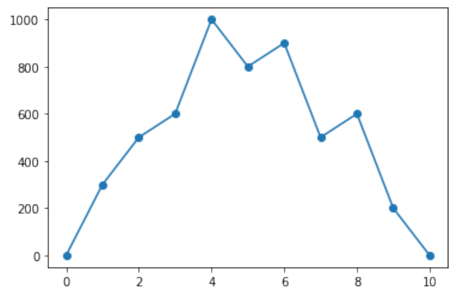
TD Informatique – Entraînement pour le concours blanc MP 2024-2025

(⋆) Exercice 1

Un altimètre est un instrument de mesure permettant de déterminer la distance verticale entre un point et une hauteur de référence. Lors d'une randonnée en montagne, un promeneur étalonne son altimètre à son point de départ, puis mesure à chaque heure l'altitude relative atteinte. À la fin de sa randonnée, il obtient une liste d'entiers naturels qu'il range dans la liste `alt`. À titre d'exemple :




```
alt = [0, 300, 500, 600, 1000, 800, 900, 500, 600, 200, 0]
```

1. On s'interdit d'utiliser la fonction `max()` de Python. Écrire une fonction `altmax`, qui prend en argument une liste `alt` et qui retourne l'altitude relative maximale atteinte lors de la randonnée.
2. Le randonneur souhaiterait savoir à quel moment son ascension a été la plus rapide en calculant le dénivelé (positif) maximal réalisé en une heure. Par exemple, pour la liste donnée en illustration, le dénivelé maximal est égal à 400 mètres et a été réalisé entre la troisième et la quatrième heure.
   Écrire une fonction Python `denivmax` prenant en argument une liste `alt` et retournant le dénivelé maximal réalisé en une heure durant sa randonnée.
3. Écrire une fonction `denivtotal` retournant la somme des dénivelés positifs réalisés durant cette randonnée. Pour l'illustration, la valeur est 1200 mètres.
4. Enfin, on appelle sommet toute altitude relative strictement supérieure à l'altitude qui lui précède et à l'altitude qui lui succède dans la liste `alt` (les extrêmités ne sont jamais des sommets). Dans l'exemple illustrant cet exercice, il y a 3 sommets (de valeurs 1000, 900 et 600, atteints à la quatrième, sixième et huitième heure de marche). Rédiger une fonction `sommets` retournant la liste des hauteurs des sommets de la randonnée.

(⋆) Exercice 2

Pour $n \in \mathbf{N}^*$, $E_n = [\![0, n-1]\!]$. On représente une fonction $f : E_n \to E_n$ par la liste `F` telle que $f(x) =$`F[x]` pour tout $x \in E_n$. Par exemple, si `F = [1, 2, 3, 0]`, alors $f(0) = 1$, $f(1) = 2$, $f(2) = 3$ et $f(3) = 0$.
On rappelle que x est un point fixe de f si $f(x) = x$.

1. Écrire une fonction `admet_point_fixe(L)` qui prend une liste L en argument et renvoie `True` si la fonction représentée par L admet un point fixe, et `False` sinon.
2. Donner la complexité de la fonction précédente.
3. Écrire une fonction `nb_points_fixes(L)` qui prend en argument une liste L et renvoie le nombre de points fixes de la fonction représentée par L.

4. On note $f^k = \underbrace{f \circ f \circ \cdots \circ f}_{k \text{ termes}}$ la k-ième itérée de f.

   (a) Pour $x \in E_n$, que vaut $f^0(x)$? que vaut $f^1(x)$? On constate que pour $k \in \mathbf{N}^*$,

   $$f^k(x) = f(f^{k-1}(x)) = f^{k-1}(f(x))$$

   (b) Écrire une fonction récursive `itere(L, x, k)` qui prend en argument une liste L, un entier x (de E_n si L est de taille n) et un entier $k \in \mathbf{N}$ et renvoie $f^k(x)$.

   (c) Écrire une fonction `nb_points_fixes_iteres(L, k)` qui prend en argument une liste L représentant f, un entier $k \in \mathbf{N}$ et renvoie le nombre de points fixes de f^k.

(⋆⋆) Exercice 3

Écrire un programme utilisant deux boucles for imbriquées afin de déterminer les deux valeurs les plus proches une liste.
Par exemple, pour `L = [45, 21, 56, 12, 1, 8, 30]`, les valeurs renvoyées doivent être 12 et 8.
