I.S.N. TP 1 – Découverte de Python 3

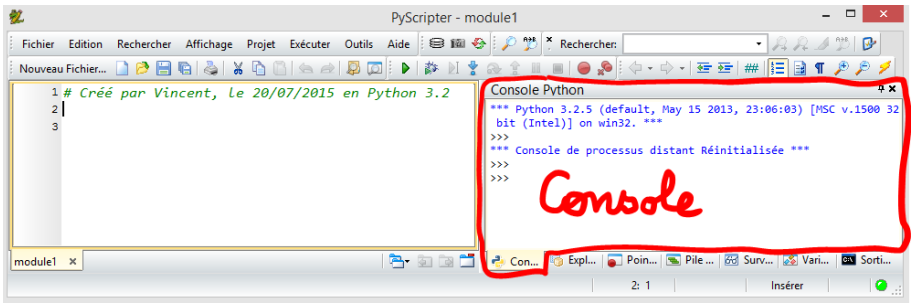
1 Introduction

Vous avez déjà rencontré des algorithmes dans le cadre des mathématiques en seconde et en première, et vous les avez programmés sur une machine (avec la calculatrice ou avec Algobox par exemple). Pour l'ISN, nous avons choisi le langage de programmation Python. On l'utilise aussi dans l'enseignement supérieur (IUT, CPGE,...) et dans le monde de l'entreprise.

L'objectif de ce premier TP est de vous permettre de découvrir certains concepts importants pour cette année. Le choix a été fait de ne pas tout expliquer ou tout formaliser dès maintenant. Si besoin, vous pouvez vous tourner vers des ressources plus complètes, ou demander à un de vos professeurs.

2 Premier contact avec le mode interactif de Python

1. Ouvrir le logiciel EduPython, repérer l'onglet intitulé « Console Python » comme dans l'image ci-dessous.



2. Utiliser la console Python comme une calculatrice, en saisissant les expressions suivantes, puis en en recopiant ce qui est affiché par l'interpréteur :

```
50+3*10
40*0.5
20
23//5
60/5
60//5
23%5
2**5
```

3. Déterminer, parmi les expressions de la question 2, celles qui donnent un résultat de type entier, et celles qui donnent un résultat de type nombre a virgule flottante. On peut vérifier en utilisant l'instruction `type(expression)` comme ci-dessous :

```
type(3*4)
type(3*4.0)
```

4. Déterminer le rôle des opérateurs `//` `/` `%` `**` en utilisant le vocabulaire suivant (donné dans le désordre) :
   - quotient de la division (décimale) de ... par ...
   - calcul de ... exposant ...
   - quotient de la division euclidienne de ... par ...
   - reste de la division euclidienne de ... par ...

3 Les variables

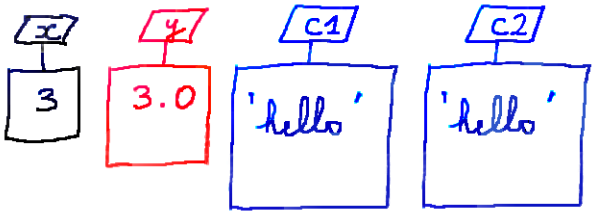
3.1 Affectation d'une valeur à une variable

On écrit `variable=valeur` pour réaliser une affectation. Attention à respecter l'ordre : le nom de la variable est toujours à gauche du symbole =

```
# ceci est un commentaire
x = 3   # affectation d'un entier
y = 3.0 # affectation d'un nombre a virgule flottante
c1 = 'hello'  # affectation d'une chaine de caracteres
c2 = "hello"  # les guillemets jouent le meme role que les apostrophes
```

Après avoir réalisé ces quatre affectations, l'état de la mémoire peut être représenté par le schéma suivant, dans lequel chaque variable est représentée par une boîte avec son étiquette, le nom de la variable étant placé sur l'étiquette, et le contenu de la variable étant placé dans la boîte.

Dans cette représentation, on a choisi des boîtes de formes différentes, pour illustrer les différents types de variables (entier, nombre à virgule flottante, chaîne de caractères).



