TD : Algorithmes sur le listes

1. Écrire une fonction d'en tête moyenne(L) qui prend en argument une liste L composée de nombres et renvoie la moyenne de ses éléments.

2. Écrire une fonction d'en tête moyennecarres(L) qui prend en argument une liste L et renvoie la moyenne du carré de ses éléments.

3. Utiliser les deux fonctions précédentes pour écrire une fonction variance(L) qui renvoie la variance des éléments de L à l'aide du cours sur les statistiques.

4. Que renvoie cette fonction si en rentre en argument une liste de nombre L ?

```
def mystere(L) :
    S=0
    N=len(L)
    for k in range(0,N)
            S=S+(L[k]-moyenne(L))**2
    S=S/N
    return S
```

5. Calculer mathématiquement ce que renvoie moyenne(range(0,10)) puis variance(range(0,10)). Vérifier ensuite avec la console vos résultats.

6. Compléter cette fonction pour qu'elle renvoie le maximum d'une liste L.

```
def maximum(L) :
    M=L[0]
    N=len(L)
    for k in range(0,N-1):
            if L[k±1]>M:
                    M=…
    return M
```

7. Écrire une fonction similaire renvoyant le minimum d'une liste L.

8. * Écrire une fonction def mode(L) renvoyant un mode de L (on rappelle qu'un mode est une valeur apparaissant le plus de fois dans L)

9. * (*D'après Maths ESSEC 2018*) Compléter le programme pour qu'il renvoie la deuxième plus grande valeur de L.

```
def max2(L) :
        N=len(L)
        If L[0]>L[1] :
                y=L[0]
                z=L[1]
```

```
else :
        y=…
        z=…
for k in range(2,N):
        if L[k]>y:
                z=…
                y=…
        else:
                if … :
                        z=…
return z
```

10. * Écrire une fonction qui prend en argument une liste de réels et qui renvoie le nombre de réels différents de la liste.

11. Écrire une fonction mini(L) qui prend en argument une liste L et qui renvoie une liste contenant le minimum de L et un indice i de L tel que L[i] vaut ce minimum.
    Par exemple mini([1,2,9,0,-1,2]) renvoie [-1,4].
    ** Utiliser cette fonction pour coder une fonction trin(L) qui renvoie une liste L' où les éléments sont rangés par ordre croissant.

Corrigé

1. def moyenne(L):
```
    S=0
    N=len(L)
    for k in range(0,N):
        S=S+L[k]
    return S/N
```

2. def moyennecarres(L):
```
    S=0
    N=len(L)
    for k in range(0,N):
        S=S+L[k]**2
    return S/N
```

3. def variance(L):
```
    return moyennecarres(L)-moyenne(L)**2        #formule de Huygens !
```

4. Elle calcule la variance à partir de la définition de la variance (l'espérance de X moins son espérance au carré)

5. Moyenne : 4,5 et variance = 100/12

6. def maximum(L) :
```
    M=L[0]
    N=len(L)
    for k in range(0,N-1):
        if L[k+1]>L[k]:
            M=L[k+1]
    return M
```

7. Il suffit de remplacer *if L[k±1]>L[k]* : par *if L[k±1]<L[k]*

8. def mode(L)
```
    N=len(L)
    m=L[0]
    for k in range(0,N-1):
        if L.count(L[k+1])> L.count(L[k]):
            m=L[k+1]
    return m
```

9.
```
y=L[1]
z=L[0]
z=y
y=L[k]
```

```
    if L[k]>z:
        z=L[k]
```

10.
```
def nbre(L):
    n=len(L) ; T=[L[0]]        #T contiendra les éléments de L sans répétition
    for k in range(0,n-1):
        if T.count(L[k+1])==0:
            T.append(L[k+1])
    return len(T)
```

11.
```
def mini(L):
    M=L[0] ; c=0            #c sera l'indice d'un minimum et M un minimum
    N=len(L)
        for k in range(0,N-1):
        if L[k+1]<L[k]:
            M=L[k+1]
            c=k+1
    return [M,c]
```

```
def trin(L):
    N=len(L) ; T=[]                #T sera la liste ordonnée
    for k in range(N):
        c= mini(L)[1]              #on récupère l'indice du minimum de L
        M=mini(L)[0]               #on récupère le minimum de L
        T.append(M)                #on ajoute cette valeur à T
        del L[c]                   # on supprime le minimum de L
    return T
```