TP Info 1 | Variables, tests & boucles

Ce premier TP a deux objectifs : vous faire découvrir l'environnement de développement Pyzo et accompagner vos premiers pas en programmation (variables, tests et boucles) sous Python.

1. Ce qu'il faut savoir

Voici quelques rappels de syntaxe.

1.1. Types et variables

Le symbole d'affectation sous Python est =. `>>> x=3`

Opérateurs sur le type `int`

- `+`, `-` et `*` pour l'addition, la soustraction et le produit usuels.
- `n//b` et `n%b` pour le quotient et le reste dans la division euclidienne par de n par $b \neq 0$.

Type `bool` et opérateurs sur le type `bool`

- Le type booléen n'a que deux valeurs `False` et `True`.
- Les opérateurs `==` et `!=` permettent de créer les expressions booléennes d'égalité et de non-égalité de deux expressions.
- Les opérateurs `and` et `or` sont employés pour les opérations logiques *et* et *ou*.

1.2. Tests

La structure minimale ne comporte qu'un `if` (ie le `else` est optionnel). La structure la plus complète peut comporter un nombre fini de `elif`.

Structure la plus complète

```
if expression booleenne 1:
    Bloc d'instructions 1
elif expression booleenne 2:
    Bloc d'instructions 2
elif expression booleenne 3:
    Bloc d'instructions 3

etc.

elif expression booleenne N:
    Bloc d'instructions N
else:
    Bloc d'instructions N+1
```

Structure minimale (le `else` est optionnel)

```
if expression booleenne:
    Bloc d'instructions
```

```
if expression booleenne:
    Bloc d'instructions 1
else:
    Bloc d'instructions 2
```

1.3. Boucles

Boucle inconditionnelle

On l'emploie quand on sait à l'avance le nombre de fois que la boucle sera exécutée.

```
for indice in range(debut,fin,pas):
    Bloc d'instructions
```

La boucle est exécutée n fois, pour chaque valeur de `indice` dans l'intervalle d'entiers $[\![0, n-1]\!]$.

On peut également utiliser `range(m,n)` pour que `indice` décrive l'intervalle $[\![m, n-1]\!]$.

Boucle conditionnelle

On l'emploie quand on ne sait pas a priori le nombre de fois que la boucle sera exécutée.

```
while expression booleenne:
    Bloc d'instruction
```

La boucle est exécutée *tant que* `condition booleenne=True`.

1.4. La commande `input`

Interagir avec l'utilisateur dans l'interpréteur Python

▷ L'instruction `x=input("Message à l'utilisateur")` a pour effet d'afficher le `message à l'utilisateur` dans l'interpréteur Python, d'attendre que l'utilisateur entre une réponse par l'intermédiaire du clavier, d'enregistrer cette réponse dans la variable `x` (attention, sous la forme d'une chaîne de caractères) dès l'activation de la touche *Return* par l'utilisateur.

▷ Avant de traiter `x`, il faudra parfois convertir le type `str` de `x` au moyen des fonctions `int` ou `float`.

2. Exercices

Les difficultés sont échelonnées de la manière suivante : aucune, ♪, ♪♪, ♪♪♪ et ♪♪♪♪. Certains énoncés sont tirés des annales des concours (oral et écrit) ; leur provenance est le plus souvent précisée. Les exercices notés ♪♪♪ et ♪♪♪♪ sont particulièrement délicats.

2.1. Début de la séance

Allumer l'ordinateur et ouvrir *pyzo* en cliquant sur l'icône correspondante.

2.2. Variables

1 . [*Découverte des variables*]

a) Taper successivement les instructions suivantes dans l'interpréteur de Pyzo :

```
x=0; y=x; x=1
```

Après ces instructions, que valent les variables `x` et `y` ? Deviner puis vérifier.

b) Dans l'éditeur, taper les lignes suivantes :

```
x=14; y=5; z=x+y; x=y; y=z; print(x+y+z)
```

À l'exécution, que va-t-il se passer ? Prédire, puis vérifier.

c) Quelles sont les valeurs de `x` et `y` après les instructions suivantes ?

```
x=4; y=3; z=x; x=y; y=z
```

Prédire, puis vérifier.

d) Quelles sont les valeurs de `x` et `y` après les instructions suivantes ?

```
x=4; y=3; (x,y)=(y,x)
```

