ISM MP, Informatique
Année 2026/2027

Corrigé du TP Informatique 09

Exercice 1

1. On saisit :

```
def matr_adj(S,A):
    """ Construit la matrice d'adjacence
    du graphe de sommets S et de liste d'adjacence A"""
    n=len(S)
    M=[[float('inf')]*n for j in range(n)]
    for x in S:
        for y,nu in A[x]:
            M[x][y]=nu
    return M
```

2. On saisit :

```
def fw_cost(M):
    n=len(M)
    W=[[M[i][j] for j in range(n)] for i in range(n)]
    for k in range(n):
        for i in range(n):
            for j in range(n):
                W[i][j]=min(W[i][j],W[i][k]+W[k][j])
    return W
```

3. On saisit :

```
def fw_cost_path(M):
    n=len(M)
    W=[[M[i][j] for j in range(n)] for i in range(n)]
    predec=[[None]*n for i in range(n)]
    for i in range(n):
        for j in range(n):
            if W[i][j]<np.inf:
                predec[i][j]=i
    for k in range(n):
        for i in range(n):
            for j in range(n):
                aux=min(W[i][j],W[i][k]+W[k][j])
                if aux<W[i][j]:
                    predec[i][j]=predec[k][j]
                W[i][j]=aux
    return W,predec
```

On saisit :

```
M=matr_adj(S,A)
print("Matrice d'adjacence")
print(np.array(M))
print()
print("Matrice des plus courts chemins")
pcc,predec=fw_cost_path(M)
print(np.array(pcc))
print()
print("Prédecesseurs")
print(np.array(predec))
```

On obtient :

```
Matrice d'adjacence
[[inf 10.  5. inf inf inf]
 [inf inf inf  3. inf inf]
 [inf  2. inf  9. 10. inf]
 [inf  1. inf inf  6.  2.]
 [inf inf  2. inf inf inf]
 [inf inf inf inf  2. inf]]

Matrice des plus courts chemins
[[inf  7.  5. 10. 14. 12.]
 [inf  4.  9.  3.  7.  5.]
 [inf  2. 11.  5.  9.  7.]
 [inf  1.  6.  4.  4.  2.]
 [inf  4.  2.  7. 11.  9.]
 [inf  6.  4.  9.  2. 11.]]

Prédecesseurs
[[None 2 0 1 5 3]
 [None 3 4 1 5 3]
 [None 2 4 1 5 3]
 [None 3 4 1 5 3]
 [None 2 4 1 5 3]
 [None 2 4 1 5 3]]
```

Exercice 2

1. On saisit :

```
def rendu1(S,v):
    if v==0:
        return 0
    mini=float('inf')
    for c in S:
        if v>=c:
            aux=rendu1(S,v-c)
            if aux<mini:
                mini=aux
    return 1+mini
```

2. On saisit :

```
def rendu2(S,v):
    def M(S,v):
        if v==0:
            return 0
        else:
            if v not in memo:
                mini=float('inf')
                for c in S:
                    if v>=c:
                        aux=M(S,v-c)
                        if aux<mini:
                            mini=aux
                memo[v]=1+mini
            return memo[v]
    memo={}
    return M(S,v)
```

3. On saisit :

```
def rendu3(S,v):
    T=[0]+v*[float('inf')]
    for j in range(1,v+1):
        mini=float('inf')
        for c in S:
            # si j peut être rendu avec une pièce (cas d'égalité)
            # alors T[0] est pris en compte d'où mini=0
            if j>=c and T[j-c]<mini:
                mini=T[j-c]
        T[j]=1+mini
    return T[v]
```

4. On trouve notamment :

```
>>> rendu3([2,3],1)
inf
>>> rendu3([1,3,4],6)
2
>>> rendu3([1,2,5,10,20,50,100,200,500],29)
4
>>> rendu3([1,2,5,10,20,50,100,200,500],1989)
11
>>> rendu3([1,2,5,10,20,50,100,200,500],111111)
225
```

5. On saisit :

```
def rendu_tab1(S,v):
    memo_nb=[0]+[float('inf')]*v
    memo_piece=[0]+[None]*v
    #memo=[[0,0]]+[[float('inf'),None]]*v
    for j in range(1,v+1):
        mini=float('inf')
        piece=None
        for c in S:
            # si i peut être rendu avec une pièce (cas d'égalité)
            # alors T[0] est pris en compte d'où mini=0
            if j>=c and memo_nb[j-c]<mini:
                mini,piece=memo_nb[j-c],c
        memo_nb[j]=1+mini
        memo_piece[j]=piece
    return memo_nb,memo_piece
```

6. On saisit :

```
def argmin_rendu(S,v):
    M_nb,M_piece=rendu_tab1(S,v)
    res=[]
    if M_piece[v]==None:
        return False
    while M_nb[v]>0:
        res.append(M_piece[v])
        v-=res[-1]
    return res
```

On obtient :

```
>>> argmin_rendu1([1,2,5,10,20,50,100,200,500],1989)
[2, 2, 5, 10, 20, 50, 200, 200, 500, 500, 500]
>>> argmin_rendu1([1,2,5,10,20,50,100,200,500],111111)
[1, 10, 100, 500, ..., 500]
```