Université de Reims Champagne-Ardenne **Pierre Delisle**
Département de Mathématiques et Informatique **Info0501**
2018/2019

Travaux Pratiques 1
Algorithmes de tri
Implémentation d'une file de priorités

Exercice 1 – Algorithmes de tri

L'objectif de cet exercice est de programmer et de comparer deux algorithmes de tri différents : le tri par insertion et le tri par tas. Avant de commencer, réfléchissez à bien structurer l'ensemble des fichiers de votre programme et à faire un makefile. Voici une suggestion d'organisation de votre répertoire de travail pour l'ensemble de ce TP :

- progTri.c : programme principal où vous créerez vos tableaux et appellerez vos fonctions de tri ;
- tri.c et tri.h : vos algorithmes de tri ;
- tas.c et tas.h : l'ensemble de vos fonctions servant à implémenter les tas ;
- util.c et util.h : l'ensemble de vos fonctions utilitaires ;
- makefile : pour compiler l'ensemble de votre programme.

Voici un pseudo-code de l'algorithme de tri par insertion qui prend un tableau $t[1..t.longueur]$ et le trie en $O(n^2)$:

```
TRI-INSERTION(t)
    pour j = 2 à t.longueur
        clé = t[j]
        i = j – 1
        tant que i > 0 et t[i] > clé
            t[i+1] = t[i]
            i = i – 1
        t[i+1] = clé
```

1) Récupérez le fichier **donneesTab1.txt** sur le site web du cours (http://cosy.univ-reims.fr/~pdelisle/enseignement.php) et importez ses données dans un tableau. La première ligne du fichier contient le nombre de valeurs du fichier et la deuxième ligne contient les valeurs séparées par un espace blanc.

2) Implémentez le tri par insertion et testez-le sur les données lues précédemment.

De son côté, comme nous l'avons vu en TD, le tri par tas peut trier ces données en $O(n \lg n)$.

3) Après avoir défini les fonctions **parent**, **gauche**, **droite**, **entasserMax** et **construireTasMax**, implémentez le tri par tas et testez-le sur les données lues précédemment.

4) Comparez le temps d'exécution de vos deux algorithmes de tri :

a) Écrivez un code permettant de générer un tableau de n entiers aléatoires et faites une copie de ce tableau dans un deuxième tableau ;
b) Estimez le temps d'exécution des deux algorithmes de tri avec n = {10, 100, 1000, 10 000, 100 000, 1 000 000, …} ;
c) Écrivez un code permettant de calculer, en secondes, le temps d'exécution de chaque algorithme de tri ;
d) Exécutez chaque algorithme de tri sur un tableau distinct et comparez les temps d'exécution avec chaque valeur de n. Observe-t-on des différences de temps significatives ? Dans l'affirmative, à partir de quelle valeur de n la différence devient-elle observable ?

Exercice 2 – Implémentation d'une file de priorités

L'objectif de cet exercice est d'implémenter une file de priorités à l'aide d'un tas.

1) Récupérez le fichier **listeTaches1.txt** sur le site web du cours (http://cosy.univ-reims.fr/~pdelisle/enseignement.php) et importez ses données dans une structure de données appropriée. La première ligne du fichier contient le nombre de tâches que vous avez à faire demain et chaque ligne suivante correspond à une de ces tâches. Sur chaque ligne, donc pour chaque tâche, la première valeur correspond à une description de la tâche et la deuxième à la priorité de la tâche.

2) Après avoir construit la file de priorités pour l'ensemble de ces tâches, affichez-là à l'écran.

3) Implémentez les fonctions **extraire-max-tas**, **augmenter-clé-tas** et **insérer-tas-max** de sorte à pouvoir retirer et ajouter des tâches à votre file de priorités. Afin de tester votre implémentation, à partir de la file de priorités définie précédemment, effectuez les opérations suivantes :

a) retirez une tâche ;
b) insérez la tâche « faire_une_sieste » de priorite 30 ;
c) donnez à la tâche « Dormir » la priorite 12
d) retirez une tâche ;
e) insérez la tâche « Embeter_mon_binome » de priorite 1.

Questions supplémentaires pour les rapides et les motivés

- Implémentez le tri par fusion vu en cours. Arrive-t-il à faire mieux que le tri par tas ?
- Le tri rapide (ou *quicksort*) a un temps d'exécution de $\Theta(n^2)$ dans le pire des cas, mais son temps d'exécution espéré est de $\Theta(n \lg n)$. Il est reconnu pour être l'un des meilleurs algorithmes de tri dans plusieurs situations. Implémentez le tri rapide et vérifiez s'il fait effectivement mieux que votre tri par tas ;
- Implémentez et comparez d'autres algorithmes de tri (tri à bulles, tri par sélection, …) ;
- Implémentez une file de priorités min.
- Un tas d-aire ressemble à un tas binaire, à ceci près qu'un nœud qui n'est pas une feuille a d enfants au lieu de 2 (à une exception potentielle près). Implémentez l'ensemble des fonctions de l'exercice 2 pour les tas d-aire.