Informatique SV L2
TP 1

1 Prise en main de Python

1.1 Démarrer l'interpréteur Python

Pour exécuter du code Python, il faut ouvrir l'interpréteur Python. Pour cela :

- Ouvrez la console : Menu UCP, Console
- Tapez *python* suivie de la touche entrez.

Vous venez de taper la commande *python* qui a démarré le logiciel correspondant. Python affiche le prompt $>>>$ indiquant qu'il attend qui vous entrez une instruction.

1.2 Opération sur des nombres et affectation de variable entière (integer)

Exécutez les instructions suivantes, et observez :

```
5 + 7
8 * 9
8 * (4 + 8)
i = 4
i
j = i * 5
j
i = 5
i
j
```

Pourquoi la valeur de la variable j n'a pas changée quand on a modifié i ?

1.3 Opération sur des chaînes de caractères

A la suite entrez :

```
codon = 'acttga'
codon = codon + 'aaacggt'
codon
codon = codon * 3
codon
len(codon)
type(codon)
codon[4]
```

Que renvoi les fonctions *len,type* ? A quel valeur correspond *codon[4]* ? *codon[0]* ?

1.4 Les types des variables

3 principaux types de variable existent :
- int : Integer, nombre entier
- float : Nombre à virgule flottante
- str : String, chaîne de caractère

Entrez les instructions suivantes :

```
pi = 3.1415926
nbr = 18
nom =  'Dupont'
```

Utilisez la fonction *type* pour déterminer le type des variables *pi*, *nbr* et *nom*

```
nom2 = 'Dupont' + 4
```

Que se passe t'il ? Pourquoi ? Si on souhaite affecter à *nom2* la chaîne de caractère *'Dupont4'* que faut-il faire ?

```
pi18 = pi * nbr
```

Quel est le type de la variable *pi18* ? Vérifiez avec la fonction *type*

```
pi_entier = int(pi)
pi_entier
type(pi_entier)
```

A quoi sert la fonction int() ?

1.5 Communication avec l'extérieur

Les fonction input() renvoi la valeur tapez par l'utilisateur. Si cette valeur est un nombre entier, elle renverra une valeur de type *int*, si elle est réel elle sera de type *float*, si elle est entre guillemet, ça sera de type *str*. Si l'utilisateur rentre une chaîne de caractère sans guillemet, ça renverra une erreur. Pour éviter cela on peut utiliser une fonction similaire appelé *raw_input* qui renvoi toujours une chaîne de caractère. On utilisera plutôt input pour demander un nombre, et raw_input une chaîne de caractère.

```
print 'Quel est votre prénom ?'
prenom = raw_input()
```

Après avoir entrer la seconde ligne, le programme demande une chaine de caractère, qui la stockera dans la variable *prenom*. Entrez le prénom avant de continuer.

```
print 'Quel age as tu' ?
age = input()
```

Le programme demande l'age, entrez le et continuez.

```
print 'Salut', prenom
print 'Tu as', age, 'ans'
```

2 Instruction conditionnelle et répétitive

2.1 Instruction conditionnelle et bloque d'instruction

Entrez les instructions suivantes. Attention à bien indenter les instructions avec la touche tabulation (à droite de la touche A, avec les deux fleches). Ne pas oublier d'aller deux fois à la ligne à la fin du bloc, pour exécuter l'instruction conditionnelle.

```
n = 5
if ( (n % 2) == 0 ):
  print n, 'est pair'
else:
  print n, 'est impair'
```

Remarque : l'opérateur % calcul le reste d'une division.

2.2 Garder vos programmes dans un fichier

Lorsque vous quittez l'interpréteur Python (avec Ctrl+D), vous perdez votre programme. Pour le conservez, il faut écrire le code dans un fichier. Pour cela ouvrez un éditeur, ici on utilise Kate. Pour ouvrir Kate allez dans le menu UCP. Enregistrez le fichier avec l'extension .py (ex : test.py). Entrez le programme suivant :

```
n = input()
if ( (n % 2) == 0 ):
  print n, 'est pair'
else:
  print n, 'est impair'
```

L'éditeur Kate colore automatiquement votre code pour qu'il soit plus facilement lisible. Kate possède une console intégré, qu'il s'appel 'Terminal'. Ouvrez-le en appuyant sur le bouton Terminal en bas. Pour exécuter le programme entrez dans la console la commande *python votre_fichier.py* (ex : *python test.py).*

Attention : Avant d'exécuter le programme pensez à l'enregistrer

2.3 Instruction répétitive : les boucles

Dans un nouveau fichier nommé *boucle.py* entrez le code suivant et exécutez le :

```
#Programme qui salut
print 'Votre prénom ?'
prenom = raw_input()
i = 0
while(i<15):
  print 'salut', prenom
  i += 1                        #l'instruction i+=1 équivaut à i = i + 1
```

1. Combien fois est exécuté le bloque d'instruction qui suit l'instruction *while* ?
2. Quel est la valeur affecté à la variable *i* à la sortie de la boucle *while* ?

