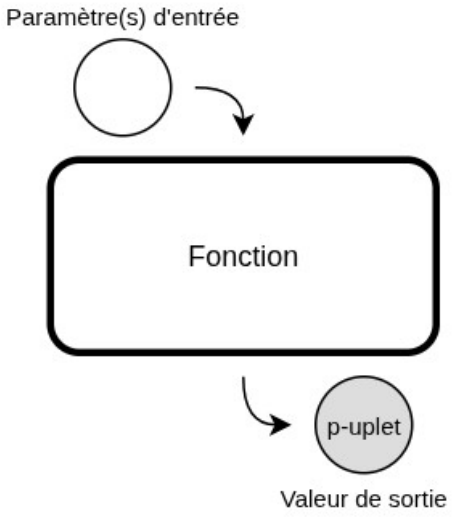
Nom :	Prénom :	Classe :

NSI 1re — Fonction : retourner un p-uplet de valeurs

Objectifs :
- Comprendre l'intérêt d'une fonction retournant un p-uplet de valeurs.
- Savoir créer une fonction retournant un p-uplet de valeurs.

Introduction

Une **fonction** qui renvoie un **p-uplet**[1] signifie que sa **valeur de sortie**[2] contient plusieurs informations, regroupées en un unique p-uplet (tuple, en Python).



Exemple

Ainsi, on pourrait imaginer par exemple une fonction `extremums` qui renverrait la valeur la plus basse ET la valeur la plus haute, pour un tableau donné :

```
1  def extremums(tableau):
2      v_min = tableau[0]
3      v_max = tableau[0]
4      for v in tableau:
5          if v < v_min:
6              v_min = v
7          if v > v_max:
8              v_max = v
9      return (v_min, v_max)
```

1 Voir le cours « p-uplet (tuple) ».
2 Voir le cours « Fonction : prototypage, création et utilisation ».

➤ **Exercice 1** — Prédire

En vous basant sur la définition de la fonction `extremums` page précédente, à votre avis, que renverrait les appels suivants ?

N°	Appel	Valeur retournée par la fonction (à compléter)
1	`extremums([23, 2, 8, 11])`	Cette fonction renvoie le p-uplet (2, 23)
2	`extremums([0, 18, 4, -2])`	Cette fonction renvoie le p-uplet (-2, 18)
3	`extremums([13, 2, 7, 14])`	Cette fonction renvoie le p-uplet (2, 14)

➤ **Exercice 2** — Exécuter

Exécuter le code de ce bac à sable Python, pour vérifier votre prédiction :

Lien

➤ **Exercice 3** — Expliquer

Lire le programme suivant puis répondre aux questions posées.

```
1  from random import randint
2  def des_6faces(j):
3      r = []
4      for _ in range(j):
5          r.append(randint(1, 6))
6      return tuple(r)
```

a) Expliquer le déroulement de la fonction `des_6faces` :

Cette fonction prend un paramètre « j » qui détermine le nombre d'itérations d'une boucle bornée (ligne 4). À chaque itération, on ajoute un nouvel élément dans un tableau indexé « r », dont la valeur varie aléatoirement entre « 1 » et « 6 ». À la fin, la fonction renvoie le tableau indexé « r » (converti en tuple) constitué de « j » éléments.

b) Qu'est-ce que la fonction renvoie lorsqu'on lui donne `3` comme paramètre ?

Dans ce cas, la fonction renvoie un p-uplet constitué de 3 éléments.

➤ **Exercice 4** — Code puzzle

Elena avait codé une fonction `infos_eleve` qui lui permettait de récupérer la note la plus basse, la note la plus haute, ainsi que la moyenne d'un élève. Mais son animal de compagnie a marché sur son clavier et a mélangé les lignes de code ... *Si ça peut aider*, elle se souvient que sa fonction renvoyait `(3, 16, 8.2)` quand on lui donnait `[4, 8, 10, 3, 16]`.

Code mélangé :

```
def infos_eleve(notes):
    return (note_min, note_max, moyenne)
        if v < note_min:
    compteur = 0
    moyenne = accumulateur / compteur
    note_min = notes[0]
            note_min = v
        if v > note_max:
        compteur = compteur + 1
    note_max = notes[0]
            note_max = v
        accumulateur = accumulateur + v
    for v in notes:
    accumulateur = 0
```

Code réparé (à compléter) :

```
def infos_eleve(notes):
    note_min = notes[0]     # On prend par défaut la valeur du premier élément
    note_max = notes[0]     # Idem
    compteur = 0            # On va avoir besoin de calculer une moyenne...
    accumulateur = 0        # donc il faut un compteur et un accumulateur
    for v in notes:
        if v < note_min:
            note_min = v
        if v > note_max:
            note_max = v
        compteur = compteur + 1
        accumulateur = accumulateur + v
    moyenne = accumulateur / compteur    # On calcule la moyenne
    return (note_min, note_max, moyenne) # On renvoie un tuple


# On peut tester cette fonction
mia = infos_eleve([4, 8, 10, 3, 16])
amir = infos_eleve([18, 19, 14, 13, 19])

print(mia)  # Affiche (3, 16, 8.2)
print(amir) # Affiche (13, 19, 16.6)
```

➤ **Exercice 5** — Créer un programme d'après un cas d'usage

Créer une fonction `conversion_heures_minutes` qui prend `m` (minutes) comme paramètre, et qui renvoie un p-uplet contenant le nombre d'heures et de minutes correspondant à `m`.

Par exemple, appeler cette fonction avec un paramètre valant `90` renverrait `(1, 30)`. Et appeler cette fonction avec un paramètre valant `120` renverrait `(2, 0)`.

```
def conversion_heures_minutes(m):
    h = m // 60
    m_restantes = m % 60
    return (h, m_restantes)

print( conversion_heures_minutes(90) )   # (1, 30)
print( conversion_heures_minutes(120) )  # (2, 0)
```

➤ **Exercice 6** — Créer un programme d'après un cas d'usage

Créer une fonction `conversion_heures_minutes_secondes` qui prend `s` (secondes) comme paramètre, et qui renvoie un p-uplet contenant le nombre d'heures, de minutes et de secondes correspondant à `s`.

Par exemple, appeler cette fonction avec un paramètre valant `7855` renverrait `(2, 10, 55)`. Et appeler cette fonction avec un paramètre valant `2543` renverrait `(0, 42, 23)`.

```
def conversion_heures_minutes_secondes(s):
    h = s // 3600
    s_restantes = s % 3600
    m = s_restantes // 60
    s = s_restantes % 60
    return (h, m, s)

print( conversion_heures_minutes_secondes(7855) ) # (2, 10, 55)
print( conversion_heures_minutes_secondes(2543) ) # (0, 42, 23)
```