurs et expressions :

Lire la partie correspondante des pages 37 et 38 ( /473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues. Faire l' exercice 2.3. Copier / coller les résultats obtenus.

#### A-9 ) Priorité des opérations :

Lire la partie correspondante des pages 38 et 39 ( /473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

#### A-10 ) Composition :

Lire la partie correspondante de la page 39 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

### **B ) Contrôle du flux d'exécution :**

#### B-1 ) Séquence d'instructions :

Lire la partie correspondante de la page 41 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

#### B-2 ) Sélection ou exécution conditionnelle :

Lire la partie correspondante des pages 42 et 43 ( /473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

### B-3 ) Opérateurs de comparaison :

Lire la partie correspondante de la page 43 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

### B-4 ) Instructions composées – blocs d'instructions :

Lire la partie correspondante de la page 44 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

### B-5 ) Instructions imbriquées :

Lire la partie correspondante de la page 44 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

### B-6 ) Quelques règles de syntaxe Python :

a ) Les limites des instructions et des blocs sont définies par la mise en page :

Lire la partie correspondante de la page 45 ( / 473 ). Ecrire un résumé des informations lues.

b ) Instruction composée : en-tête, double point, bloc d'instructions indenté :

Lire la partie correspondante des pages 45 et 46 ( /473 ). Ecrire un résumé des informations lues.

c ) Les espaces et les commentaires sont normalement ignorés :

À part ceux qui servent à l'indentation, en début de ligne, les espaces placés à l'intérieur des instructions et des expressions sont presque toujours ignorés (sauf s'ils font partie d'une chaîne de caractères). Il en va de même pour les commentaires : ceux-ci commencent toujours par un caractère dièse ( # ) et s'étendent jusqu'à la fin de la ligne courante.

Dans le programme suivant, les commentaires après le symbole **#** ne sont pas lues.

```
a= 3          # Programme de Test
if a == 3 :   # Variable a
    print("NSI") # Condition de test
else ;
    print("Informatique")
```

**#** permet d'écrire un commentaire sur une seule ligne. Pour écrire un commentaire sur plusieurs lignes, il faut l'encadrer par **"""** et **"""**.

Exemple d'utilisation d'un commentaire sur plusieurs lignes avec **"""** :

```
""" Programme de Test
sur la variable a """
a= 3
""" == correspond en informatique
à l'égalité mathématique = """
if a == 3 :
    print("NSI")
else ;
    print("Informatique")
```

## **C ) Instructions répétitives :**

### **C-1 ) Réaffectation :**


Lire la partie correspondante de la page 47 ( / 473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Faire l'exercice 4.1. Ecrire un résumé des informations lues.

### **C-2 ) Répétitions en boucle – L'instruction while :**

Lire la partie correspondante des pages 48 à 51 ( /473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Faire les exercices 4.2, 4.3 et 4.4. Ecrire un résumé des informations lues.

## **Comment conserver les programmes avec EduPython ?**

Les scripts sont pour l'instant écrits en mode console et disparaissent une fois la session expirée.

Pour enregistrer un programme, sélectionner dans le menu " Fichier ", " Nouveau " puis " Nouveau Module Python ". Donner ensuite un nom à votre programme en sélectionnant dans le menu " Fichier ", le menu " Sauvegarder Sous ... ".  
Pour exécuter le programme Python, utiliser le bouton de commande .

Lire la partie correspondante de la page 53 ( /473 ). Effectuer les opérations demandées et copier / coller les résultats obtenus. Ecrire un résumé des informations lues.

Faire et enregistrer les exercices 4.5 à 4.9.

## **D ) Les principaux types de données :**

### **D-1 ) Les données numériques :**

Le langage informatique Python détecte automatiquement le type de données. La fonction `type()` le précise.

#### Exemples :

```
>>> a1 = 3
>>> type(a1)
>>> <class 'int'>
```

La variable `a1` contient la valeur numérique 3 qui est de type 'int' ( integer qui signifie entier ).

```
>>> a2 = 3,1415
>>> type(a2)
>>> <class 'float'>
```

La variable `a2` contient la valeur numérique 3.1415 ( et non pas 3,1415 ) qui est de type 'float' ( floating qui signifie nombre flottant ou nombre réel ).

Python utilise donc 2 types de données numériques. Les nombres entiers de type 'int' et les nombres réels de type 'float'. Les opérations réalisables sont les opérations d'addition, de soustraction, de division ( / ) et de multiplication ( \* ). Les parenthèses sont utilisées de la même façon qu'en mathématiques.

## D-2 ) Les données alphanumériques ou chaînes de caractères :

### Exemples :

```
>>> n = "a"                # déclaration avec un guillemet "  
>>> type(n)  
>>> <class 'str'>  
>>> lettre='b'            # déclaration avec un guillemet '  
>>> type(lettre)  
>>> <class 'str'>
```

La variable n contient le caractère "a", la variable lettre contient le caractère 'b'. Ces deux variables sont de type 'str' ( string qui signifie chaîne de caractères ). Deux variables de type 'str' peuvent être **additionnées**.

### Exemple :

```
>>> n1 = "Bonjour"  
>>> n2 = " à tous !"  
>>> n1 + n2  
>>> "Bonjour à tous !"
```

Chaque caractère de la chaîne peut être appelé en fonction de sa position encadrée par [ et ]. La première position est 0, la suivante 1, 2, 3, ...

### Exemple :

```
>>> Mot = "Informatique"  
>>> Mot[0]  
>>> 'I'  
>>> Mot[1]  
>>> 'n'  
>>> Mot[2]  
>>> 'f'  
>>> Mot[11]  
>>> 'e'
```

Le nombre de caractères dans la chaîne est donnée par la fonction **len()**, abréviation de length ( longueur en anglais ).

### Exemples :

```
>>> Mot = "Informatique"
>>> len(Mot)
>>> 12
>>> Phrase = " Bonjour à tous !"
>>> len(Phrase)
>>> 17                                     # Les espaces sont aussi comptés
```

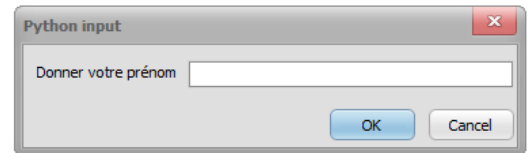
### D-3) Applications

Sauvegarder chacune d'elles dans un module python différent que vous enregistrerez dans votre espace de travail.