On peut aussi réaliser un tableau de l'état de la mémoire :

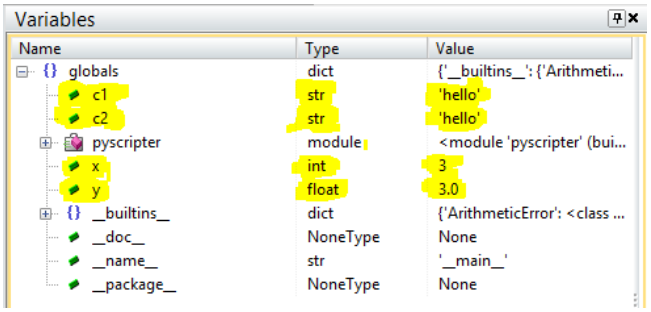
x	y	c1	c2
3	3.0	'hello'	'hello'

3.2 Affichage du contenu d'une variable

Pour afficher le contenu d'une variable, on peut saisir son nom dans la console :

```
x
y
c1
c2
```

On peut également utiliser l'onglet « Variables » :



3.3 Exercices : réaffectation

Quelle est la valeur de la variable `tmp` après l'exécution de ces instructions ? (On pourra faire un tableau de l'état de la mémoire)

```
tmp = 2
tmp = tmp + 8
tmp = tmp * tmp
tmp = tmp //2
```

Quelles sont les valeurs des variables `a` et `b` après l'exécution de ces instructions ? (On pourra faire un tableau de l'état de la mémoire)

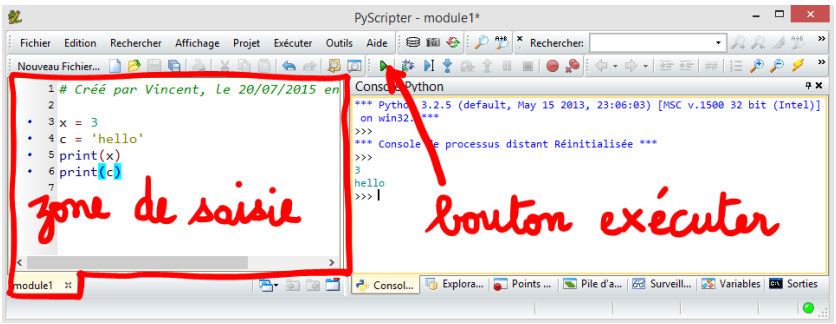
```
a = 2
b = 3
a = a + b
b = a
a = a + 1
b = a * b
```

4 Premier script Python

4.1 Principe

Jusqu'à présent, nous avons utilisé le mode interactif de Python, c'est-à-dire que l'on écrivait directement les instructions dans la console, et chaque instruction était exécutée immédiatement après la saisie de la touche Entrée.

Il existe une autre façon de procéder. On peut écrire les instructions dans un fichier texte, et ensuite on demande à l'interpréteur d'exécuter toutes les instructions, les unes après les autres. L'avantage est qu'on peut sauvegarder les instructions dans un fichier que l'on peut réutiliser plus tard. Lorsqu'on fait ceci, on dit qu'on écrit un script Python.



4.2 Affichage du contenu d'une variable

Lorsqu'on souhaite afficher le contenu d'une variable dans la console, on utilise `print` comme ci-dessous :

```
print(x)  # pour afficher le contenu de x
print(x, y) # pour afficher x et y sur la meme ligne
print(c, d, sep="") # pour afficher c et d, sans espace de separation
print(t, end="") # pour afficher t, sans aller a la ligne
```

4.3 Entrée des données

L'entrée des données s'effectue ainsi :

```
z=int(input("Entrez un entier"))    # pour saisir un entier
t=float(input("Entrez un nb a virgule flottante")) # pour saisir un nombre a virgule
    flottante
e=input("Entrez une chaine de caracteres")    # pour saisir une chaine de caracteres
```

4.4 Exercice

Écrire un script qui demande le nom de l'utilisateur et qui affiche Bonjour suivi du nom de l'utilisateur. (Ce script est court : deux lignes suffisent). Par exemple, si l'utilisateur saisit Alice, le programme affiche :

```
Bonjour Alice !
```

Si l'utilisateur saisit Bob, le programme affiche :

```
Bonjour Bob !
```

5 Tests et conditions

Pour que notre script réagisse en fonction des différentes situations rencontrées, on a besoin d'effectuer des tests d'égalité, des comparaisons, ...

