Recherche de minimums et maximums

Dans de nombreux problèmes on recherche une solution optimale parmi un ensemble de possibilités. On peut vouloir obtenir une valeur la plus grande possible, ou un élément qui est le plus proche d'un autre, ou un élément associé à un coût le plus faible possible, etc. On doit donc souvent déterminer des maximums et des minimums.
✎ Jusqu'à la question 9 (incluse) il n'y a pas de code à écrire ; il suffit de noter sur la feuille la réponse à la question posée.

Les valeurs et les indices dans une liste

Si **L** est une liste de longueur N, les *indices* de cette liste sont les entiers $0, 1, 2, \ldots, N-1$ et les *valeurs* (associées dans la liste à ces indices) sont $\mathbf{L}[0], \mathbf{L}[1], \ldots, \mathbf{L}[N-1]$.

1. On considère la liste $\mathbf{L} = [0, 3, -1, 3]$. Quels sont les indices de cette liste ?

   Réponse : _____

2. Quelles sont les valeurs associées ?

   Réponse : _____

3. Quelle est la plus grande valeur apparaissant dans **L** ?

   Réponse : _____

4. À quel(s) indice(s) dans la liste se trouve cette plus grande valeur ?

   Réponse : _____

Le maximum d'une liste

On suppose que l'on dispose d'une liste **L** qui est non vide, de longueur N et qui contient des nombres. On veut déterminer son maximum c'est à dire quelle est la plus grande valeur parmi $\mathbf{L}[0], \mathbf{L}[1], \ldots, \mathbf{L}[N-1]$.

5. Si `L=[1,3,2,1]`, quel est le maximum de **L** ?

   Réponse : _____

6. Si `L=[1.2,-3.6,2.7,7.1,3.3,0.9,3.6,-8.1,6.5,7.0,5.5]`, quel est le maximum de `L`?

Réponse : _____

7. Si `L=[3]`, quel est le maximum de `L`?

Réponse : _____

8. Si `L=[3,3,3]`, quel est le maximum de `L`?

Réponse : _____

9. Si `L=[-1,-2]`, quel est le maximum de `L`?

Réponse : _____

Le principe de l'algorithme pour obtenir le maximum de la liste `L` est le suivant :

- On va devoir parcourir toute la liste ;
- On définit une variable `maxi` qui contient la plus grande valeur rencontrée dans la liste à ce moment du parcours ;
- Cette variable est initialisée en lui donnant comme valeur la première valeur de la liste, c'est à dire `L[0]` ;
- À chaque itération, on va comparer `L[i]` (l'élément que l'on est en train de considérer) à `maxi`. Suivant que `L[i]` est plus grand ou plus petit que `maxi` on devra éventuellement changer la valeur de `maxi` ;
- Le résultat renvoyé est `maxi`.

10. Écrire la fonction `maximum(L)` qui met en œuvre cet algorithme et renvoie le maximum obtenu.

11. Tester la fonction avec `print(maximum([1,3,2,1]))`.

Réponse : _____

12. Tester la fonction avec `print(maximum([-1,-2]))`.

Réponse : _____

13. Adapter la fonction `maximum(L)` pour obtenir une fonction `minimum(L)` qui détermine et renvoie le minimum de la liste `L`.

14. Tester cette fonction avec `print(minimum([2,1,3,1]))`.

Réponse : _____

L'indice du maximum d'une liste

On dispose toujours d'une liste `L` qui est non vide et qui contient des nombres. On veut maintenant déterminer l'indice `imax` pour lequel `L[imax]` est le plus grand.

15. Si `L=[1,3,2,1]`, que vaut `imax`?

Réponse : _____

16. Si `L=[3]`, que vaut `imax`?

Réponse : _____

17. Si `L=[-1,-2]`, que vaut `imax`?

Réponse : _____

18. Si `L=[3,1,3]`, que peut valoir `imax` ?

Réponse : _____

Plus précisément, on souhaite en fait obtenir *l'un des indices* pour lequel `L[imax]` est le plus grand (car le maximum de `L` peut apparaitre plusieurs fois dans `L`).
Le principe de l'algorithme est le suivant :

- On doit toujours parcourir toute la liste ;
- On définit une variable `imax` qui contient l'indice dans la liste `L` de la plus grande valeur rencontrée à ce moment du parcours ;
- Cette variable est initialisée en lui donnant comme valeur le premier indice de la liste `L`, c'est à dire 0 ;
- À chaque itération, on va comparer `L[i]` (l'élément que l'on est en train de considérer) à `L[imax]`. Suivant que `L[i]` est plus grand ou plus petit que `L[imax]` on devra éventuellement changer la valeur de `imax` ;
- Le résultat renvoyé est `imax`.

19. Écrire la fonction `imaximum(L)` qui met en œuvre cet algorithme et renvoie l'indice du maximum de `L`.

20. Tester la fonction avec `print(imaximum([1,3,2,1]))`.

Réponse : _____

21. Tester la fonction avec `print(imaximum([-1,-2]))`.

Réponse : _____

22. Adapter la fonction `imaximum(L)` pour obtenir une fonction `iminimum(L)` qui détermine et renvoie l'indice du minimum de la liste `L`.

