

**Exercice 1 BPCRST**

Durant l'épidémie de Covid-19 en France, le nombre de cas positifs était remonté quotidiennement aux autorités pour en suivre l'évolution. Récupérez sur cahier de prépa la liste des nombres de cas quotidiens déclarés durant les premiers mois de l'épidémie (fichier "données-TP11.py").

Q1 Tracer le graphe du nombre de cas en fonction du temps.

Le graphe précédent présente de nombreux "pics" liés aux weekends, pendant lesquels les données sont moins remontées. Pour lisser ces irrégularités, on représente souvent la moyenne du nombre de cas sur une semaine glissante. Cela signifie qu'en face du jour k on représente non pas le nombre x_k de cas déclarés ce jour, mais la moyenne des nombres de cas déclarés lors de la semaine précédant le jour k .

Q2 Écrire une fonction Python prenant en argument une liste de nombres et renvoyant la liste contenant les moyennes glissantes de ces nombres sur 7 jours.

Q3 Tracer sur le même graphique le nombre de cas de Covid par jour, et le nombre moyen sur une semaine glissante.

Exercice 2 Mastermind

Le but de l'exercice est d'écrire un code simulant le jeu du Mastermind. Dans ce jeu, un joueur doit deviner par des essais successifs une combinaison secrète choisie par l'ordinateur. Pour chaque combinaison proposée, l'ordinateur indique le nombre d'éléments correctement placés ainsi que le nombre d'éléments présents dans la combinaison secrète mais à une autre place.

Q1 Écrire une fonction `liste_alea` prenant en argument un entier n et renvoyant une liste de n chiffres, choisis aléatoirement. On rappelle qu'un chiffre est un entier compris entre 0 et 9. On utilisera la fonction `choice` de la bibliothèque `random` (qu'il faudra importer). Cette fonction permet de choisir un élément x aléatoirement parmi les éléments d'une liste L via la syntaxe suivante :

```
1 x = random.choice(L)
```

Q2 Écrire une fonction `bonne_place` prenant en arguments deux listes de même taille L et M et renvoyant le nombre d'indices k tels que les éléments de L et M d'indice k sont égaux.

Par exemple, `bonne_place([1, 3, 7, 5, 2], [7, 3, 8, 4, 2])` doit renvoyer 2.

On ne demande pas de vérifier que les listes prises en arguments sont de même taille.

Q3 On souhaite définir une fonction `distrib` prenant en argument une liste de chiffres L et renvoyant une liste à 10 éléments dont l'élément d'indice k est le nombre de fois où le chiffre k apparaît dans L . Par exemple, `distrib([3, 2, 6, 1, 3, 5, 3, 1])` doit renvoyer la liste `[0, 2, 1, 3, 0, 1, 1, 0, 0, 0]`.

1. (Sans ordinateur). Que doit renvoyer `distrib([0, 3, 7, 3, 1, 2])` ?

2. Écrire la fonction `distrib`.

Q4 On souhaite définir une fonction `elt_communs` prenant en arguments deux listes de chiffres `L` et `M`, et renvoyant le nombre de chiffres communs aux deux listes. On souhaite compter les répétitions, mais ne pas tenir compte de la position des chiffres. Pour cela, on propose de :

- calculer les listes D_L et D_M , obtenues en appliquant `distrib` à `L` et `M`,
- calculer et renvoyer la somme $\sum_{k=0}^9 \min(D_L[k], D_M[k])$, c'est le nombre d'éléments en commun dans `L` et `M` (en êtes-vous convaincu ? À méditer)

1. (*Sans ordinateur*). En appliquant l'algorithme proposé, déterminer ce que doit renvoyer `elt_communs([2, 3, 2, 7, 1, 4, 2], [1, 5, 3, 3, 7, 2, 2])`.

2. Écrire la fonction `elt_communs`.

On rappelle que $\min(x, y)$ renvoie le minimum entre les deux nombres x et y .

Q5 L'ordinateur a choisi une liste `mystere` de 5 chiffres via la fonction `liste_alea` :

```
1 mystere = liste_alea(5)
```

Écrire une fonction `jeu` prenant en argument une liste `L` de 5 chiffres (correspondant à la proposition d'un joueur) et renvoyant le couple (a, b) , où a est le nombre de chiffres bien placés dans `L`, et b le nombre de chiffres communs à `L` et à `mystere`, mais mal placés dans `L`. Par exemple, si la liste `mystere` vaut `[7, 3, 7, 1, 4]`, alors `jeu([1, 3, 7, 5, 2])` doit renvoyer le couple $(2, 1)$.

Q6 Jouez contre l'ordinateur pour trouver la liste `mystere` !