1) La fonction `input("Texte")` permet d'affecter une valeur de type chaîne de caractère à une variable par l'intermédiaire d'une boîte de dialogue.

Exemple :

```
>>> Var1 = input("Donner votre prénom ")
```



Indiquer votre prénom dans la boîte de dialogue, puis demander le contenu de Var1.

```
>>> Var1
>>> Sacha
>>> type(Var1)
>>> <class 'str'>
```

Enregistrer et exécuter le programme suivant :

```
Var1 = input("Donner votre prénom ")
print("Bonjour ",Var1," !") # La virgule sert de séparateur entre le texte et la variable.
```

2) Ecrire un programme donnant le nombre de lettres dans le mot. Ce mot doit être saisi dans une boîte de dialogue.

Le programme doit afficher par exemple : Le mot "numérique" contient 9 lettres.

3) Enregistrer et exécuter le programme suivant :

```
Var1 = input("Donner votre prénom ")
for i in Var1 :
    print(i)
```



Que fait ce programme ?

Quelles sont les valeurs prises par la variable i ?

Copier / coller un exemple.

Compléter le en changeant print(i) par print(i,end=" / ").

Qu'apporte cette modification ?

4) Enregistrer et exécuter le programme suivant :

```
Mot = input("Ecrire un mot ")
```

```
Lettre = input("Donner une lettre")
```

```
Compteur = 0
```

```
for j in Mot:
```

```
    if j == Lettre:
```

```
        Compteur = Compteur +1
```

```
print("Le mot ", Mot, "contient ", Compteur," lettres ", Lettre,".")
```

Que fait ce programme ?

Quelles sont les variables ? A quoi sert la variable j ? A quoi sert la variable

Compteur ? Quelle est la différence entre = et == ?

Copier / coller un exemple.

5) Enregistrer et exécuter le programme suivant :

```
Nombre1=input("Premier nombre ? ")
```

```
print(Nombre1)
```

```
Nombre2=input("Deuxième nombre ? ")
```

```
print(Nombre2)
```

```
print(Nombre1+Nombre2)
```

Copier/coller votre résultat. Est-ce que cette addition donne le bon résultat ?

Reprenez le programme en ajoutant **int** devant input.

```
Nombre1=int(input("Premier nombre ? "))
```

```
print(Nombre1)
```

```
Nombre2=int(input("Deuxième nombre ? "))
```

```
print(Nombre2)
```

```
print(Nombre1+Nombre2)
```

Copier/coller votre résultat. Est-ce que cette addition donne maintenant le bon résultat pour des nombres entiers ?

Est-ce que cette addition donne maintenant le bon résultat pour des nombres réels ?

Donner vos explications.

Reprenez le programme en ajoutant float devant input. Cette addition donne maintenant le bon résultat avec des nombres réels.

Copier / coller un exemple.

6) La fonction round(valeur numérique, Nombre de chiffres après la virgule) permet d'arrondir une valeur numérique.

Exemples :

```
>>> round(1.99,1)
>>> 2
>>> round(5.588,2)
>>> 5.59
>>> round(3.14159,4)
>>> 3.1416
```

Créer un programme où deux nombres réels sont saisis dans deux boîtes de dialogue différentes et qui affiche la somme, la soustraction, le produit et la division ( arrondi à 6 chiffres ) de ces deux nombres.

Résultats attendus pour 2 et 3 :

La somme de 2 et de 3 est 5.  
La différence de 2 et de 3 est -1.  
Le produit de 2 et de 3 est 6.  
La division de 2 par 3 est 0,666667.

Copier / coller votre programme et un exemple de calculs.

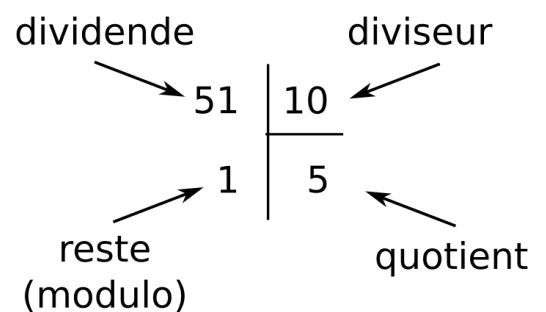
7) Division euclidienne.

Pour diviser deux nombres entiers, vous utilisez cette présentation depuis l'école primaire.

Par exemple,  $51 = 10 \times 5 + 1$ .

Le quotient est 5 et le reste est 1.

L'opération // calcule le quotient et % le reste.



```
>>> 51 // 10
>>> 5
>>> 51 % 10
>>> 1
```

Créer un programme où deux nombres entiers sont saisis dans deux boîtes de dialogue différentes et qui affiche le quotient et le reste de la division de ces deux nombres.

Résultat attendu pour 51 et 10 :

Le quotient de la division de 51 par 10 est 5.

Le reste de la division de 51 par 10 est 1.

Copier / coller votre programme et un exemple de calculs.

8) Un premier exemple de jeu : deviner un nombre entier généré par l'ordinateur.

En Python, des fonctions appartenant à des librairies ( bibliothèques ou modules ) peuvent être appelées.

Par exemple la fonction `randint(n1,n2)` de la librairie `random` crée un nombre entier aléatoire entre les nombres entiers `n1` et `n2`.

Pour appeler toutes les fonctions d'une librairie, il faut utiliser la syntaxe suivante :

```
from NomLibrairie import *
```

que l'on peut traduire par "De la librairie `NomLibrairie`, importer toutes les fonctions".

Enregistrer et exécuter le programme suivant.

```
from random import *
for j in range(10):
    print(randint(0,100), end= " / ")
```

Que fait ce programme ? Quelles sont les valeurs prises par la variable `i` ? Combien de fois est répété l'instruction `print(randint(0,100), end= " / ")` ?

