Partie 3 – Chapitre 1 : Premiers pas en Python

Voici une présentation des instructions de base du langage Python. Elles seront rapidement approfondies et complétées.
Le fichier monPremierProgramme.py permet de tester les instructions présentées.

I. Utilisation de Python

Nous utiliserons la version 3.3 de Python.
Sur Windows, elle est intégrée à la distribution d'EduPython qui est enrichie d'une interface de développement et des bibliothèques les plus courantes. Celle-ci se télécharge via le site http://edupython.tuxfamily.org

Pour les autres systèmes d'exploitations, il faudra se contenter de la version "basique" de Python qui se télécharge sur le site officiel : http://www.python.org/download/ (l'installation des bibliothèques se fera alors manuellement).

II. Mise en œuvre : programme Python dans un fichier

Avec EduPython (Windows) :
Suivre les instructions données en classe.

Sans EduPython (ou sur Linux/ Mac) :
Avec l'IDLE, créer un nouveau fichier (File – New window).
Taper le code, sauvegarder avec une extension *.py* puis exécuter avec un : run → run module (touche F5).

III. Instructions de base

Affectation d'une valeur à une variable

Il suffit d'écrire le nom de la variable, le symbole = et la valeur qu'on souhaite lui affecter.
Exemples :

```
a = 5
b = 6.3
c = a*b
```

Il n'est pas nécessaire de déclarer les variables au préalable comme on le faisait dans Algobox.

Ecriture à l'écran : afficher une variable, un message ou les deux

Affichage d'un texte :

```
print ("Premier test de Python")
```

Affichage de variables :

```
print (a)
print (b)
print (c)
```

Affichage simultané de texte et de variables :

```
print ("Le produit de",a,"par",b,"est :",c)
```

Saisie de la valeur d'une variable

Pour une chaîne de caractères :

```
nom = input("Quel est votre prénom : ")
```

Pour saisir un entier, on fait précéder « input » de « int(…) »

```
age = int(input("Entrez votre age : "))
```

Pour saisir un réel, on fait précéder « input » de « float(…) »

```
taille = float(input("Entrez votre taille : "))
```

Faire les exercices 1, 2, 3 et 4 de la feuille d'exercices

Instruction conditionnelle SI…ALORS… (SINON …)

Il n'y a pas de END !
Pour indiquer le contenu des différentes zones, on utilise une tabulation.
Il faut également mettre des deux-points en fin de ligne du if et du else.

```
if (age>=18) :
    print ("Vous êtes majeur.\nVous pouvez entrer")
else :
    print ("Vous êtes encore trop jeune pour entrer.")
    print ("Il faut encore attendre",18-age,"ans.")
```

Le caractère spécial « \n » correspond au retour à la ligne (touche Return / Entrée).

On peut combiner les conditions avec les opérateurs logiques « **or** » (OU) et « **and** » (ET).
Comme avec Algobox, « == » s'utilise pour comparer deux valeurs et « **!=** » pour voir si deux valeurs sont différentes.

Faire les exercices 5 et 6 de la feuille d'exercices

Boucle itérative POUR … DE A…

Il n'y a pas de END !
Pour indiquer le contenu de la boucle, on utilise une tabulation.
Il faut également mettre des deux-points en fin de ligne du for.

Attention : le dernier entier n'est pas atteint !

Ainsi le programme suivant correspond à la boucle « pour i allant de 1 à **10** » :

```
for i in range(1,11):
    print (i,"multiplié par 9 donne",9*i)
```

Faire l'exercice 7 de la feuille d'exercices

Boucle itérative TANT QUE …

Il n'y a pas de END !
Pour indiquer le contenu de la boucle, on utilise une tabulation.
Il faut également mettre des deux-points en fin de ligne du while.

```
a = 0
while (a<5):
    print ("Dans la boucle, a =",a)
    a = a+1
print ("Une fois sorti de la boucle, a =",a)
```

Faire l'exercice 8 de la feuille d'exercices

Commentaires

Les commentaires peuvent s'écrire sur une ligne si elle commence par un # (dièse) :

```
# Commentaire sur une ligne : ......
```

Les commentaires peuvent aussi s'écrire sur plusieurs lignes en commençant et terminant par " " " :

```
"""
Commentaires
sur plusieurs
lignes
"""
```

Exercices

Exercice 1 :
Ecrire un algorithme dont le but est de demander à l'utilisateur d'entrer les dimensions d'un rectangle et qui renvoie l'aire de celui-ci.
Réaliser le programme *aire.py* correspondant.

