DÉPARTEMENT D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE

SIGLE DU COURS: IFT 2015 (E14)

NOM DU PROFESSEUR: Neil Stewart

TITRE DU COURS: Structures de Données

EXAMEN FINAL

Date	:	Mercredi, 30 juillet, 2014
Heure	:	9:00-12:00
Lieu	:	AA-1355

DIRECTIVES PÉDAGOGIQUES:

- Vous disposez de trois heures pour compléter cet examen.
- Documentation permise: une page 8" x 11", recto-verso.
- Mettez tout de suite votre nom et votre matricule dans la case (Figure 1).
- Répondez sur l'examen. Il y a deux pages brouillons (page 7 et page 10).
- L'espace alloué pour la réponse indique la longueur de la réponse cherchée.
- Il y a 10 questions, avec 100 points au total.

PLAGIAT. Constitue un plagiat:

- faire exécuter son travail par un autre
- utiliser, sans le mentionner, le travail d'autrui
- ÉCHANGER DES INFORMATIONS LORS D'UN EXAMEN
- falsifier des documents

Le plagiat est passible de sanctions allant jusqu'à l'exclusion du programme.

Figure 1: Mettez votre nom et matricule ici.

1. Question 1 *(10 points)*

   Dans la Figure 2 nous voyons un exemple de SkipList randomisé. Les niveaux dans l'exemple au stade actuel sont 0, 1, 2 et 3.

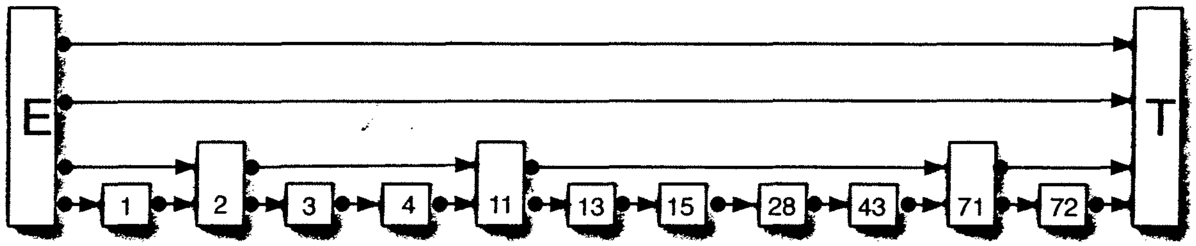

Figure 2: Skiplist

Ensuite, nous voulons insérer la clef 7. Pour ce faire, nous faisons appel à la procédure

$$\textbf{pour}(nivmax = 0; random() == 0; nivmax{+}{+});$$

qui par hasard nous retourne deux fois 0, suivi d'un 1.

Dessinez le SkipList après l'insertion de la clef.

2. Question 2 *(10 points)*

   (a) Supposons que nous ayons utilisé une liste auto-structurée, implantée en tableau, pour emmagasiner N clefs, et que, après un certain nombre d'accès, les clefs se sont ordonnées en probabilité descendante (c'est-à-dire, la probabilité que la clef sera accédée) dans le tableau. Supposons en fait que la probabilité reliée à la i-ième clef est $\frac{1}{iH_N}$, $i = 1, \ldots, N$, où H_N est le nombre Harmonique. (Il s'agit de la loi de Zipf, qui arrive souvent dans la pratique.) Trouvez le coût moyen de recherche dans ce cas, et faîtes la comparaison avec le cas où les clefs sont équiprobables.

   (b) Expliquez le lien entre l'idée de la méthode de la question 2a et les Splay Tree. Jusqu'à quel point cette comparaison est-elle valable?

3. Question 3 *(10 points)*

   Supposons un arbre rouge-noir, avec cinq noeuds, et avec la clef 4 à la racine. L'enfant droit de la racine contient la clef 5, tandis que l'enfant gauche (rouge) de la racine contient la clef 2. Enfin, les enfants du noeud qui contient la clef 2 contiennent respectivement les clefs 1 et 3.

   Dessinez l'arbre rouge-noir (en indiquant les couleurs des noeuds), dessinez un arbre 2-4 équivalent, et dessinez un "1-2-3 Deterministic Skip-List" équivalent.

4. Question 4 *(10 points)*

   Dessinez le résultat de faire les opérations suivantes, à partit d'un arbre vide, pour un SplayTree. Indiquez les sous-étapes au besoin.

   (a) Insérez les clefs 1, 2, 3, 4, 5, dans l'ordre.

   (b) Accéder à la clef 2.

   (c) Ensuite, dans une opération séparée, retirer la clef 2.

5. Question 5 *(10 points)*

   Pour l'algorithme de Prim:

   (a) Expliquez brièvement la motivation pour utiliser un monceau pour emmagasiner les "meilleures distances" à chaque étape.

   (b) Donnez la complexité de l'algorithme qui en résulte en utilisant $\Theta(\cdot)$, $|V|$ et $|E|$.

   (c) Expliquez clairement pourquoi la réponse à la question 5b montre que cette approche donne un *moins* bon résultat que l'algorithme original dans le cas d'un graphe dense.

6. Question 6 *(10 points)*

Dans le contexte du hashing avec le double hachage:

(a) Expliquez avec un exemple simple le besoin de *pierres tombales;*

(b) Dans le Devoir 3 il y avait une astuce suggérée pour réduire le coût de recherche, en modifiant les "chemins de collision". Donnez un exemple qui montre l'idée de base de cette "astuce", c'est-à-dire comment cette idée pourrait réduire le coût de recherche. (Je ne demande qu'un seul petit exemple, pas tout l'algorthme: aucune notation mathématique est nécessaire.)

7. Question 7 *(10 points)*

Dans le contexte des listes généralisées avec ficelles ("Threaded lists"), nous avons vu un algorithme de parcours. Il y avait une variable *mpf* qui indiquait si nous étions arrivés dans un noeud en montant par une ficelle, ou pas. Donnez un exemple simple qui montre qu'il est obligatoire de pouvoir savoir dans lequel des deux cas nous nous trouvons.

8. Question 8 *(10 points)*

   Mentionnez un cas où le ré-hachage sera possiblement nécessaire.

9. Question 9 *(10 points)*

   Démontrez que la hauteur d'un arbre rouge-noir ne dépasse jamais $2\lg(N+1)$, où N est le nombre de noeuds.

10. Question 10 *(10 points)*

    Donnez deux raisons pour prendre M un nombre premier dans la méthode "addressage ouvert".