Que génère `randint(0,100)` ?

Enregistrer et exécuter le programme suivant.

```
from random import *
Alea = randint(0,100)
NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
if NombreDonne == Alea :
    print("Vous avez gagné !")
else :
```

```
print("Vous avez perdu ! Le nombre cherché était ",Alea,".")
```

Copier / coller un exemple.

Le jeu fonctionne mais un seul essai est possible. Comment modifier le programme pour obtenir des essais jusqu'à trouver le bon résultat ? Pour nous guider, il faut aussi savoir si le nombre donné est plus petit ou plus grand que Alea.

Il faut utiliser une boucle while ( tant que ). Modifier et rajouter les éléments en **rouge** dans votre programme.

```
from random import *
Alea = randint(0,100)
NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
while NombreDonne != Alea :                               # != signifie différent de
    if NombreDonne > Alea :
        print(" Plus petit !")
        NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
    else :
        print(" Plus grand !")
        NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
print("Vous avez gagné ! Le nombre cherché était bien ",Alea,".")
```

Copier / coller un exemple.

Pour progresser, il est maintenant nécessaire de connaître le nombre d'essais avant de trouver le bon résultat.

Comment procéder ?

L'idée consiste à augmenter de 1 une variable, par exemple appelée Compteur, à chaque fois que la variable NombreDonne est > Alea ou < Alea. La variable Compteur contient à la fin du jeu le nombre d'essais, elle doit être initialisée à 1 en début de jeu car le nombre minimal d'essais est 1.

Ajouter les instructions en **rouge** à votre programme.

```
from random import *
Alea = randint(0,100)
Compteur = 0
NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
while NombreDonne != Alea :                               # != signifie différent de
    if NombreDonne > Alea :
        print(" Plus petit !")
        Compteur = Compteur + 1
        NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
    else :
```

```

print(" Plus grand !")
Compteur = Compteur + 1
NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
print("Vous avez gagné ! Le nombre cherché était bien ",Alea, ".")
print("Vous avez trouvé en ", Compteur, " essai(s).")

```

Copier / coller les résultats de 10 exemples. Est-ce que vous vous améliorez ?  
Calculer la moyenne du nombre d'essais.

Mesure du temps de jeu.

La librairie time contient une fonction time() qui donne le nombre de secondes écoulées entre le 01 janvier 1970 ( origine du temps en informatique ) et l'instant d'utilisation de la fonction.

```

>>> from time import *
>>> time()
>>> 1648069881.802245
>>> time()
>>> 1648070520.366819

```

Convertissez le nombre obtenu en années.

```

1 minute = 60 secondes
1 heure = 60 minutes = 60*60 = 3600 secondes
1 jour = 24 heures = 24*3600 = 86400 secondes
1 année = 365,25 jours = 365,25*86400 = 31557600 secondes

```

```

>>> 1648069881.802245 / 31557600 = 52.22418313820585

```

Environ 52,22 années se sont écoulées entre le 01 janvier 1970 et le 23 mars 2022.

Comment trouver le temps de jeu ?

Au début du jeu, il faut initialiser une variable, par exemple appelée DebutJeu à time() et à la fin du jeu une variable FinJeu à time().

Pour obtenir le temps de jeu, on calcule la différence FinJeu – DebutJeu. Le résultat est donné en secondes.

Ajouter les instructions en rouge à votre programme.

```

from random import *
from time import time

```

```

Alea = randint(0,100)
Compteur = 0
NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
DebutJeu = time()
while NombreDonne != Alea :                                # != signifie différent de
    if NombreDonne > Alea :
        print(" Plus petit !")
        Compteur = Compteur + 1
        NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
    else :
        print(" Plus grand !")
        Compteur = Compteur + 1
        NombreDonne = int(input("Donner un nombre entier entre 0 et 100 "))
FinJeu = time()
TempsJeu = round(FinJeu – DebutJeu,3)
print("Vous avez gagné ! Le nombre cherché était bien ",Alea, ".")
print("Vous avez trouvé en ", Compteur, " essai(s) et en ",TempsJeu," secondes.")

```

Copier / coller les résultats de 10 exemples. Calculer la moyenne du nombre d'essais et du temps écoulé.

## 9) Graphiques avec la librairie matplotlib.pyplot

### 9-1) Placer des points de couleur différente

Enregistrer et exécuter le programme suivant.

```

from matplotlib.pyplot import *
# Importation de la librairie matplotlib.pyplot
plot(5,10,'.',color='#0000FF')
# Place un point de couleur bleue aux coordonnées (5,10)
show()
# Montre le graphique demandé

```

Copier / coller le résultat obtenu.

Enregistrer et exécuter le programme suivant.

```

from matplotlib.pyplot import *
for x in range(0,100):
    plot(x,10,'.',color='#0000FF')
show()

```

Que trace le programme ?

Modifier le programme pour qu'il trace aussi une droite rouge à la hauteur 5 et une droite verte à la hauteur 15.

Enregistrer et exécuter votre programme. Copier / coller le résultat obtenu.

Modifier le programme pour qu'il trace aussi maintenant trois autres droites perpendiculaires aux précédentes aux abscisses  $x = 5$ ,  $x = 10$  et  $x = 15$ . Ces 3 nouvelles droites doivent être respectivement de couleur jaune, magenta et cyan.

Enregistrer et exécuter votre programme. Copier / coller le résultat obtenu.

## 9-2) Représentation graphique de fonctions mathématiques

Enregistrer et exécuter le programme suivant.

```
from matplotlib.pyplot import *  
for x in range(-100,100) :  
    plot(x,x*x,'.',color='#0000FF')  
show()
```

Que trace le programme ? Tracer l'axe de symétrie de cette courbe. Copier / coller le résultat obtenu.

Certaines fonctions mathématiques ne sont pas accessibles directement avec Python, il faut les importer du module math. C'est le cas des fonctions cosinus et sinus.