5.1 Le type booléen

Une expression est dite booléenne lorsqu'elle prend la valeur `True` ou bien la valeur `False`. On parlera aussi de variable de type booléen. Le type booléen est donc un autre type de données, que nous n'avions pas encore rencontré.

```
b = 10 # ceci est une affectation (simple egal)
print(b>8) # affiche True
print(b==5) # affiche False, le test d'egalite s'ecrit avec un double egal
print(b!=10) # affiche False (!= signifie est different de)
print(0<= b <=20) # affiche True
print(True or False) # affiche True
print(True and False) # affiche False
print(not True) # affiche False
```

5.2 Structure conditionnelle

Voici comment utiliser la structure « Si ... Alors ... Sinon ... » :

```
if condition: # si la condition est vraie
    action1   # alors executer cette instruction
else:         # sinon
    action2   # executer cette instruction
# Attention a respecter l'indentation (avec la touche tab seulement, ne pas melanger
    avec les espaces)
```

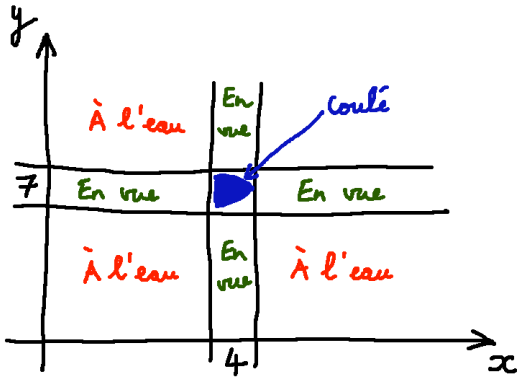
On peut si besoin utiliser la structure suivante :

```
if condition:
    action1
elif condition2: # sinon si la condition2 est verifiee
    action2
else:         # sinon
    action3   # executer cette instruction
# remarque : il est possible de mettre plusieurs fois elif
```

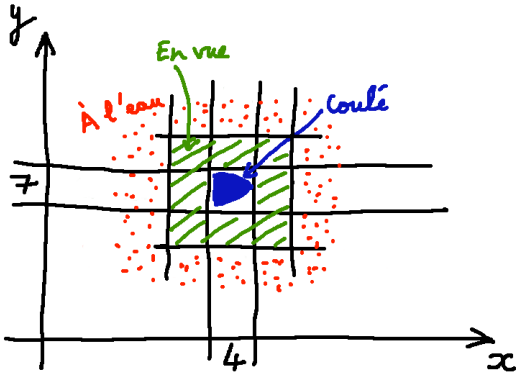
5.3 Exercice

Saisir ce script simplifié de bataille navale et le tester sur quelques exemples.

```
xBateau = 4
yBateau = 7
print("A vous de jouer")
x = int(input("Abscisse x de votre tir ?"))
y = int(input("Ordonnee y de votre tir ?"))
if x==xBateau and y==yBateau:
    print("Coule")
else:
    if x==xBateau or y==yBateau
        print("En vue")
    else:
        print("A l'eau")
```



En général, à la bataille navale, un bateau n'est « En vue » que si la case visée est immédiatement voisine de celle du bateau. Modifier le script précédent pour qu'il respecte cette nouvelle règle.



Proposer ensuite un « jeu de tests » satisfaisant pour ce script, c'est-à-dire un ensemble de tests qui permettent de s'assurer que le script fonctionne correctement dans tous les cas possibles.

