INF3212 : Complexité Algorithmique 1
Fiche de TD n° 2

Exercice 1: Donner une description complète des problèmes ci-dessous. Ecrire ensuite pour chacun un algorithme glouton et évaluer la complexité de l'algorithme.

1. Rendu de la monnaie.
2. Sac à dos.
3. Location de véhicule.
4. Arbre couvrant de poids minimal.

Exercice 2: Un tas (binaire, max) est un arbre binaire quasi-complet entassé à gauche dont l'étiquette de chaque n ?ud est supérieure à celle de ces fils. Un tas comporte trois opérations :

— Entasser qui permet de conserver la structure de tas,
— Construire qui permet de construire un tas,
— Tri qui permet de faire un tri d'un tableau en place

Dans la suite nous considérons une implémentation d'un tas par un tableau.

1. Soit i l'indice d'un noeud de du tas dans le tableau. Quel est l'indice de son parent ? L'indice de son fils gauche ? L'indice de son fils droit ?
2. L'objectif de la fonction **Entasser** est de conserver la propriété de tas. Si cette propriété est violée, on fait alors descendre la clé violant la propriété dans le tas. Pour cela, on fait l'hypothèse que le tas est presque correct (seul notre noeud courant peut violer la propriété) et on rétabli la structure de données. Dans ce cas, la valeur maximale du tas enracinée en notre n ?ud courant est soit celui-ci soit dans un de ces deux fils. Écrire l'algorithme et évaluer sa complexité.
3. L'objectif de la fonction **Construire** est de convertir un tableau en tas : de passer de la représentation d'un tableau simple en structure de données plus complexe : un tas. On va alors appliquer la fonction Entasser à tous les éléments du tableau dans l'ordre décroissant. Écrire l'algorithme et évaluer sa complexité.
4. L'objectif de la fonction **Tri** est de trier un tableau. Pour cela, on construit un tas max à partir du tableau, puis pour chaque racine on la sort du tas et on recommence. Écrire l'algorithme et évaluer sa complexité.

Exercice 3: Implémenter en `C` les 3 versions suivantes du tri fusion :

1. Le tableau est scindé en 2
2. Le tableau est scindé en 3
3. Le tableau est scindé en 4

Initialiser des tableaux d'entiers de taille 1000, 5000, 10000, 50000, 10000 et sur chacun de ces tableaux, exécuter les 3 versions du tri et noter les temps d'exécution. Représenter les valeurs obtenues sur des courbes et commenter.

Exercice 4: Rappeler le problème de la multiplication de deux polynômes.

1. Écrire l'algorithme brute force de résolution de ce problème et évaluer sa complexité.
2. Écrire l'algorithme de Karatsuba de résolution et évaluer sa complexité.

Exercice 5: Les éléments d'un tableau donné de taille n sont des objets dont on peut tester l'égalité, mais qu'on ne peut pas comparer (ou non plus classer). Le champion du tableau est l'élément présent dans le tableau strictement plus que $n/2$ fois.

1. Proposer un algorithme force brute pour trouver le champion s'il existe. Évaluer sa complexité.
2. Proposer un algorithme DPR pour trouver le champion s'il existe. Évaluer sa complexité.