Compléter le programme suivant pour qu'il représente la fonction cosinus ( rouge ) et sinus ( bleue ) sur l'intervalle  $[-10, 10]$ . Copier / coller le résultat obtenu.

```
from matplotlib.pyplot import *  
from math import *  
for x in range(-100,100) :  
    plot(x/10,cos(x/10),".",color='#FF0000')  
for x in range(-100,100) :  
    plot(...,...., ".",color="#.....")  
show()
```

## E ) Les fonctions :

Nous avons déjà rencontré des fonctions comme par exemple la fonction print(), l'objectif est ici de créer soi-même ses propres fonctions.

### Syntaxe :

```
def Nom_De_La_Fonction(liste de paramètres) :      # Attention à :  
    Instructions                                     # Indentation
```

Pour les parties E-1) et E-2), copier / coller les exemples suivants. Copier / coller les résultats obtenus et expliquer le fonctionnement des programmes.

### E-1) Fonctions sans paramètres :

#### Exemples :

1) Afficher une ligne de caractère:

```
def AfficheCaractere() :  
    for i in range(0,21) :  
        print(" # ", end= " ")
```

```
>>> AfficheCaractere()
```

2) Une fonction simple qui appelle une autre fonction simple :

```
def AfficheCaractere() :  
    for i in range(0,21) :  
        print("#",end=" ")
```

```
def AfficheTripleCaractere() :  
    AfficheCaractere()  
    AfficheCaractere()  
    AfficheCaractere()
```

```
>>> AfficheTripleCaractere()
```

3) Calculer le volume d'une sphère de rayon R :

```
from math import pi  
def VolumeSphere():  
    rayon=float(input("Donner un rayon en cm "))
```



```
print("Le volume de cette sphère est de ",4*1/3*rayon*rayon*rayon*pi,"cm3.")
>>> VolumeSphere()
```

## E-2) Fonctions avec paramètres :

### Exemples :

1) Calculer le volume d'une sphère de rayon R :

```
from math import pi
rayon =float(input("Donner un rayon en cm"))
def VolumeSphere(rayon) :
    print("Le volume de cette shère est de ",4*1/3*rayon*rayon*rayon*pi,"cm3.")
VolumeSphere(rayon)
```

2) Calculer le volume d'une sphère de rayon de 1,2,3,4 à 10 cm:

```
from math import pi
def VolumeSphere(rayon):
    print(round(4*1/3*rayon*rayon*rayon*pi,4),"cm^3")
for i in range(1,11) :
    print("Le volume de la sphère de rayon ",i,"cm est ")
    VolumeSphere(i)
```

3) Calculer l'aire d'un triangle de hauteur h et de base b :

```
a=float(input("Donner la base en cm"))
b=float(input("Donner la hauteur en cm"))

def AireTriangle(base,hauteur) :
    print(round(base*hauteur/2,4),"cm².")

print("L'aire d'un triangle de base",a,"cm et de hauteur",b,"cm est ")
AireTriangle(a,b)
```

4) Définir la fonction  $g(x) = 4x^3 - 2x^2 + 3$  et calculer  $g(0)$ ,  $g(1)$ , ...,  $g(10)$  :

```
def g(x) :
    return 4*x**3 -2*x**2 + 3

for i in range (0,11) :
    print("g(",i,") = ",g(i),end=" ; ")
```

## F ) La librairie tkinter :

La bibliothèque tkinter contient des classes d'objets caractérisés par des attributs (propriétés de l'objet contenues dans des variables de types divers) et des méthodes (actions qui peuvent être réalisées sur l'objet).

La ligne de commande " from tkinter import \* " permet d'importer toutes les classes contenues dans la librairie tkinter.

### F-1) Gestion d'une fenêtre Tk() :

#### Création d'une fenêtre (objet Tk()) :

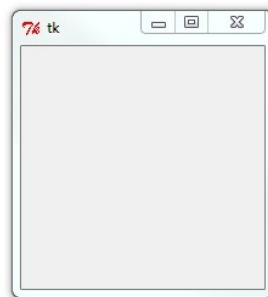
##### Exemple :

```
from tkinter import *          # importation de la bibliothèque graphique tkinter

Fenetre1=Tk()                  # création de l'objet Tk() qui se nomme Fenetre1

Fenetre1.mainloop()           """ activation indispensable d'une boucle de
programme qui scrute tout événement au niveau des périphériques d'entrée ( souris,
clavier, etc ... )"""
```

On obtient l'affichage :



L'objet Fenetre1 créé possède déjà toutes les caractéristiques attendues dans un environnement fenêtré : réduction, agrandissement, fermeture, positionnement de la fenêtre.

Créer un programme permettant l'affichage de l'objet Fenetre2. Copier/coller le résultat obtenu.

## Paramétrage de la fenêtre :

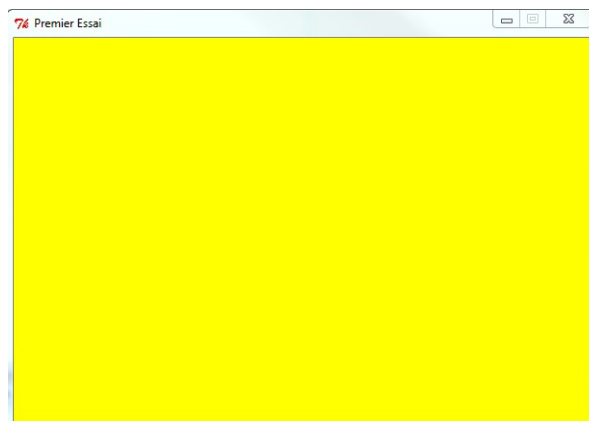
### Exemple :

```
from tkinter import *

Fenetre1=Tk()
Fenetre1.title('Premier Essai') # La fenêtre porte le nom " Premier Essai "
Fenetre1['bg']='yellow'        # L'arrière plan ( back ground ) est jaune (yellow)
Fenetre1.geometry("600x400+500+200") """ la fenêtre a une dimension de
600x400 pixels et se trouve positionné à 500 pixels en abscisses et à 200 pixels en
ordonnées par rapport à l'environnement windows, l'origine du repère étant le coin
supérieur gauche de l'écran """
Fenetre1.resizable(width=NO,height=NO) """ la fenêtre ne peut plus être
redimensionnée, la valeur largeur ( width ) est NO ( au lieu de YES par défaut ) et la
valeur hauteur ( height ) est NO ( au lieu de YES par défaut ). """

Fenetre1.mainloop()
```

On obtient l'affichage :



Remarque : Le choix des couleurs se fait avec le code # RRVVBB où RR, VV et BB peuvent varier entre 00 et FF.