2.4 Boucle imbriquée

Dans un nouveau fichier nommé *boucle_imbrique.py* entrez le code suivant et exécutez le :

```
#programme affichant des dièses
i=0
while(i<10):
  j=0
  while(j<i):
    print '#',
    j += 1
  i += 1
  print ''
```

1. Combien de fois sera exécuté l'instruction *print '#' ?*
2. Exécutez le programme pour vérifier. La virgule qui suit le *print '#'*, sert à ne pas avoir de retour à la ligne.

3 Calcul mathématique et fonctions

Rappels de cours

La librairie (ou module) mathématique de python

Python possède un grand nombre de modules qui permettent d'ajouter des fonctionnalités au langages. Pour utiliser une fonction prédéfinis contenue dans un module, on doit d'abord l'importer en utilisant les instructions from/import. Ex :

```
from math import *
```

Cette ligne signifie "On importe toutes les fonctions du module nommé math". Si on souhaite seulement importer la fonction *sin* (pour que le programme s'exécute plus rapidement au démarrage), on peut écrire :

```
from math import sin
```

Pour connaître l'ensemble des librairies disponibles par défaut dans Python, et savoir comment utiliser leurs fonctions on peut consulter la documentation officielle : http://docs.python.org/lib/

Définir ses propres fonctions

On peut définir ses propres fonctions. Pour cela on utilise l'instruction *def* suivi éventuellement de paramètres de la fonctions, et du bloc d'instruction qui sera exécuté lors de l'appel de la fonction. L'instruction *return* permet de définir la valeur que la fonction renvoie. Exemple d'une fonction qui renvoi le volume d'une sphère $V = \frac{4}{3}\Pi R^3$

```
from math import pi   #permet d'avoir la valeur de pi
#Fonction qui calcul le volume d'une sphère de rayon r
def volumeSphere(r):
  r3 = r**3                  #calcul de R^3
  v = 4 * pi * r3 / 3
  return v                   #volumeSphere renvoie la valeur de v
#Appel de la fonction volumeSphere pour
#calculer le volume d'une sphère de rayon 5
volume = volumeSphere(5)
#Affiche le résultat
print volume
```

3.1 Minimum

```
def minimum(a,b):
  #partie à completer

#programme principale
print 'Entrez le premier nombre'
nb1 = input()
print 'Entrez le second nombre'
nb2 = input()
print 'Le minimum est', minimum(nb1,nb2)
```

1. Écrire le contenu de la fonction *minimum* qui renvoi le minimum entre a et b.
2. Tester le programme
3. Créer une seconde fonction appelée *minimum3* , qui appel la fonction *minimum* deux fois pour calculer le minimum entre trois nombres. Tester cette fonction.

4 Jeux : trouver le nombre juste

Le but de l'exercice qui va suivre et de faire deviner un nombre par l'utilisateur en lui indiquant si le nombre proposé est trop grand ou trop petit.

1. Dans un nouveau fichier entrez le code correspondant à ce programme. Pour indication en voila les grandes lignes
   - Demander à l'utilisateur le nombre à deviner (Pour pouvoir jouer, on considère que c'est une autre personne qui rentre ce nombre). On affichera "Entrez le nombre à deviner"

- Tant que l'utilisateur n'a pas trouvé le bon nombre :
  - On demande un nombre (on affichera "Entrez un nombre")
  - On indique si il trop grand, trop petit, ou si il a gagné
- Afficher "Fin de la partie"

2. Écrire une fonction *demanderNombre* correspondant à la partie "demander une nombre", qui renvoi un nombre demandé à l'utilisateur (En affichant "Entrez un nombre"). Modifier le programme pour utiliser cette fonction à deux reprises. Ainsi on réutilise le code qui sert à demander un nombre.

3. Ajouter une variable qui permet de compter le nombre d'essai. Afficher le nombre d'essai à la fin de la partie

4. Modifier la fonction *demanderNombre* pour qu'elle prenne un paramètre correspondant au message à afficher. Ainsi les deux messages différents pourront être affiché.

5. Pour pouvoir jouer tout seul, on peut demander à l'ordinateur un nombre tiré aléatoirement. Modifier la première partie du programme pour obtenir ce nombre au lieu de le demander. On utilisera une fonction du module *random* en consultant sa documentation sur Internet.