



# Runtrack Python

*Python is powerful... and fast; and open; and ... many other things.*

## Job 01

---

Écrire une fonction qui retourne une liste nommée « **fruits** » qui contient les string suivantes : « **pomme** », « **cerise** », « **orange** ».  
Afficher en console le résultat de la fonction.

Résultat attendu :

```
['pomme', 'cerise', 'orange']
```

## Job 02

---

Écrire une fonction qui contient une liste nommée « **fruits** » qui contient les string « **pomme** », « **cerise** », « **orange** ».  
Affichez le 2ème élément de la liste.

Résultat attendu : `cerise`

## Job 03

---

Écrire une fonction qui contient une liste nommée « **fruits** » qui contient les strings « **pomme** », « **cerise** », « **orange** ». Cette fonction doit à son appel ajouter à la liste « **fruits** » une String « **Melon** » à la fin de cette liste.



Résultat attendu :

```
['pomme', 'cerise', 'orange', 'Melon']
```

## Job 04

---

Écrire une fonction qui contient une liste nommée « **fruits** » qui contient les strings **pomme, cerise, orange, melon**. Cette fonction doit à son appel ajouter à la liste « **fruits** » une string « **mangue** » à l'**index 2**.

Résultat attendu : **['pomme', 'cerise', 'mangue', 'orange', 'melon']**

## Job 05

---

Écrire un programme qui crée une liste nommée « **L** » d'au moins **5 entiers** puis successivement :

- Afficher la **deuxième valeur** de la liste
- Écrire une fonction qui remplace **L[3]** par la somme des cases voisines **L[2] & L[4]**, puis afficher à nouveau le tableau.
- Puis afficher la dernière valeur de la liste.

Résultat attendu :

```
[1, 2, 3, 4, 5]
2
[1, 2, 3, 8, 5]
5
```



## Job 06

---

Écrire un programme qui échange les valeurs de la **première** et de la **dernière** case d'une liste quelconque non vide.

Résultat attendu :

```
[1, 2, 3, 4, 5]
[5, 2, 3, 4, 1]
```

## Job 07

---

Écrire un programme qui **compte le nombre de multiples de 3** présents dans la liste : **L = [8, 24, 48, 2, 16]**.

## Job 08

---

Écrire un programme qui **calcule la somme de toutes les valeurs paires** de la liste : **L = [8, 24, 27, 48, 2, 16, 9, 7, 84, 91]**.

## Job 09

---

Écrire un programme qui **récupère** la valeur, **maximum** et le **minimum** des éléments de la liste : **L = [8, 24, 27, 48, 2, 16, 9, 102, 7, 84, 91]**.

Résultat attendu :

```
la valeur max est : 102
la valeur min est : 2
```



## Job 10

---

Écrire un programme qui **calcule le produit de toutes les valeurs de la liste** comprises dans l'intervalle **[25, 90]**.

**L = [8, 24, 27, 48, 2, 16, 9, 102, 7, 84, 91]**

## Job 11

---

Écrire un programme qui crée la liste d'entiers **L = [7, 11, 42, 39, 2]**, votre programme devra pouvoir modifier la liste en augmentant de 1 la valeur de chaque élément de la liste.

## Job 12

---

Écrire un programme qui trie dans l'ordre croissant une liste passée en paramètre.

**SANS UTILISER DE FONCTION SYSTÈME (len, sort, round.....)**

## Job 13

---

Écrivez un programme qui **supprime les doublons** de la liste suivante : [10, 20, 30, 20, 10, 50, 60, 40, 80, 50, 40].

**SANS UTILISER DE FONCTION SYSTÈME (len, sort, round.....)**



## Job 14

---

Créer un programme contenant une fonction nommée « **my\_long\_word** ». Cette fonction doit prendre deux paramètres, un chiffre entier et une chaîne de caractère.

Cette fonction doit retourner les mots plus longs que le chiffre passé en paramètre.

**Exemple :**

**my\_long\_word**(3, « La peur est le chemin vers le côté obscur, la peur mène à la colère, la colère mène à la haine, la haine mène à la souffrance »)

**Output :** « peur chemin vers côté obscur peur mène colère colère mène haine haine mène souffrance »

**SANS UTILISER DE FONCTION SYSTÈME (len, sort, round.....)**

## Job 15

---

Écrivez un programme qui **arrondi les nombres** de la liste : **[22.4, 4.0, 16.22, 9.10, 11.00, 12.22, 14.20, 5.20, 17.50]**. Puis **retourner** la liste dans l'ordre croissant.

**SANS UTILISER DE FONCTION SYSTÈME (len, sort, round.....)**

## Compétences visées

---

- Installer un environnement de développement python
- Maîtriser les bases de python
- Implémenter un algorithme



## Rendu

---

Créer sur github un répertoire nommé "runtrack-python". Créer dans ce répertoire un dossier « **jour04** », et pour chaque étape, un dossier « **jobXX** » où **XX** est le numéro de l'étape.

## Base de connaissances

---

- [Les listes en Python](#)
- [Documentation officielle : La structure de données](#)