# TD 5 : « *Programmation Dynamique* »

## Exercice n°01 :

Considérons le problème de « ***Découpe Minimale de Barres*** » énoncé comme suit : on dispose d'un ensemble de barres métalliques ayant toutes la même longueur ***L*** (en mètres) et un ensemble de découpes demandées par des clients tel que chacune est caractérisée par une taille (en mètres). L'objectif est de réaliser toutes les découpes demandées en utilisant le minimum de barres possibles.

*Exemple :* pour **L = 10** et **D = [3, 4, 6, 7]**, la solution optimale consiste à réaliser la $2^{ème}$ et $3^{ème}$ découpes dans une barre, et la $1^{ère}$ et $4^{ème}$ découpes dans une autre barre ; à savoir, deux barres comme solution minimale.

**1.** Voici quelques indications pour résoudre ce problème d'une façon exhaustive :

- Définir une classe ***Barre*** comme suit :
  - ***int longueur*** : la longueur restante sur la barre ;
  - ***ArrayList<Integer> decoupesFaites*** : les découpes déjà faites sur la barre.
- La méthode à concevoir peut avoir la signature suivante :

  ***ArrayList<Barre> DMB(int L, int[] D, int indice, ArrayList<Barre> barresUtilisees)***

  Tel que :
  - ***indice*** : indice de la découpe courante à réaliser (ça prend **0** au niveau de l'appel initial) ;
  - ***barresUtilisees*** : une liste contenant les barres déjà consommées. Ça se pourrait qu'il existe encore de la place sur ces barres pour pouvoir réaliser les prochaines découpes.
- ***Récursivité de la méthode :*** à chaque appel de la méthode, on cherche à réaliser uniquement la découpe courante. Pour cela, on a deux possibilités à explorer :
  - On crée une nouvelle barre ; on réalise la découpe courante sur cette barre; on ajoute la nouvelle barre à la liste des barres déjà utilisées ; on génère un appel récursif en pointant sur la découpe suivante; et on récupère la solution de cet appel ;

    **OU**
  - Parmi les barres utilisées, on cherche s'il y en a une qui peut contenir la découpe courante. Si oui, alors on réalise la découpe courante sur cette barre ; on génère un appel récursif et on récupère sa solution. ***Attention :*** il doit y avoir une solution possible pour chaque barre permettant de réaliser la découpe.

- On compare entre toutes les solutions possibles pour retourner celle contenant le nombre minimum de barres.
- ***Cas d'arrêt :*** Si l'indice dépasse la taille du tableau de découpes alors on retourne la liste actuelle des barres utilisées.

**2.** Vérifiez s'il y a un chevauchement de sous-problèmes avec cette méthode exhaustive.

**3.** Si oui, transformez votre méthode en une méthode dynamique en ajoutant l'aspect de mémorisation.

## Exercice n°02 :

**1.** Proposez une méthode récursive exhaustive **int *SAD_Naif*(int C, int[] P, int [] G, int Indice)** pour le problème du *Sac à dos 0/1*. À chaque appel de cette méthode, on considère uniquement l'objet se trouvant à ***Indice***. Pour cet objet, on a deux possibilités :

- On le met dans le sac : dans ce cas là, on gagne son gain et on génère un appel récursif en changeant la capacité restante du sac ;
- On ne le met pas dans le sac : dans ce cas là, on ne considère pas son gain et on génère un appel récursif avec la même capacité du sac.

**2.** Montrez qu'il existe un chevauchement de sous-problèmes avec cette méthode exhaustive.

**3.** Transformez votre méthode en une méthode dynamique en ajoutant l'aspect de mémorisation.