

Correction du Devoir Surveillé N°2

Durée : 1 h 30 min.

N.B. :

- La simplicité et la lisibilité des solutions seront tenues en compte lors de la correction et de l'évaluation ;
 - Les fonctions prédéfinies sont strictement interdites sauf celle qui retourne le nombre d'éléments d'une liste.
-

Exercice 1 : Modification d'une chaîne de caractères.**#Programme & Procédure Modification d'une chaîne de caractères****#Procédure Modification_chaine_caracteres**

def Modification_chaine_caracteres(i,x) :

"Retourne une chaîne de caractères Modifiée"

global ch

n=len(ch)

if 0<=i<n :

L=list(ch)

ch=' '

L[i]=str(x)

for k in range(n) :

ch=ch+L[k]

else:

print("Erreur")

#Programme Modification d'une chaîne de caractères

ch=input('Donner une chaîne de caractères:')

i=int(input('Donner l'indice du caractère à remplacer:'))

x=input('Donner le caractère :')

Modification_chaine_caracteres(i,x)

print(ch)

Exercice 2 : Suite de Thue-Morse.**#Programme & Fonction Thue-Morse****#Fonction Thue-Morse**

def Thue_Morse(k) :

"Retourne une liste dont les éléments sont les valeurs de la suite de Thue-Morse"

T=k*[0]

for i in range(k//2) :

T[2*i+1]=1-T[i]

T[2*i]=T[i]

if k%2 !=0 :

T[2*(k//2)]=T[k//2]

return(T)

Correction du Devoir Surveillé N°2

Durée : 1 h 30 min.

N.B. :

- La simplicité et la lisibilité des solutions seront tenues en compte lors de la correction et de l'évaluation ;
 - Les fonctions prédéfinies sont strictement interdites sauf celle qui retourne le nombre d'éléments d'une liste.
-

#Fonction Thue_Morse_Compressee

```
def Thue_Morse_Compressee(T) :
    "Retourne une liste compressée de la liste Thue-Morse"
    S=[0]
    m=1
    for i in range(1,k) :
        if T[i-1]==T[i] :
            m=m+1
            if i==k-1 :
                S=S+[m]
        else :
            S=S+[m]
            m=1
    return(S)
```

#Fonction Thue_Morse_Decompressee_Portion

```
def Thue_Morse_Decompressee_Portion(S,q) :
    "Retourne une liste décompressée d'une portion de liste compressée Thue-Morse"
    if 0<q<len(S) :
        if q%2!=0 :
            DcP=S[q]*[0]
        else :
            DcP=S[q]*[1]
    return(DcP)
return(False)
```

#Fonction Thue_Morse_Decompressee Tout Portion

```
def Thue_Morse_Decompressee (S) :
    "Retourne une liste décompressée de toute les portions de la liste compressée Thue-Morse"
    DctP=[ ]
    for i in range(1,len(S)) :
        DctP=DctP+Thue_Morse_Decompressee_Portion(S,i)
    return(DctP)
```

#Fonction Taux_Compression

```
def Taux_Compression(ThueMorse) :
    "Retourne le taux de compression de la liste Thue-Morse"
    return(len(S)/len(T))
```

Correction du Devoir Surveillé N°2

Durée : 1 h 30 min.

N.B.:

- La simplicité et la lisibilité des solutions seront tenues en compte lors de la correction et de l'évaluation ;
 - Les fonctions prédéfinies sont strictement interdites sauf celle qui retourne le nombre d'éléments d'une liste.
-

#Programme Thue-Morse

```
k=int(input("Donner le nombre des éléments d'une liste de Thue-Morse :"))
q=int(input("Donner l'indice d'une portion de la liste compressée Thue-Morse :"))
T=Thue_Morse(k)
print(T)
S=Thue_Morse_Compressee(T)
print(S)
print(Thue_Morse_Decompressee_Portion(S,q))
print(Thue_Morse_Decompressee(S))
print(Taux_Compression(T))
```

Exercice 3 : Mot de Dyck.**#Programme & Fonction Dyck****#Fonction Dyck**

```
def Dyck(L) :
    "Test si une liste est un mot de Dyck ou non"
    n=len(L)
    s=0
    for k in range(n) :
        s=s+L[k]
        if s<0 :
            return(False)
    return(s==0)
```

#Programme Dyck

```
L=[1,1,-1,-1,-1,-1,1,-1,1,1,-1,1]
print(Dyck(L))
```

Correction du Devoir Surveillé N°2

Durée : 1 h 30 min.

N.B.:

- La simplicité et la lisibilité des solutions seront tenues en compte lors de la correction et de l'évaluation ;
- Les fonctions prédéfinies sont strictement interdites sauf celle qui retourne le nombre d'éléments d'une liste.

Exercice 4 : Transformation d'un tableau à deux dimensions en un tableau à une dimension.**#Programme & Fonction Transfère un tableau 2D dans un tableau 1D****# Fonction Transfère un tableau 2D dans un tableau 1D**

```
def Trans_2D_1D(M) :  
    "Transfère une liste de liste dans une liste"  
    V=[ ]  
    for X in M :  
        V=V+X  
    return(V)
```

#Programme & Fonction Transfère un tableau 2D dans un tableau 1D

```
n=int(input('Donner le nombre de lignes de M:'))  
m=int(input('Donner le nombre de colonnes de M:'))  
M=[m*[0]for i in range(n)]  
for i in range(n):  
    for j in range(m):  
        M[i][j]=int(input('M[%d][%d]='%(i,j)))  
print(M)  
print(Trans_2D_1D(M))
```