6 La boucle for

6.1 Exemple concret

Exécuter le script suivant, et noter ce qui est affiché dans la console :

```
for i in range(0, 6):
    print(i)
```

6.2 Syntaxe de la boucle for

Lorsqu'on saisit

```
for i in range(entier1, entier2):
    instructions
```

On obtient le fonctionnement suivant :

- La variable `i` prend la valeur `entier1`, et on exécute `instructions`,
- puis `i` est incrémenté (c.-à-d. augmenté de 1), et on exécute `instructions`,
- puis `i` est incrémenté, et on exécute `instructions`,
- ...
- à la fin, `i` est incrémenté et vaut `entier2` mais alors on s'arrête sans exécuter `instructions`.

6.3 Exercices (réf. Dowek et coauteurs p. 33 sq.)

1. Écrire un script de 2 ou 3 lignes, qui permet d'afficher en sortie les 31 lignes suivantes :

```
1 janvier
2 janvier
3 janvier
...
28 janvier
29 janvier
30 janvier
31 janvier
```

2. Écrire un script qui recueille au clavier les températures de 7 jours successifs et calcule la température moyenne de la semaine.
3. Modifier le script précédent pour que l'utilisateur puisse préciser le nombre de jours avant de donner les températures.
4. Écrire un script qui calcule et affiche la liste des diviseurs d'un nombre entier naturel entré au clavier.
5. Combien de points affiche le script suivant ?

```
for i in range(0,10):
    print(".",end="")
for j in range(0,10):
    print(".",end="")
print()
```

6. Et celui-ci ?

```
for i in range(0,10):
    for j in range(0,10):
        print(".",end="")
print()
```

7 La boucle while

7.1 Premier exemple

Exécuter le script suivant, et noter ce qui est affiché dans la console :

```
a = 0
while a < 5 :
    a = a + 1
    print(a)
```

Tant que la condition `a < 5` est vraie, on exécute le bloc d'instruction qui a été indenté.

Vérifier ce qui a été affiché, en complétant ce tableau de l'état de la mémoire :

valeur de `a` à la ligne 2	valeur de `a` à la ligne 4

7.2 La question de la terminaison d'une boucle while

Que donnerait l'exécution du script suivant ? (Ne pas tester avec l'ordinateur, sinon ...)

```
a = 0
while a < 5 :
    print(a)
```

7.3 Deux exercices

1. Écrire un script qui demande à l'utilisateur de fournir des nombres positifs et qui calcule la moyenne de ces nombres. Le script effectue le calcul de la moyenne et affiche le résultat une fois que l'utilisateur a fourni un nombre négatif (qui n'est pas pris en compte dans le calcul).
2. Écrire un script qui affiche un tableau de valeurs pour la fonction $f : x \mapsto x^2 - 2x - 2$. L'utilisateur choisit les bornes `debut` et `fin` de l'intervalle sur lequel on calcule ces valeurs, ainsi que le pas `pas` entre deux valeurs. Comparer avec le tableau de valeurs fourni avec la calculatrice.

8 Prolongements

Voici des ressources intéressantes dans le cadre de l'ISN. (Demander conseil à un professeur pour vous guider) :

1. Apprendre à programmer avec Python 3, de Gérard Swinnen : livre disponible en format pdf (licence CC) `http://inforef.be/swi/download/apprendre_python3_5.pdf`
   - Lecture des chapitres 2, 3 et 4.
2. France IOI, site internet d'entraînement à la programmation et à l'algorithmique, `http://www.france-ioi.org/`
   - Se créer un compte, aller ensuite dans Cours et problèmes, choisir Python, puis Parcours général, et commencer à résoudre des problèmes de validation.
3. Dowek et coauteurs, Informatique et sciences du numérique, chapitres 1 (les ingrédients des programmes) et 2 (les boucles). Lien ici, – 22,78 Mio
   - Exercice 2.11 p. 38 du livre, sur la « suite de Syracuse ».
   - Exercice 2.12 p. 38 du livre, sur le PPCM.
4. Site de Fabrice Sincère `http://fsincere.free.fr/isn/python/cours_python.php`
   - Chapitre 2 – Exercice 2.1 : Validité d'un numéro de sécurité sociale.
   - Chapitre 2 – Exercice 2.4 : Indice de masse corporelle
   - Chapitre 3 – Exercice 3.1 : Tables de multiplication
   - Chapitre 3 – Exercice 3.5 : Jeu du nombre à deviner.