

Algorithmique et Programmation 1

TD2 : Types de base, Expressions, Variables, Répétitions

1 Le calcul de l'impôt

On cherche à faire une application qui calcule le montant de l'impôt sur le revenu en fonction du montant du revenu de l'utilisateur et du nombre de parts fiscaux de son foyer¹. Cette application est à destination d'utilisateurs imposables mais modestes (première tranche). Ce programme commence par :

```
revenu_imposable = float(input("Quel est le montant de votre revenu imposable ? "))
parts = float(input("Quel est le nombre de parts fiscales de votre foyer ? "))
```

La variable `revenu_imposable` est une variable de type `float` représentant le revenu imposable saisi par l'utilisateur et la variable `parts` est aussi un flottant représentant le nombre de parts de son foyer. Dans cet exercice, vous devrez définir des variables en fonction de celles renseignées par l'utilisateur en utilisant seulement les opérateurs arithmétiques (+, -, *, /, //, %) et l'opérateur d'affectation =.

1. Le *revenu net imposable* est obtenu en diminuant de 10% le montant du revenu imposable. Par exemple si le revenu imposable est de 10000 €, alors le *revenu net imposable* sera de 9000 €. Définir une variable représentant cette grandeur.
2. Le montant de l'impôt est égal à 14% du *revenu net imposable* moins 1372.98 fois le *nombre de parts*. Définir une variable représentant le montant des impôts de l'utilisateur ;
3. Écrire une instruction qui affiche le montant des impôts intégré dans un message du type :
Le montant de vos impots est de (---) euros

2 Les boucles for

Réaliser les tâches suivantes à l'aide d'une boucle `for`.

1. Soient deux instructions quelconques, par exemple :

```
print("L'amour est enfant de bohème")
print("Qui n'a jamais jamais connu de loi")
```

Écrire un programme qui exécute ces instructions 101 fois.

2. Écrire un programme qui demande à l'utilisateur d'entrer un nombre entier positif N et qui affiche tous les entiers compris entre 0 et N (un entier sur chaque ligne) ;
3. Écrire un programme qui demande à l'utilisateur d'entrer un nombre entier positif N et qui affiche tous les éléments de la suite $u_n = 2n + 3$ pour n compris entre 0 et N ;
4. Soit le programme suivant :

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

1. <https://www.economie.gouv.fr/particuliers/tranches-imposition-impot-revenu>

Quelle somme calcule le programme précédent ? Quelle est sa valeur ?

5. Écrire un programme qui demande à l'utilisateur d'entrer un nombre entier positif N et qui affiche la somme des entiers *pairs* compris entre 0 et $2N$ (on compte aussi $2N$ lui-même).
6. Écrire un programme qui demande à l'utilisateur d'entrer un nombre entier positif N et qui affiche le factoriel de N . Je rappelle que le factoriel de N (qui se note $N!$) est égal au produit de tous les entiers compris entre 1 et N .

3 Jeter de dés

Le programme suivant :

```
import random
for i in range(10):
    print(random.random())
```

affiche dix valeurs flottantes choisies aléatoirement dans l'intervalle $[0, 1[$. Pour réaliser les tâches suivantes, aucune fonction supplémentaire ne sera nécessaire. Vous pourrez simplement utiliser les opérateurs arithmétiques.

1. Modifier le programme précédent pour qu'il affiche dix valeurs (entières ou non) choisies aléatoirement dans l'intervalle $[0, 6[$;
2. Modifier le programme précédent pour qu'il affiche dix valeurs entières choisies aléatoirement entre 0 et 5 ;
3. Modifier le programme précédent pour qu'il affiche dix valeurs entières choisies aléatoirement entre 1 et 6.

4 Pour aller plus loin : la masse d'un écrou

On cherche à calculer la masse d'un écrou. De cet objet, on connaît le rayon extérieur R , le rayon r du trou, la hauteur h , et la masse volumique d (cf figure 1).