Exercice 2 :
Ecrire un programme *operations.py* qui demande à l'utilisateur d'entrer deux réels et qui affiche la somme, le produit et la moyenne de ces deux nombres.

Exercice 3 :
Ecrire un programme *choisirUnNombre.py* pour traduire les opérations suivantes :
Choisir un nombre. Lui ajouter 5. Multiplier le résultat par 2. Soustraire 7 au nombre trouvé. Afficher le résultat.

Exercice 4 :
Voici ci-contre la proposition d'un programme *echange.py* pour échanger les valeurs de deux variables :
Tester le et expliquer pourquoi il ne remplit pas le rôle voulu.
Modifier le programme pour que l'échange ait bien lieu.
Ecrire l'algorithme correspondant.

```
a = 5
b = 3
print ("a = ",a)
print ("b = ",b)
a = b
b = a
print ("a = ",a)
print ("b = ",b)
```

Exercice 5 :
Ecrire un algorithme qui demande à l'utilisateur d'entrer deux entiers et qui renvoie le minimum des deux.
Ecrire un programme *min.py* correspondant.

Exercice 6 :
Ecrire un algorithme qui demande à l'utilisateur d'entrer sa moyenne au baccalauréat et qui renvoie le message correspondant : échec, passable, AB, B ou TB. Ecrire le programme *mention.py* correspondant.

Exercice 7 :
On considère le corps de l'algorithme ci-contre :

a) Détailler ce que fait l'algorithme quand l'utilisateur saisit la valeur n = 6.
b) Compléter les entêtes de l'algorithme.
c) Ecrire le programme *mystere.py* correspondant.

```
DEBUT
  Afficher le message
    "Entrer un entier n plus grand que 0"
  Saisir n
  S prend la valeur 0
  Pour k allant de 0 à n
    S prend la valeur S+k
  Afficher S
FIN
```

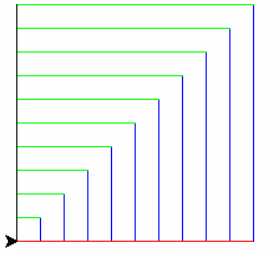
Pour les plus rapides seulement :
En s'inspirant de l'exercice, réaliser un programme *factorielle.py* pour obtenir le nombre appelé factorielle n, noté n! ,
défini par $n! = 1 \times 2 \times \ldots \times n$ pour $n \geq 1$ et par $0! = 1$.

Exercice 8 (Approche graphique des boucles POUR et TANT QUE) :
a) Tester l'algorithme *equilateral.py*

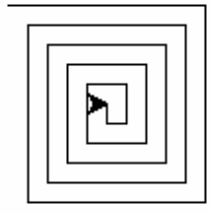
Les principales fonctions du module turtle sont :

–	reset()	Efface le dessin
–	goto(x,y)	Se déplace auxcoordonnées x et y
–	forward(distance)	Avance d'une distance donnée en pixels
–	backward(distance)	Recule
–	up()	Releve le crayon (permet d'avancer sans dessiner)
–	down()	Abaisse le crayon (pour pouvoir recommencer à dessiner)
–	color(couleur)	Couleur peut être une chaîne prédéfinie ('red', 'blue', etc.)
–	left(angle)	Tourne à gauche d'un angle donné (exprimé en degré)
–	right(angle)	Tourne à droite

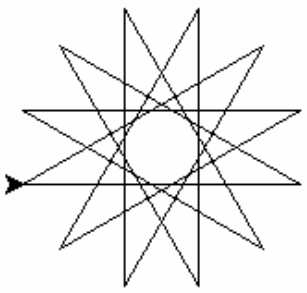
b) Enregistrer le programme précédent sous le nom *carre.py* et modifier le pour obtenir un carré.
c) Ecrire les programmes permettant d'obtenir les dessins suivants :



Aide : La boucle « Pour i de 1 à 10 » s'écrit "**for i in range (1,11) :** "



Aide : La boucle « Tant que taille ≠ 0 » s'écrit " **while taille !=0 :** "
Le premier segment a une taille de 100 puis on enlève 10 tous les deux segments



Aide : C'est un angle de 150° qui intervient.