Créer un programme permettant l'affichage de l'objet Fenetre2. La fenêtre se nomme "TEST1", la couleur d'arrière plan est verte et sa dimension est de 200x200. Copier/coller le résultat obtenu.

## F-2) Les objets d'une fenêtre Tk() :

### 1) L'objet **Label()** :

Il permet d'écrire du texte dans la fenêtre et la méthode **place()** appliquée à l'objet Label() permet de positionner l'objet.

Exemple :

```
from tkinter import *
```

```
Fenetre1=Tk()
```

```
Fenetre1.geometry("250x200+500+200")
```

```
Texte1=Label(Fenetre1,text='Bonjour tout le  
monde !',font=("Arial",14),fg='white',bg='red', width=20, height=5)
```

""" Fenetre1 indique la fenêtre dans laquelle le texte doit apparaître, la variable text contient le texte indiqué, la variable font indique la police et la taille des caractères, fg ( foreground ) la couleur du premier plan, bg ( background ) la couleur d'arrière plan, width la largeur de la zone de texte et height la hauteur de la zone de texte """

```
Texte1.place(x=12,y=40)
```

""" la méthode place positionne le texte en x=12 ( abscisse ) et en y=40 ( ordonnée ), l'origine du repère de coordonnées (0;0) se trouve au coin supérieur gauche de la fenêtre. """

```
Fenetre1.mainloop()
```

On obtient l'affichage :



Créer un programme permettant l'affichage de l'objet Fenetre3. La fenêtre se nomme "TEST2", la couleur d'arrière plan est verte et sa dimension est de 100x100. Le texte "Numérique et sciences informatiques" doit apparaître en noir sur fond jaune. Copier/ coller le résultat obtenu.

## 2) L'objet **Entry()** :

Il permet la création de zones de saisie dans la fenêtre graphique choisie.  
La méthode **place()** permet de positionner cet objet.

### Exemple :

```
from tkinter import *
```

```
Fenetre1=Tk()
```

```
Fenetre1.geometry("250x200+500+200")
```

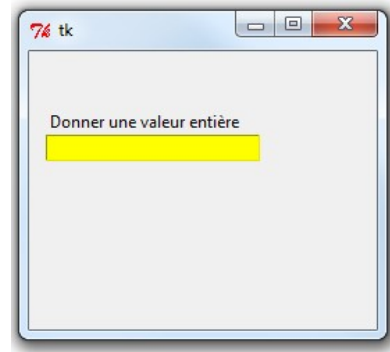
```
Texte1=Label(Fenetre1,text='Donner une valeur  
entière')
```

```
Texte1.place(x=12,y=40)
```

```
Entree1=Entry(Fenetre1,width=25,fg='red',bg='yellow')
```

```
Entree1.place(x=12,y=60)
```

```
Fenetre1.mainloop()
```



Modifier le programme précédent pour que le texte et la zone d'entrée s'affichent au milieu de la fenêtre. Copier/coller le résultat obtenu.

## 3) L'objet **Button()** :

Il permet la création de boutons de commande dans la fenêtre graphique choisie. La méthode **place()** permet de positionner cet objet.

### Exemple :

```
from tkinter import *
```

```
Fenetre1=Tk()
```

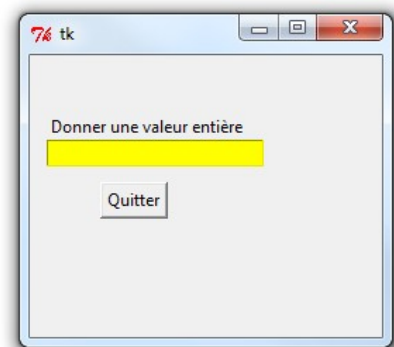
```
Fenetre1.geometry("250x200+500+200")
```

```
Texte1=Label(Fenetre1,text='Donner une valeur entière')
```

```
Texte1.place(x=12,y=40)
```

```
Entree1=Entry(Fenetre1,width=25,fg='red',bg='yellow')
```

```
Entree1.place(x=12,y=60)
```



```

Bouton1=Button(Fenetre1,text='Quitter',command=Fenetre1.destroy)
# L'appui sur le bouton enclenche l'exécution de la commande Fenetre1.destroy
Bouton1.place(x=50,y=90)

Fenetre1.mainloop()

```

Modifier le programme précédent pour que le texte, la zone d'entrée et le bouton "Quitter" s'affichent au milieu de la fenêtre. Copier/coller le résultat obtenu.

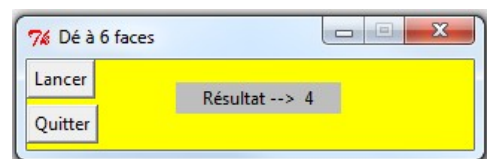
### Exemples d'utilisation de ces trois objets

#### Exemple 1 : Simulation d'un lancer de dé

```

from tkinter import *
from random import randint

```



```

def Nouveau_Lancer() :                # Fonction Nouveau Lancer
    nb=randint(1,6)                   # nb est un nombre entier compris entre 0 et 6
    Texte.set('Résultat --> '+str(nb))
# La variable Texte qui est une chaîne de caractère prend la valeur ' Résultat --> nb
# Les deux variables Résultat et str(nb) qui sont de type caractère sont concaténées
# Pour additionner ces deux chaînes de caractère, on utilise l'opération +
# La méthode set met à jour la variable Texte
Fenetre1=Tk()
Fenetre1.title('Dé à 6 faces')
Fenetre1.geometry("300x60+600+400")
Fenetre1.resizable(width='false',height='false')
Fenetre1['bg']='yellow'
Bouton_Lancer=Button(Fenetre1,text='Lancer',command=Nouveau_Lancer)
# Sur clic du bouton, la fonction Nouveau_Lancer est activée
Bouton_Lancer.place(x=0,y=0)

Bouton_Quitter=Button(Fenetre1,text='Quitter',command=Fenetre1.destroy)
Bouton_Quitter.place(x=0,y=30)

Texte=StringVar() # Création de la variable tkinter Texte qui est du type StringVar
""" Les variables tkinter DoubleVar() mémorisent un nombre réel, IntVar() un
nombre entier et StringVar() une chaîne de caractère. """
Affichage_Resultat=Label(Fenetre1,textvariable=Texte,fg='black',bg='grey',width=15
)

