

Fiche de TD numéro 1**Algorithmique, Listes & Dictionnaires**

Exercice 1 : Spécifier puis écrire une fonction `decomp_facteurs_premiers` qui, pour un entier n , retourne la liste des entiers qui forment sa décomposition en facteurs premiers.

```
>>> decomp_facteurs_premiers(5)
[5]
>>> decomp_facteurs_premiers(40) # 40 = 2 x 2 x 2 x 5
[2, 2, 2, 5]
>>> decomp_facteurs_premiers(1)
[]
```

Indice : 1 n'est pas un nombre premier et n'a pas non plus de décomposition en facteurs premiers.

Pour décomposer 40, on peut procéder ainsi :

- 40 est divisible par 2. 40 divisé par 2 donne 20.
- 20 est divisible par 2. 20 divisé par 2 donne 10.
- 10 est divisible par 2. 10 divisé par 2 donne 5.
- 5 n'est pas divisible par 3.
- 5 n'est pas divisible par 4.
- 5 est divisible par 5. 5 divisé par 5 donne 1.

Exercice 2 : L'algorithme d'Héron d'Alexandrie permet de calculer une approximation de la racine carrée d'un entier. Ainsi, si on cherche la racine carrée de l'entier (positif) a , on obtiendra une approximation de plus en plus précise sous la forme d'une fraction en calculant les termes des suites n et d qui représentent le numérateur et le dénominateur de cette fraction :

$$\begin{aligned} n_0 &= 1 \\ d_0 &= 1 \\ \forall i > 0 \quad n_{i+1} &= n_i^2 + ad_i^2 \\ \forall i > 0 \quad d_{i+1} &= 2n_id_i \end{aligned}$$

Spécifier puis écrire une fonction qui prend en argument deux entiers positifs, a et k , et qui retourne un arrondi de la racine carrée de a en calculant $\frac{n_k}{d_k}$.

```
>>> racineCarree(25, 4)
5.000023178253949
>>> racineCarree(5, 4)
2.2360679774999781
```

Gestion d'une base de films

Vous devez écrire un programme qui gère une petite base de données en mémoire. La base contient les informations concernant les films à l'affiche dans un cinéma. La base est implémentée sous la forme d'une liste de dictionnaires où chaque film correspond à un dictionnaire dans la liste. Chaque dictionnaire contient les clés `titre`, `genre` et `duree`. Exemple d'une base :

```
base = [
{'titre': '007 Spectre', 'genre': 'Aventure', 'duree': 148},
{'titre': 'Le pont des espions', 'genre': 'thriller', 'duree': 141},
```

```
{'titre': 'Hunger Games', 'genre': 'fantastique', 'duree': 136},
{'titre': 'Avatar', 'genre': 'fantastique', 'duree': 162},
{'titre': 'Jurassic world 2', 'genre': 'Aventure', 'duree': 122},
]
```

Exercice 3 : Spécifiez puis écrivez la fonction `insere` qui reçoit en paramètre la base, ainsi que le titre, le genre et la durée du film. Ces informations doivent être insérées dans la base passée en argument.

```
>>> insere(base, 'Swallow', 'thriller', 115)
>>> base
base = [
{'titre': '007 Spectre', 'genre': 'Aventure', 'duree': 148},
{'titre': 'Le pont des espions', 'genre': 'thriller', 'duree': 141},
{'titre': 'Hunger Games', 'genre': 'fantastique', 'duree': 136},
{'titre': 'Avatar', 'genre': 'fantastique', 'duree': 162},
{'titre': 'Jurassic world 2', 'genre': 'Aventure', 'duree': 122},
{'titre': 'Swallow', 'genre': 'thriller', 'duree': 115}]
```

Exercice 4 : Spécifiez puis écrivez la fonction `films_par_genre` qui renvoie la liste des titres des films d'un genre donnée. Cette fonction prend en paramètres le genre recherché et la base des films.

```
>>> films_par_genre('fantastique', base)
['Hunger Games', 'Avatar']
```

Exercice 5 : Spécifiez puis écrivez la fonction `base_genres` qui prend une base de films et renvoie un dictionnaire qui associe à chaque genre la liste des films de genre.

```
>>> base_genres(base)
{'fantastique': ['Hunger Games', 'Avatar'],
 'Aventure': ['007 Spectre', 'Jurassic world 2'],
 'thriller': ['Le pont des espions', 'Swallow']}
```

Exercice 6 : Spécifiez puis écrivez la fonction `film_plus_long` qui renvoie le titre du film qui a la plus longue durée. Nous supposons que les durées des films de la base sont toutes différentes.

```
>>> film_plus_long(base)
'Avatar'
```

Exercice 7 : Spécifiez puis écrivez la fonction `presents` qui renvoie la liste de titres de films présents dans la base parmi une liste de films.

```
>>> presents(['Avatar', 'saw', 'Greenland', 'Swallow'], base)
['Avatar', 'Swallow']
```

Exercice 8 : Spécifiez puis écrivez la fonction `meilleur_film` qui reçoit une listes de tuples qui associe à chaque film le nombre d'entrées ainsi que la base de films. La fonction doit renvoyer le film qui a le meilleur score de vues de la base.

```
>>>> l = [('Avatar', 3400000), ('007 Spectre', 2450000), ('Saw', 1710000),
('Hunger Games', 3100000), ('Jurassic world 2', 5100000), ('Swallow', 2730000)]
>>> meilleur_film(l, base)
'Jurassic world 2'
```