

# TP N°0-1 : Introduction à Python

UP Mathématiques\*

2 novembre 2023



## Exercice 1: Les variables

- a) Assignez les valeurs respectives 3, 5, 7 à trois variables a, b, c. Effectuez l'opération  $a-b//c$ . Interprétez le résultat obtenu.
- b) Testez les lignes d'instructions suivantes. Décrivez ce qui se passe :

```
r = 12
pi = 3.14159
s = pi * r**2
print(s)
print(type(r), type(pi), type(s))
```

Quelle est, à votre avis, l'utilité de la fonction `type()` ?

- c) Écrivez un programme qui convertisse en degrés Celsius une température exprimée au départ en degrés Fahrenheit, ou l'inverse. La formule de conversion est :

$$T_F = T_C \times 1,8 + 32$$

## Exercice 2: Fonction `input()`

Dans tous ces exercices, utilisez la fonction `input()` pour l'entrée des données.

- a) Écrivez un programme qui convertisse en mètres par seconde et en km/h une vitesse fournie par l'utilisateur en miles/heure. (Rappel : 1 mile = 1609 mètres)

---

\*École Supérieure PRivée d'Ingénierie et de Technologies (ESPRIT).

b) Écrivez un programme qui calcule le périmètre et l'aire d'un triangle quelconque dont l'utilisateur fournit les 3 côtés. (Rappel : l'aire d'un triangle quelconque se calcule à l'aide de la formule :

$$S = \sqrt{d \cdot (d - a) \cdot (d - b) \cdot (d - c)}$$

dans laquelle  $d$  désigne la longueur du demi-périmètre, et  $a$ ,  $b$ ,  $c$  celles des trois côtés.)

### Exercice 3: Corriger l'erreur dans le code

a) Le code suivant renvoie une erreur. Trouver et corriger l'erreur:

```
prenom = input('Entrez votre prénom : ')
age = input('Entrez votre age : ')
annee_naissance = 2018 - age
print("Bonjour Mr/Mme", prenom)
print("vous êtes né en", annee_naissance)
```

#### Indication.

- utiliser la fonction `type()`
- utiliser la fonction `int()`

b) Afficher le message suivant: Bonjour Mr/Mme `prenom`, votre age est `age` et vous êtes né en `annee_naissance`.

**Indication.** Remplacer les points par ce qui convient dans le code:

```
print(" Bonjour Mr/Mme {}, votre age est {}....".format(...))
```

### Exercice 4: Polynômes creux

L'objectif de cet exercice est la manipulation des polynômes creux à une seule variable.

Un polynôme creux est un polynôme dont certains coefficients sont nuls.

Un polynôme est construit à partir de monômes.

Un monôme est une expression de la forme  $ax^n$  où  $a$  ( $a \neq 0$ ) est le coefficient du monôme et  $n$  ( $n \geq 0$ ) son degré.

Un monôme est représenté par un dictionnaire à un élément dont la clé est le degré  $n$  et la valeur est le coefficient  $a$ .

#### Exemple :

Le monôme  $8x^2$  est représenté par le dictionnaire `{2:8}`.

Un polynôme creux est alors défini comme une association de monômes de degrés différents.

**Exemple :**

Le polynôme  $-x^4 + 8x^2 - 5x$  est représenté par le dictionnaire  $\{2:8, 1:-5, 4:-1\}$ .

Le dictionnaire  $\{0:1, 5:1, 8:1\}$  représente le polynôme  $x^8 + x^5 + 1$ .

a) Compléter le script de la fonction `ajout_monome(P,monome)`, sachant que `P` est un dictionnaire représentant un polynôme. On rappelle que cette méthode ajoute un monôme saisi au clavier (en faisant les contrôles nécessaires) si le paramètre `monome` est nul ou ajoute le monôme nommé `monome` sinon.

```
def ajout_monome(P,monome={}):
    """
    Cette méthode ajoute un monôme saisi au clavier à P
    si le paramètre monome est nul ou ajoute le monôme nommé monome sinon
    """
    if len(monome)==0:
        # Partie à compléter
    else:
        degre=list(monome.keys())[0]
        coeff=list(monome.values())[0]
        assert degre>=0
        assert type(degre)==int
        assert type(coeff)==int or type(coeff)==float
        assert len(monome)==1
        P.update(monome)
```

**Indication.** L'appel à la fonction `ajout_monome` peut être utilisé comme suivant :

```
In[1]: P = {0:1, 5:1, 8:1} # x^8 + x^5 + 1
In[2]: ajout_monome(P)
donner le degré du monôme : 2
donner le coefficient du monôme : 1
Out[2]: {0: 1, 5: 1, 8: 1, 2: 1.0}
```

ou bien

```
In[3]: ajout_monome(P, {2:8})
Out[3]: {0: 1, 5: 1, 8: 1, 2: 8}
```

b) Écrire le code de la fonction `degree(P)` qui prend en paramètre le polynôme `P` représenté sous forme d'un dictionnaire et renvoie son degré.

```
def degree(P):
    # à compléter
```

- c) Écrire le code de la fonction `call(P,x0)` qui retourne la valeur du polynôme `P` passé en paramètre sous forme d'un dictionnaire pour un réel `x0` donné.

```
def call(P, x0):  
    # à compléter
```

- d) Écrire le code de la fonction `add(P,P1)` qui retourne le polynôme somme des deux polynômes `P` et `P1`.

```
def add(P, P1):  
    # à compléter
```

**Indication.** aucun monôme nul ne doit apparaître dans le polynôme résultat.

- e) En déduire une fonction `diff(P,P1)` qui retourne le polynôme différence des deux polynômes `P` et `P1`. **Remarque :** aucun monôme nul ne doit apparaître dans le polynôme résultat.

- f) Écrire le code de la fonction `mul(P,P1)` qui retourne le polynôme produit de deux polynômes. **Remarque :** aucun monôme nul ne doit apparaître dans le polynôme résultat.

- g) Écrire le code de la fonction `derive(P)` qui retourne le polynôme dérivé de `P`.

- h) Écrire le code de la fonction `affiche(P)` qui retourne la chaîne de caractères représentant l'expression du polynôme ordonné par ordre décroissant.

Pour le polynôme représenté par `{4:4, 0:4, 12:6, 9:1, 7:-1}`, la chaîne retournée est : `"6 * X^12 + X^9 - X^7 + 4 * X^4 + 4"`

- i) Écrire le code de la fonction `primitive(P)`, qui retourne le polynôme représentant la primitive. On suppose que la constante d'intégration est nulle.

- j) On définit, l'intégrale d'un polynôme creux `P` en `x` entre les bornes `a` et `b`, par :

$$S = \int_a^b P dx$$

Écrire le script de la fonction, nommée `integrale`, permettant de retourner la valeur de `S` à partir d'un polynôme `P`, de type `PolynomeCreux`, et des bornes d'intégration `a` et `b` réels.

- k) Écrire une fonction `courbe()` affichant la dans un même repère un polynôme `P`, sa primitive et sa dérivée.