CPE Lyon - 3IRC / 4ETI Année 2022/2023
Structures de données et algorithmes avancés

Séance 4 - Tris

Exercice 1. Tri par sélection

Implémentez en Python le tri par sélection.

Exercice 2. Tri fusion

Question 1. Exprimez, à l'aide d'une récurrence, le temps d'exécution du tri fusion sur un tableau de taille n.

Question 2. Utilisez successivement :

- la méthode d'itération
- la méthode de l'arbre de récursion
- le Master Theorem

pour démontrer que le tri fusion est en $\mathcal{O}(n \cdot \log(n))$ comme vu en cours.

Question 3. Implémentez en Python le tri fusion, **sous forme récursive**.

Exercice 3. Tri rapide

Implémentez en Python le tri rapide, **sous forme récursive**.

Exercice 4. Comparaisons numériques

Question 1. Ecrivez une fonction `tabAlea` Python qui prend en entier deux paramètre k et n et qui génère un tableau de k entiers aléatoires compris entre 1 et n

Question 2. Ecrivez un programme qui utilise la fonction `tabAlea` pour générer des tableaux de tailles différentes (par exemple 10, 100, 1000, 10000, 100000... éléments) et qui affiche **le temps de calcul** des algorithmes des trois premiers exercices sur ces entrées (attention à bien donner le même tableau aux trois algorithmes !), et affichez les résultats obtenus sur une courbe.

Exercice 5. Problème de la distribution de bonbons

On dispose d'une carton de n paquets de bonbons ; chaque paquet peut contenir un nombre quelconque de bonbons. On souhaite distribuer des paquets à m enfants, de sorte :

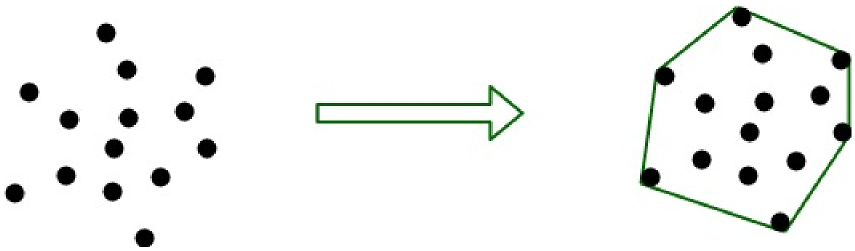
- chaque enfant reçoit exactement un paquet,
- la différence entre le paquet contenant le plus de bonbons et le paquet contenant le moins de bonbons soit **minimisée**.

Question 1. On peut résoudre ce problème en calculant tous les sous-ensembles de m paquets, et en regardant lequel donne la meilleure solution. **Ecrivez un algorithme récursif pour cette approche**. Quelle est sa complexité ?

Question 2. Cherchez et implémentez à présent une méthode *efficace* qui calcule la meilleure distribution de paquets de bonbons. Quelle est sa complexité ?

Exercice 6. Calcul de l'enveloppe convexe d'un ensemble de points

Le calcul de l'enveloppe convexe est un problème classique de géométrie algorithmique. Il consiste, étant donné un ensemble de points, à calcul son *enveloppe convexe*, c'est-à-dire le plus petit ensemble convexe qui les contient tous. Ce problème a de nombreuses applications, par exemple en analyse et reconnaissance de formes.



Question 1. L'algorithme **QuickHull** est une approche de type "Diviser pour régner", similaire à QuickSort, pour calculer une enveloppe convexe. Le principe est le suivant :

```
1. Prendre les points x1 et x2 d'abscisses minimale et maximale
2. La ligne L joignant ces deux points sépare l'ensemble de points en deux. Pour chacune des deux parti
   a. Trouver le point P le plus éloigné de L ; il est clair qu'aucun point à
      l'intérieur du triangle P-x1-x2 ne peut appartenir à l'enveloppe convexe

   b. Recommencer récursivement avec les ensembles à l'extérieur des lignes P-x1 et
      P-x2
```

Exemple :

```
Input : points[] = {{0, 3}, {1, 1}, {2, 2}, {4, 4},
                    {0, 0}, {1, 2}, {3, 1}, {3, 3}};
Output :  The points in convex hull are:
          (0, 0) (0, 3) (3, 1) (4, 4)
```

Question 2. Il existe un algorithme asymptotiquement meilleur : le *parcours de Graham*. Il consiste à trouver le point de plus petite abscisse, à trier tous les autres points par rapport à l'angle qu'ils font avec ce dernier (et l'axe des abscisses), puis à considérer les triplets de points successifs, pour déterminer lesquels sont dans l'enveloppe. Implémentez cet algorithme.

Exercice 7. Calcul du nombre d'éclipses des éléments d'un tableau

Etant donné un tableau de nombre T, on dira que $T(a)$ `éclipse` $T(b)$ si $a > b$ (i.e. $T(a)$ est à droite de $T(b)$ et $T(a) > T(b)$. Pour chaque élément du tableau, son *nombre d'éclipses* est le nombre d'éléments du tableau qui l'éclipsent. On souhaite calculer le nombre d'éclipses de chaque élément du tableau.
Par exemple :

```
Entrée:  [2, 7, 5, 3, 0, 8, 1]
Sortie:  [4, 1, 1, 1, 2, 0, 0]
```