Prédire, puis vérifier.

e) Quelles sont les valeurs des différentes variables après les instructions suivantes ?

```
a1=1; i=1; ai=3; i=3; ai=6
```

2 . [*Échange des contenus de deux variables*]
On suppose que les variables `x` et `y` ont pour valeurs respectives les entiers `3` et `4`. On souhaite échanger le contenu de ces deux variables.

a) Proposer une méthode qui utilise une variable auxiliaire appelée `temp`.

b) On exécute la séquence d'instructions suivante : `x=x+y; y=x-y; x=x-y`. Quel est le contenu des variables `x` et `y` en fin de séquence ?

c) Même question avec `x,y=y,x`.

2.3. Tests

3 . [*Tests*]

Quelle est la valeur de la variable `y` après la suite d'instructions ci-contre ? Prédire et vérifier dans un interpréteur.

```
p=1
d=0
r=0
h=1
z=0
f=p and (d or r)
g=not r
m=not p and not z
g=g and (d or h or m)
if f or g:
    y=1
else:
    y=0
```

4 . [*Calcul de la valeur absolue*]
Écrire un programme demandant à l'utilisateur un entier relatif x et renvoyant sa valeur absolue.

5 . [*Un algorithme-mystère*]

a) Que calcule l'algorithme suivant ?

Algorithme 1 : Algorithme mystère

Données : Trois entiers relatifs a, b et c
si $a > b$ **alors**
  si $a > c$ **alors**
    res ← a
  sinon
    res ← c
sinon
  si $b > c$ **alors**
    res ← b
  sinon
    res ← c
Renvoyer res

b) Écrire un programme réalisant l'algorithme précédent, les trois entiers a, b et c étant choisis par l'utilisateur.

2.4. Boucles

a. Découverte des boucles `for` et `while`

6. [***Ma première boucle***]
Dans l'éditeur, taper puis sauver et exécuter les lignes suivantes :

```
somme=0
for i in range(1,5):
    somme=somme+i
```

Après exécution, que valent `i` et `somme` ?

7. [***La boucle mystère***]
Que vaut `f` la fin des instructions suivantes si $n = 5$? Vérifier sur machine.

```
f=0
i=1
while i<n+1:
    f=f+i
    i=i+1
```

8. [***Programme à corriger***]
Deviner ce que calcule puis corriger le programme suivant :

```
n=int(input("Entrez la valeur de n : "))
for i in range(n):
    if i%2=0:
        carre=i**2
    else:
        carre=0
    total=total+carre
```

9. [***Un premier calcul de puissance***]
Calculer 3^{841} en appliquant l'algorithme basique suivant :

Algorithme 2 : Calcul de 3^{841}

Initialisation : res $\leftarrow$ 1;
pour $1 \leqslant i \leqslant 841$ **faire**
  res $\leftarrow$ res $\times$ 3
Renvoyer Res

Comparer avec le résultat de `3**841`.

10. [***Une factorielle***]
Calculer 100! en appliquant l'algorithme suivant :

Algorithme 3 : Calcul de 100!

Initialisation : res $\leftarrow$ 1;
pour $2 \leqslant i \leqslant 100$ **faire**
  res $\leftarrow$ res $\times$ i
Renvoyer Res

Comparer avec le résultat fourni par la fonction `factorial` de la bibliothèque `math` :

```
>>> import math
>>> math.factorial(...)
```

11 . [*Exponentiation modulaire*]
On note a mod b le reste dans la division euclidienne de a par b.

a) Sans calculatrice : quelle est la dernière décimale de 17×923 ?

b) Quelle est la dernière décimale de $123345678987654 \times 836548971236$?

c) Prouver que, pour connaître $ab \mod 10$, on fait le produit p de $a \mod 10$ par $b \mod 10$, et on calcule $p \mod 10$.

d) Montrer que ce résultat reste valable modulo n'importe quel entier.

e) On considère l'algorithme suivant :

Algorithme 4 : Calcul de $a^b \mod c$

Initialisation : res $\leftarrow 1$;
pour $1 \leqslant i \leqslant b$ **faire**
  res $\leftarrow$ (res $\times$ a) mod c
Renvoyer Res

i) Expliquer l'intérêt de cette façon de procéder par rapport à la version naïve *on calcule ab, puis on réduit le résultat modulo c*.

ii) Calculer ainsi $123456^{654321} \mod 1234567$.

b. Quel type de boucle choisir ?

