DS d'informatique N°1 – Gains en bourse

Certaines questions demandent d'écrire du code python, vous veillerez autant que possible à utiliser une syntaxe valide et à indenter correctement le code (utilisez des lignes verticales pour marquer les différents niveaux d'indentation)

On cherche à calculer le gain maximum possible à la Bourse sur une action pendant une période de n jours, en ne faisant une opération d'achat et de vente d'une seule action. On suppose que les cours quotidiens de cette action sont enregistrés dans un tableau d'entiers naturels ($a_i \in \mathbb{N}$) de n éléments ($0 \leqslant i < n$).

On définit l'*amplitude* de la variation du cours comme la valeur absolue du maximum de la variation de ce cours pendant la période observée, c'est à dire la quantité suivante :

$$amplitude = \max_{0 \leqslant i \leqslant j < n} |a_j - a_i| = \max_{0 \leqslant i < n} a_i - \min_{0 \leqslant i < n} a_i$$

Dans ce qui suit, on utilisera le cours de bourse représenté par la liste suivante :

```
cours = [5, 1, 5, 8, 8, 10, 3, 4, 1, 2]
```

Ce cours de bourse a une amplitude de 9.

Complexité.

Le temps d'exécution (ou complexité en temps) $T(f)$ d'une fonction f de la variable a est le nombre d'opérations élémentaires nécessaires au calcul de $f(a)$. Lorsque ce temps d'exécution dépend d'un paramètre n, il sera noté $T_n(f)$. On dit que la fonction f s'exécute :

- en temps linéaire par rapport au paramètre n, s'il existe $K > 0$ tel que pour tout n, $T_n(f) < Kn$
- en temps quadratique par rapport au paramètre n, s'il existe $K > 0$ tel que pour tout n, $T_n(f) < Kn^2$

Python - listes et tuples.

Dans ce sujet, nous adopterons la syntaxe du langage Python. On rappelle qu'en Python, les listes sont des tableaux dynamiques à une dimension. Sur les listes, on dispose des opérations suivantes, qui ont toutes une complexité constante :

- `[]` crée une liste vide (c.-à-d. ne contenant aucun élément)
- `len(liste)` renvoie la longueur de la liste `liste`.
- `liste[i]` renvoie l'élément d'indice `i` de la liste liste s'il existe ou produit une erreur sinon (noter que les éléments sont indicés à partir de `0`).
- `liste[-i]` renvoie l'élément d'indice `len(liste)-i` de la liste liste s'il existe ou produit une erreur sinon. En particulier, `liste[-1]` renvoie le dernier élément de la liste.

> **Important :** L'usage d'autres fonctions agissant sur les listes telles que `min(liste)`, `max(liste)`, `sorted(liste)`, `liste.sort()` est rigoureusement interdit (ces fonctions devront être programmées explicitement si nécessaire).

On rappelle également que l'on peut récupérer directement les valeurs contenues dans un tuple (qui est une liste non modifiable) de la façon suivante : après l'instruction `a,b,c = (1,2,4)`, la variable `a` contient la valeur `1`, `b` contient la valeur `2` et `c` contient la valeur `4`. Cette instruction génère une erreur si le nombre de variables à gauche est différent de la taille du tuple à droite.

Une fonction `truc(a)` peut renvoyer plusieurs valeurs sous la forme d'un tuple en utilisant l'instruction

`return v1, v2, v3`.

On peut stocker ces valeurs dans des variables avec l'instruction `V1, V2, V3 = truc(a)`

Question 1. Écrire une fonction `maximum(a)` et une fonction `minimum(a)` qui renvoient respectivement la valeur maximale et la valeur minimale du cours représenté par la liste `a`.

Question 2. Écrire une fonction `moyenne(a)` qui renvoie la valeur moyenne du cours représenté par la liste `a`.

Question 3. Écrire une fonction `amplitude(a)` qui retourne comme résultat l'amplitude de la variation du cours représenté par la liste `a`. Donner un ordre de la complexité en temps de cette fonction en fonction de n.

Le *gain maximum* est le gain maximum possible sur la période observée, c'est-à-dire la quantité suivante :

$$gain = \max_{0 \leqslant i \leqslant j < n} (a_j - a_i)$$

Question 4. Donner un exemple où l'amplitude est différente du gain maximum. Que représente l'amplitude en terme de gain ou de perte ?

Question 5. En suivant textuellement la définition du gain, écrire une fonction `gain(a)` qui retourne, en temps quadratique (par rapport à n), le gain maximal possible sur le cours représenté par le tableau a.

Par exemple `gain(cours)` renvoie `9`.

Question 6. Modifier la fonction précédente pour écrire une fonction `gainModif(a)` qui renvoie également les deux dates i et j d'achat et de vente de l'action permettant d'obtenir le gain maximum sur le tableau a. On impose également cette fois d'avoir $j - i$ minimum. Votre fonction devra suivre le modèle suivant :

```
def gainModif(a):
    # Votre code ici
    # ...
    #
    return i, j, gainMax
```

Par exemple, `gainModif(cours)` renvoie `(1,5,9)`

Pour tout i ($0 \leqslant i < n$) on définit le gain courant maximum GC_i comme le gain maximum possible obtenu en vendant son action au temps i, c'est-à-dire :

$$GC_i = \max_{0 \leqslant k \leqslant i} (a_i - a_k)$$

Pour l'exemple de cours donnée en début d'énoncé, on a $GC_0 = 0$, $GC_1 = 0$ et $GC_2 = 4$.

Question 7. En calculant progressivement le gain courant maximum, écrire une fonction `gain1(a)` qui retourne, en temps linéaire (par rapport à n), le gain maximum possible sur le cours représenté par la liste `a`.

Question 8. Modifier la fonction précédente pour écrire une fonction `gain1Modif(a)` qui renvoie également les deux dates i et j d'achat et de vente de l'action permettant d'obtenir le gain maximum sur la liste `a`. On impose également $j - i$ minimum.

On considère maintenant le possibilité de faire séquentiellement deux transactions pendant la période observée, c'est-à-dire de considérer deux dates d'achat i et i', et deux dates de vente j et j' telles que $0 \leqslant i \leqslant i' \leqslant j \leqslant j' < n$.

Question 9. Écrire la fonction `gain2(a)` qui retourne, en temps quadratique (par rapport à n), le gain maximum possible en faisant deux transactions successives sur le cours de l'action représentée par la liste `a`. On pourra réutiliser les fonctions définies aux questions précédentes.

Par exemple `gain2(cours)` renvoie `10`.

Question 10. Modifier la fonction précédente pour écrire une fonction `gain2Modif(a)` qui renvoie également les dates i, j, i' et j' d'achat et de vente de l'action permettant d'obtenir le gain maximum sur la liste `a`. On impose encore une fois $j - i$ et $j' - i'$ minimum.