**Université MB de M'sila**
**Faculté de MI**
**Département d'informatique**
**Formation master 1 IDO S1 2024/2025**
**Module OC1**

Nom & Prénom.....
.....

**INTERROGATION ECRITE (durée 30')**

**Un examen de 5 heures est composé de 4 questions indivisibles, chaque question i a une note $n_i$ et nécessite un temps de réponse $t_i$ en heures. Ces détails sont récapitulés dans le tableau suivant :**

| Question i | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Note $n_i$ | 7 | 6 | 5 | 8 |
| Temps $t_i$ | 2 | 1 | 2 | 3 |

**On désire déterminer quelles questions choisir pour avoir une note totale maximale.**

a) **Quel est le problème modèle vu cours qui est similaire à celui-ci ?**
b) **Déterminer l'espace de recherche et calculer sa taille (nombre de solutions candidates).**
c) **Ecrire la formulation mathématique de ce problème d'optimisation.**
d) **Donner la solution de cette instance en utilisant un algorithme de programmation dynamique.**
e) **Ecrire l'algorithme correspondant.**
f) **Evaluer les complexités temporelle et spatiale de cet algorithme.**

**Réponses :**

1. **Problème du sac-à-dos (KSP).** ……0.5pt
2. **Espace de recherche S= ensemble des parties de {1,2,3,4} et |S|= $2^4$.** ……1pt
3. $\textit{Formulation matematique}$ : $\begin{cases} max \sum_{i=1}^{i=4} x_i n_i \\ x_i \epsilon \{0,1\} \\ \sum_{i=1}^{4} x_i t_i \leq 5 \end{cases}$ . ……1.5pts
4. **Resolution par l'algo de PD**……2.5pts

| | | cj | 0 | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|---|---|---|
| Item | ni | ti | 0 | 0 | 0 | 0 | 0 | 0 | |
| 1 | 7 | 2 | 0 | 0 | 7 | 7 | 7 | 7 | x1=1 |
| 2 | 6 | 1 | 0 | 6 | 7 | 13 | 13 | 13 | x2=1 |
| 3 | 5 | 2 | 0 | 6 | 7 | 13 | 13 | 18 | x3=1 |
| 4 | 8 | 3 | 0 | 6 | 7 | 13 | 14 | 18 | x4=0 |

**La solution optimale est 1110 et f(1110)= n1+n2+n3=18**……0.5pt

5. **Algorithme** ……3pt

```
For (i=0;i<=n;i++)  Tab[i][0]=0;
For (j=0;i<=C;j++)  Tab[0][j]=0;
For (i=0;i<=n;i++)
     For (j=0;i<=C;j++)
        If w[j] <= j
              Tab[i][j]= Tab[i-1][j];
        Else
              Tab[i][j]= max (Tab[i-1][j]; Tab[i-1][j-w[j]]+v[j]
I=n; j= C;
While (i>0)
    {  If (Tab[i][j]== Tab[i-1][j] )  { x[j]= 1 ; j-=w[j];}
       i-- ; }
return  x, Tab[n,C];
```

6. **Complexité temporelle = Complexité spatiale = O(n.T)** ……1pt