```

```

""" La variable textvariable prend la valeur Texte qui s'affiche dans le Label
Affichage_Resultat """
Affichage_Resultat.place(x=100,y=15)

```

```

Fenetre1.mainloop()

```

Modifier le programme précédent pour simuler un lancer de dé à 20 faces. Les boutons "Lancer" et "Quitter" doivent se trouver maintenant à droite de l'affichage du résultat. Copier/coller le résultat obtenu.

### Exemple 2 : Une multiplication par 5

```

from tkinter import *

```

```

fenetre1=Tk()

```

```

def Multiplication_Par_5() :

```

```

    Resultat_Multiplication.set((5*float(Entree1.get())))) """La fonction calcule le
produit de la valeur entière de Entree1 par 5 et place le résultat dans
Resultat_Multiplication, la méthode get() permet de récupérer la valeur Entree1, la
méthode set() s'occupe de la mise à jour """

```

```

Etiquette1=Label(fenetre1,text="Donner une valeur").place(x=0,y=0)

```

```

Entree1=Entry(fenetre1,width=10)

```

```

Entree1.insert(0,1)          # La valeur par défaut est 1; 0 est un code de la
méthode insert

```

```

Entree1.place(x=5,y=20)

```

```

Resultat_Multiplication=DoubleVar()

```

```

# Création de la variable tkinter Resultat_Multiplication qui est du type DoubleVar

```

```

Etiquette3=Label(fenetre1,textvariable=Resultat_Multiplication)          """Etiquette3
affiche Resultat_Multiplication """

```

```

Etiquette3.place(x=140,y=90)

```

```

Etiquette4=Label(fenetre1,text='Résultat').place(x=0,y=90)

```

```

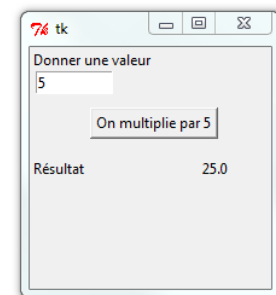
Bouton1=Button(fenetre1,text="On multiplie par
5",command=Multiplication_Par_5) Bouton1.place(x=50,y=50)

```

```

fenetre1.mainloop()

```



Rajouter un bouton de commande et une zone de résultat pour multiplier par 7.  
Copier/coller le résultat obtenu.

Exemple 3 : Afficher une table de multiplication demandée en utilisant un menu déroulant.

```
from tkinter import *
```

```
fenetre1=Tk()
```

```
fenetre1.geometry("290x300+500+200")
```

```
def Table():          # Création de la fonction Table()
    Nombre_Choisi=float(Menu_Deroulant.get())
    Resultat_Multiplication.set((1*Nombre_Choisi))
    Numero_Table.set((1*Nombre_Choisi))
```

```
Affichage1=Label(fenetre1,text="Choisir le numéro de la table de multiplication")
Affichage1.place(x=20,y=0)
```

# L'objet Spinbox est un menu déroulant

```
Menu_Deroulant = Spinbox(fenetre1, from_=1, to=10,width=5,justify=CENTER)
Menu_Deroulant.place(x=115,y=25)
```

```
Resultat_Multiplication=DoubleVar()
Affichage2=Label(fenetre1,textvariable=Resultat_Multiplication)
Affichage2.place(x=180,y=105)
```

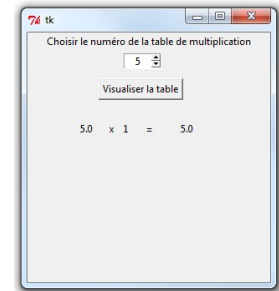
# Affichage2 affiche Resultat\_Multiplication

```
Affichage3=Label(fenetre1,text='x 1 = ').place(x=95,y=105)
```

```
Numero_Table=IntVar()
Affichage4=Label(fenetre1,textvariable=Numero_Table).place(x=60,y=105)
```

```
Bouton1=Button(fenetre1,text="Visualiser la table",command=Table)
Bouton1.place(x=85,y=55)
```

```
fenetre1.mainloop()
```



Modifier le programme précédent pour pouvoir obtenir les cinq premiers résultats de la table de multiplication demandée. Copier/coller le résultat obtenu.



#### 4) L'objet Canvas() :

Cet objet permet de créer une zone de dessin et d'animation.

Exemple 1 : Tracé de droite, cercle, ellipse et polygone.

```
from tkinter import *

fenetre1= Tk()
fenetre1.geometry("380x380+500+200")
fenetre1.resizable(width=NO,height=NO)

# Création de la zone graphique dans " fenetre1 "
Zone_Graphique = Canvas(fenetre1,width=300,height=300,bg='yellow')
Zone_Graphique.place(x=40,y=40)
# Création de Texte1 dans la zone graphique
Texte1=Zone_Graphique.create_text(150,10,text="Objets graphiques")
# Coordonnées du point de départ de la droite (X1,Y1)
# Coordonnées du point d'arrivée de la droite (X2,Y2)
X1,Y1=0,0
X2,Y2=300,300
# Tracé de Droite1, tracé de Droite2 dans la zone graphique
Droite1=Zone_Graphique.create_line(X1, Y1, X2, Y2, fill="black")
Droite2=Zone_Graphique.create_line(300, Y1, 0, Y2, fill="black")

X4,Y4=20,20
# Tracé de Rectangle1 dans la zone graphique
Rectangle1=Zone_Graphique.create_rectangle(X1, Y1, X4, Y4, fill="red")
# Tracé de Cercle1 dans la zone graphique
Cercle1=Zone_Graphique.create_oval(100, 100, 150, 150, fill="red")
# Tracé de Ellipse1 dans la zone graphique
Ellipse1=Zone_Graphique.create_oval(150, 150, 210, 180, fill="blue")
# Tracé de Polygon1 dans la zone graphique
Polygone1=Zone_Graphique.create_polygon(300,300,250,200,200,200, fill="black")

fenetre1.mainloop()
```

Modifier le programme précédent pour intervertir la position du carré et du triangle, de l'ellipse et du cercle. Copier/coller le résultat obtenu.