12 . [*Un premier calcul*]
Calculer

$$\sum_{k=10}^{100} k^2$$

13 . [*Divergence vers* $+\infty$ *de la somme des carrés d'entiers*]
Écrire un programme qui demande à l'utilisateur d'entrer un entier naturel M et renvoie le plus petit entier n tel que $1^2 + 2^2 + \cdots + n^2 \geqslant$ M.

14 . [*Sum of all the multiples of 3 or 5 below 1000, Euler project n^o 1*]
If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23. Find the sum of all the multiples of 3 or 5 below 1000.

15 . [*Le jeu du nombre secret*]
Dans le jeu du nombre secret, l'ordinateur commence par tirer au sort un entier entre 0 et 99 (le nombre « *secret* »). L'utilisateur doit deviner ce nombre en un minimum d'essais. À chaque fois que l'utilisateur propose un nombre, l'ordinateur lui répond « *trop petit* » ou « *trop grand* » de façon à le guider. Écrire un programme permettant de jouer au jeu du nombre secret. On pourra utiliser la fonction `random.randint(a,b)` (du module `random`) génère un nombre entier pseudo-aléatoire dans $[a, b[$.

2.5. Pour aller plus loin

On ne traitera ces exercices qu'à condition d'avoir réussi tous les exercices précédants.

16 . [*Conversion*]
Écrire un programme demandant un nombre entier `n` de secondes à l'utilisateur et renvoyant sa conversion en un quadruplet `(jours,heures,minutes,secondes)`.

17 . [*Puissances itérées de Knuth*]
Écrire un programme calculant

$$x \uparrow\uparrow n = x^{x^{\cdot^{\cdot^{x}}}} \text{ où } x \text{ apparaît } n \text{ fois}$$

pour un nombre entier $x \geqslant 1$ et un nombre entier $n \geqslant 1$ choisis par l'utilisateur. On respectera la convention suivante : les puissances se calculent *du haut vers le bas*. Par exemple,

$$2^{2^{2^2}} = 2^{2^4} = 2^{16} = 65536$$

La notation $x \uparrow\uparrow n$ est due à Donald Knuth.

18 . [*Nombre de diviseurs d'un entier naturel non nul*]
Écrire un programme demandant à l'utilisateur un nombre entier naturel n non nul et renvoyant le nombre de diviseurs dans $\mathbb{N}^*$ de n.

19 . [*Somme des décimales*]
Écrire une fonction prenant en entrée un entier n et renvoyant la somme des décimales de n.

20 . [*Even Fibonacci numbers, Euler project n^o 2*]
Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be :

$$1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

21 . [*1000-digits Fibonacci numbers, Euler Project n^o 25*]
The Fibonacci sequence is defined by the recurrence relation :

$$F_n = F_{n-1} + F_{n-2}, \text{ where } F_1 = 1 \text{ and } F_2 = 1$$

Hence the first 12 terms will be : $F_1 = 1$, $F_2 = 1$, $F_3 = 2$, $\ldots$, $F_{10} = 55$, $F_{11} = 89$, $F_{12} = 144$. The 12-th term, F_{12}, is the first term to contain three digits. What is the first term in the Fibonacci sequence to contain exactly 1000 digits ?

22 . [*Sum of digits, Euler project n^o 16*]
Hence $2^{15} = 32768$, the sum of its digits is equal to $3+2+7+6+8 = 26$. What is the sum of the digits of the number 2^{1000} ?

23 . [*Self Powers*]
Find the last ten digits of $S = 1^1 + 2^2 + 3^3 + \cdots + 1000^{1000}$.

24 . [*Test de primalité*]

a) Écrire un programme demandant à l'utilisateur un entier naturel `n` et renvoyant s'il est premier ou non sous la forme d'un booléen.

b) Afficher les 20 premiers nombres premiers.

25 . [*The 10001th Prime, Euler project n^o 6*]
By listing the first six prime numbers : 2, 3, 5, 7, 11, and 13, we can see that the 6th prime is 13. What is the 10 001st prime number ?