23. Tester cette fonction avec `print(iminimum([2,1,3,1]))`.

Réponse : _____

L'élément d'une liste pour lequel une fonction est minimale

On dispose toujours d'une liste `L` non vide, de longueur N et contenant (pour simplifier) des nombres. On dispose également ici d'une fonction f et on s'intéresse aux nombres $f(\texttt{L}[0]), f(\texttt{L}[1]), \ldots, f(\texttt{L}[N-1])$.

24. On prend ici la liste $\texttt{L} = [-2,-1,0,1,2]$ et $f(x) = (x-1)^2$. Quels sont les nombres $f(L[0])$, $f(L[1])$, $f(L[2])$, etc. ?

Réponse : _____

25. Quelle est le plus petit nombre parmi $f(L[0])$, $f(L[1])$, $f(L[2])$, etc. ?

Réponse : _____

26. Pour quelle valeur apparaissant dans la liste `L` la fonction f donne-elle le plus petit résultat ?

Réponse : _____

On veut déterminer quelle valeur, apparaissant dans la liste **L**, donne le plus petit résultat possible lorsque l'on applique f. Le principe de l'algorithme est le suivant :

- On doit toujours parcourir toute la liste ;
- On définit une variable **vmin** qui contient la valeur dans la liste **L** qui est associée au plus petit résultat quand on applique f (parmi les valeurs de la liste déjà rencontrées) ;
- Cette variable est initialisée en lui donnant comme valeur le premier élément de la liste **L**, c'est à dire **L[0]** ;
- À chaque itération, on va comparer **f(L[i])** à **f(vmin)**. Suivant que **f(L[i])** est plus grand ou plus petit que **f(vmin)** on devra éventuellement changer la valeur de **vmin** ;
- Le résultat renvoyé est **vmin**.

27. Écrire la fonction **vminimale(L,f)** qui met en œuvre cet algorithme et renvoie l'élément de la liste **L** pour lequel f est minimale.

28. Tester la fonction avec :

```
def f(x):
    return (x-1)**2

print(vminimale([-2,-1,0,1,2],f))}.
```

Réponse : _____

29. Tester la fonction avec **print(vminimale([-1,0,2],f))**.

Réponse : _____

30. Adapter la fonction **vminimale(L,f)** pour obtenir une fonction **vmaximale(L,f)** qui détermine et renvoie l'élément de la liste **L** pour lequel f est la plus grande.

31. Tester cette fonction avec **print(vmaximale([2,1,3,1],f))**.

Réponse : _____

Corrections

Q 1. $\{0,1,2,3\}$.

Q 2. $0,3,-1,3$ (une valeur est répétée).

Q 3. 3, obtenue plusieurs fois.

Q 4. Indices 1 et 3.

Q 5. 3.

Q 6. 3.

Q 7. 3.

Q 8. 3 (obtenu plusieurs fois).

Q 9. -1.

Q 10.

```
def maximum(L):
    maxi = L[0]
    for i in range(1,len(L)):
        if L[i]>maxi:
            maxi = L[i]
    return maxi
```

Q 11.

```
print(maximum([1,3,2,1]))
```

```
3
```

Q 12.

```
print(maximum([-1,-2]))
```

```
-1
```

Q 13.

```
def minimum(L):
    mini = L[0]
    for i in range(1,len(L)):
        if L[i]<mini:
            mini = L[i]
    return mini
```

Q 14.

```
print(maximum([2,1,3,1]))
```

```
3
```

Q 15. 1.

Q 16. 0.

Q 17. 0.

Q 18. 0 ou 2.

Q 19.

```
def imaximum(L):
    imax = 0
    for i in range(1,len(L)):
        if L[i]>L[imax]:
            imax = i
    return imax
```

Q 20.

```
print(imaximum([1,3,2,1]))
```

```
1
```

Q 21.

```
print(imaximum([-1,-2]))
```

```
0
```

Q 22.

```
def iminimum(L):
    imin = 0
    for i in range(1,len(L)):
        if L[i]<L[imin]:
            imin = i
    return imin
```

Q 23.

```
print(iminimum([2,1,3,1]))
```

```
1
```

Q 24. 9, 4, 1, 0, 1.

Q 25. 0.

Q 26. 1.

Q 27.

```
def vminimale(L,f):
    vmin = L[0]
    for i in range(1,len(L)):
        if f(L[i])<f(vmin):
            vmin = L[i]
    return vmin
```

Q 28.

```
def f(x):
    return (x-1)**2

print(vminimale([-2,-1,0,1,2],f))
```

```
1
```

Q 29.

```
print(vminimale([-1,0,2],f))
```

```
0
```

Q 30.

```
def vmaximale(L,f):
    vmax = L[0]
    for i in range(1,len(L)):
        if f(L[i])>f(vmax):
            vmax = L[i]
    return vmax
```

Q 31.

```
print(vmaximale([2,1,3,1],f))
```

```
3
```