## Exemple 2 : Division euclidienne.

```
from tkinter import *

fenetre1=Tk()
fenetre1.title('Division euclidienne')
fenetre1.geometry("250x210+500+200")
fenetre1.resizable(width=NO,height=NO)

def Division() :
    Quotient.set((int(Entree1.get())//int(Entree2.get()))
    Reste.set((int(Entree1.get())%int(Entree2.get())))

"""La fonction calcule le quotient ( // ) de la valeur entière de Entree1 par Entree2 et
place le résultat dans Quotient,
calcule le reste ( % ) de la valeur entière de Entree1 par Entree2 et place le résultat
dans Reste,
la méthode get() permet de récupérer la valeur Entree1 et Entree2, la méthode set()
s'occupe de la mise à jour """

""" Création et positionnement d'une zone graphique """
Zone_Graphique = Canvas(fenetre1,width=250,height=250)
Zone_Graphique.place(x=0,y=0)
""" Création du symbole de division par tracé de deux droites """
Droite1=Zone_Graphique.create_line(125, 20, 125, 130, fill="black")
Droite2=Zone_Graphique.create_line(125, 60, 200, 60, fill="black")
# Objets Label, affichage du texte
Etiquette1=Label(fenetre1,text="Dividende").place(x=50,y=10)
Etiquette2=Label(fenetre1,text="Diviseur").place(x=150,y=10)
Etiquette3=Label(fenetre1,text='Reste').place(x=60,y=65)
Etiquette4=Label(fenetre1,text='Quotient').place(x=150,y=65)
# Objets Entry, zones de saisie
Entree1=Entry(fenetre1,width=5,justify=CENTER)
Entree1.insert(0,"?")
# La valeur par défaut est "?"; 0 est un code de la méthode insert
Entree1.place(x=60,y=30)
Entree2=Entry(fenetre1,width=5,justify=CENTER)
Entree2.insert(0,"?")
# La valeur par défaut est "?"; 0 est un code de la méthode insert
Entree2.place(x=155,y=30)
# Création des variables tkinter de type IntVar() Quotient et Reste
Quotient=IntVar()
Reste=IntVar()
```

```

# Affichage des résultats : quotient et reste
"""Etiquette5 affiche Quotient """
Etiquette5=Label(fenetre1,textvariable=Quotient,bg="white",fg="black").place(x=170,y=95)
"""Etiquette6 affiche Reste """
Etiquette6=Label(fenetre1,textvariable=Reste,bg="white",fg="black").place(x=70,y=95)
# Bouton de commande " Diviser "
Bouton1=Button(fenetre1,text="Diviser",command=Division)
Bouton1.place(x=100,y=170)
fenetre1.mainloop()

```

Modifier le programme pour faire apparaître le résultat de l'addition, de la soustraction et de la multiplication du dividende et du diviseur. Copier/coller le résultat obtenu.

Exemple 3 : Affichage aléatoire de disques dans une zone graphique.

```

from tkinter import *
from random import *
# Liste des couleurs utilisées pour le cercle
Couleur=['blue','yellow','black','green2','red','pink','azure','cyan']
# Création et caractéristique de la fenêtre principale
fenetre1 = Tk()
fenetre1.title('Cercle aléatoire')
fenetre1.geometry("400x400+500+200")
fenetre1.resizable(width=NO,height=NO)
# Dessine un cercle de centre (x,y) et de rayon r, de couleur choisie dans la liste
Couleur
def Cercle():
    x = randint(0,Largeur)
    y = randint(0,Hauteur)
    r = 5
    Couleur_Cercle=Couleur[randint(0,7)]
    Canevas.create_oval(x-r, y-r, x+r, y+r, outline='black', fill=Couleur_Cercle)
# Efface les éléments de la zone graphique
def Effacer():

    Canevas.delete(ALL)
# Création de la zone graphique
Largeur = 300
Hauteur = 300
Canevas = Canvas(fenetre1, width = Largeur, height =Hauteur, bg ='white')
Canevas.place(x =50, y =20)

```

```

# Création du bouton de commande " Action "
BoutonAction = Button(fenetre1, text ='Action', command = Cercle)
BoutonAction.place(x=130,y=350)
# Création du bouton de commande " Effacer "
BoutonEffacer = Button(fenetre1, text ='Effacer', command = Effacer)
BoutonEffacer.place(x=180,y=350)
# Création du bouton de commande " Quitter "
BoutonQuitter = Button(fenetre1, text ='Quitter', command = fenetre1.destroy)
BoutonQuitter.place(x=230,y=350)
fenetre1.mainloop()

```

Modifier le programme pour faire afficher 5 cercles à la fois de même couleur.  
Copier/coller le résultat obtenu.