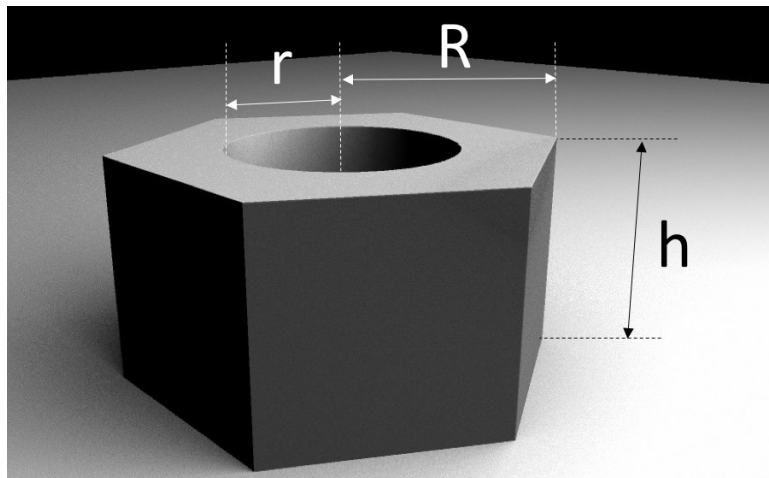


FIGURE 1 – On cherche à calculer la masse d'un écrou de hauteur h , dont le rayon total est R et dont le rayon du trou est r . La masse volumique est de d .

On sait aussi que :

- la masse de l'écrou est égal au volume de l'écrou multiplié par sa masse volumique d ;
- le volume de l'écrou est le volume d'un prisme hexagonal (de côté R et de hauteur h) moins le volume du trou de rayon r ;

- le volume du trou cylindrique est égal à la surface d'un disque de rayon r multipliée par la hauteur h ;
- le volume du prisme hexagonal est la surface de la section hexagonale multipliée par la hauteur h ;
- la surface d'un hexagone régulier de rayon R est égal à 6 fois la surface d'un triangle équilatéral de côté R ;
- la surface d'un triangle équilatéral de côté R est de $\frac{R^2\sqrt{3}}{4}$.

On cherche à faire un programme *python* où l'utilisateur entrerait la valeur de R , r , h et d et qui calculerait la masse de l'écrou. Le programme commence donc par les lignes suivantes :

```
R = float(input("Entrer le rayon extérieur de l'écrou : "))
r = float(input("Entrer le rayon du trou : "))
h = float(input("Entrer la hauteur de l'écrou : "))
d = float(input("Entrer la masse volumique : "))
```

Vous êtes peut-être capables de calculer la masse de l'écrou avec une seule instruction sans autres variable que celles définies ci-dessus. Mais il est bien plus facile d'écrire une variable par propriété. Le programme résultant ne sera pas moins performant et sera beaucoup plus lisible.

1. Définir une variable représentant la surface d'un triangle équilatéral de côté R (en python, $\sqrt{3}$ s'écrit : `math.sqrt(3)`);
2. Définir une variable représentant la surface d'un hexagone régulier de côté R ;
3. Définir une variable représentant le volume d'un prisme hexagonal de côté R et de hauteur h ;
4. Définir une variable représentant le volume d'un cylindre de rayon r et de hauteur h ;
5. Définir une variable représentant le volume de l'écrou ;
6. Définir une variable représentant la masse de l'écrou ;
7. Afficher cette masse.

5 Des variables qui permettent d'accélérer les calculs

On cherche à calculer la série :

$$f(x) = \sum_{i=0}^N ix^i = x + 2x^2 + 3x^3 + \dots + Nx^N \quad (1)$$

Voici une proposition de programme pour calculer la valeur de $f(x)$ pour $x = 10$ et $N = 5$:

```
x = 10
somme = x + 2*x**2 + 3*x**3 + 4*x**4 + 5*x**5
print(somme)
```

Pour $N = 5$, le temps nécessaire pour réaliser ce calcul est négligeable, mais pour des N plus importants, ce temps peut s'avérer bien plus important. Voici quelques éléments vous permettant de proposer un programme bien plus performant :

- L'opérateur puissance (`**`) est plus de 100 fois plus coûteux (en temps de calcul) que l'opérateur de multiplication (`*`).
- Dans l'algorithme ci-dessus, pour chaque terme, nous calculons x^i alors que, pour le terme précédent, nous avons déjà fait presque tout le travail en calculant x^{i-1} . Les variables nous permettent de garder en mémoire la valeur de x^{i-1} et de l'utiliser pour calculer x^i par une simple multiplication.

1. Proposer un programme plus rapide pour $N = 5$;
2. Proposer un programme plus rapide pour N quelconque (avec une boucle `for`).