Exemple 4 : Rebond d'une balle dans une zone graphique.

```

from tkinter import *
from math import *
from random import*
# Création de la fenêtre principale
fenetre1 = Tk()
fenetre1.title("Rebond Balle")
fenetre1.geometry("500x410+500+200")
fenetre1.resizable(width=NO,height=NO)
LARGEUR = 480
HAUTEUR = 320
RAYON = 15 # Rayon de la balle
# Position initiale de la balle au milieu de la zone graphique
X = LARGEUR/2
Y = HAUTEUR/2
# Création de la zone graphique
Canevas = Canvas(fenetre1,height=HAUTEUR,width=LARGEUR,bg='white')
Canevas.place(x=8,y=10)
# Création de la balle
Balle = Canevas.create_oval(X-RAYON,Y-
RAYON,X+RAYON,Y+RAYON,width=1,fill='red')
# Direction initiale aléatoire de la balle
vitesse = uniform(1.8,2)*5
angle = uniform(0,2*pi)
DX = vitesse*cos(angle)
DY = vitesse*sin(angle)

```

### # Déplacement de la balle

def deplacement():

    global X,Y,DX,DY,RAYON,LARGEUR,HAUTEUR

#### # Rebond à droite

    if X+RAYON+DX > LARGEUR:

        X = 2\*(LARGEUR-RAYON)-X

        DX = -DX

#### # Rebond à gauche

    if X-RAYON+DX < 0:

        X = 2\*RAYON-X

        DX = -DX

#### # Rebond en bas

    if Y+RAYON+DY > HAUTEUR:

        Y = 2\*(HAUTEUR-RAYON)-Y

        DY = -DY

#### # Rebond en haut

    if Y-RAYON+DY < 0:

        Y = 2\*RAYON-Y

        DY = -DY

    X = X+DX

    Y = Y+DY

#### # Affichage

    Canevas.coords(Balle,X-RAYON,Y-RAYON,X+RAYON,Y+RAYON)

#### # Mise à jour toutes les 30 ms

    fenetre1.after(30,deplacement)

deplacement()

fenetre1.mainloop()

Ajouter au programme un bouton de commande pour arrêter l'animation, un autre pour la redémarrer. La balle doit être jaune. Copier/coller le résultat obtenu.

Exemple 5 : Commande d'un objet graphique avec le clavier.

from tkinter import \*

def Clavier(event):

    """" Gestion de l'événement Appui sur une touche du clavier """"

    global PosX,PosY

```

touche = event.keysym
print(touche)
# déplacement vers le haut
if touche == 'a':
    PosY -= 20
# déplacement vers le bas
if touche == 'q':
    PosY += 20
# déplacement vers la droite
if touche == 'm':
    PosX += 20
# déplacement vers la gauche
if touche == 'l':
    PosX -= 20
# on dessine le pion à sa nouvelle position
Canevas.coords(Pion,PosX -10, PosY -10, PosX +10, PosY +10)
# Création de la fenêtre principale
Mafenetre = Tk()
Mafenetre.title('Pion')

# position initiale du pion
PosX = 230
PosY = 150

# Création d'un widget Canvas (zone graphique)
Largeur = 480
Hauteur = 320
Canevas = Canvas(Mafenetre, width = Largeur, height =Hauteur, bg ='white')
Pion = Canevas.create_oval(PosX-10,PosY-
10,PosX+10,PosY+10,width=2,outline='black',fill='red')
Canevas.focus_set()
Canevas.bind('<Key>',Clavier)
Canevas.pack(padx =5, pady =5)

# Création d'un widget Button (bouton Quitter)
Button(Mafenetre, text ='Quitter', command =
Mafenetre.destroy).pack(side=LEFT,padx=5,pady=5)

Mafenetre.mainloop()

```

Quelles sont les touches du clavier qui permettent de modifier la position du pion ?  
 Modifier le programme pour attribuer le mouvement aux flèches de direction. Le pion

doit être maintenant carré, de couleur bleue et se trouvait exactement au centre de la fenêtre graphique au lancement de l'application. Copier/coller le résultat obtenu.

Exemple 6: Commande ( saisir-déplacer et lâcher ) d'un objet graphique avec la souris.

```
from tkinter import *
```

```
def Clic(event):
```

```
    """ Gestion de l'événement Clic gauche """
```

```
    global DETECTION_CLIC_SUR_OBJET
```

```
    # position du pointeur de la souris
```

```
    X = event.x
```

```
    Y = event.y
```

```
    print("Position du clic -> ",X,Y)
```

```
    # coordonnées de l'objet
```

```
    [xmin,ymin,xmax,ymax] = Canevas.coords(Carre)
```

```
    print("Position objet -> ",xmin,ymin,xmax,ymax)
```

```
    if xmin<=X<=xmax and ymin<=Y<=ymax:
```

```
        DETECTION_CLIC_SUR_OBJET = True
```

```
    else:
```

```
        DETECTION_CLIC_SUR_OBJET = False
```

```
    print("DETECTION CLIC SUR OBJET ",DETECTION_CLIC_SUR_OBJET)
```

```
def Drag(event):
```

```
    """ Gestion de l'événement bouton gauche enfoncé """
```

```
    X = event.x
```

```
    Y = event.y
```

```
    print("Position du pointeur -> ",X,Y)
```

```
    if DETECTION_CLIC_SUR_OBJET == True:
```

```
        # Limite de l'objet dans la zone graphique
```

```
            if X<0: X=0
```

```
            if X>Largeur: X=Largeur
```

```
            if Y<0: Y=0
```

```
            if Y>Hauteur: Y=Hauteur
```

```
        # Mise à jour de la position de l'objet (drag)
```

```
        Canevas.coords(Carre,X-TailleCarre,Y-
```

```
TailleCarre,X+TailleCarre,Y+TailleCarre)
```

```
DETECTION_CLIC_SUR_OBJET = False
```

# Création de la fenêtre principale

```
Mafenetre = Tk()
```

```
Mafenetre.title("Saisir-Déplacer et lâcher")
```

# Création d'un widget Canvas

```
Largeur = 480
```

```
Hauteur = 160
```

```
TailleCarre = 20
```

```
Canevas = Canvas(Mafenetre,width=Largeur,height=Hauteur,bg='white')
```

# Création d'un objet graphique

```
Carre = Canevas.create_rectangle(0,0,TailleCarre*2,TailleCarre*2,fill='maroon')
```

# La méthode bind() permet de lier un événement avec une fonction

```
Canevas.bind('<Button-1>',Clic) # événement clic gauche (press)
```

```
Canevas.bind('<B1-Motion>',Drag) # événement bouton gauche enfoncé (hold down)
```

```
Canevas.focus_set()
```

```
Canevas.pack(padx=10,pady=10)
```

```
Mafenetre.mainloop()
```

Que permet de réaliser ce programme ? Donner une description de son fonctionnement. A quoi correspondent les valeurs affichées dans le mode "console" ? Quel est l'origine du repère ?