Université Benyoucef Benkhedda – Alger 1
Faculté des Sciences
Département Mathématiques et Informatique
EXAMEN ASD – L2 Informatique
Février 2020 – Durée 01h30

Consignes Importantes

Les documents ne sont pas autorisés.
Le correcteur prêtera attention à la qualité de la rédaction et vous remercie d'avance d'écrire lisiblement.
Merci également de numéroter vos copies en indiquant leur nombre total.

Questions de cours (1,5 points)

Dire si les affirmations suivantes sont vraies ou fausses, en justifiant (brièvement) votre réponse.

1. Le temps d'accès au dernier élément d'une liste simplement chainée est en `O(n)`.
2. Dans un arbre binaire de recherche le minimum est toujours à la racine.
3. Le temps d'accès à l'élément maximum d'un tas min est en `O(1)`.
4. Pour un arbre binaire de recherche donné, l'obtention de la liste des nombres triés est en temps `O(n)`.
5. Dans un tas, la primitive `supprimer` consiste à supprimer une valeur située dans une feuille.

Exercice 1. (3 points)

Ecrire une fonction récursive permettant le calcule de x^k, avec x réel non nul et k entier relatif. La fonction doit impérativement être de complexité dans le pire des cas inférieure à O(n),

Exercice 2. (4 points)

- Soit `AB` un arbre binaire donnée. Ecrire les algorithmes (récursifs) permettant de convertir un arbre binaire AB donné en un arbre binaire de recherche `ABR` tout en veillant à ne pas ajouter de nœuds additionnels ou de nouvelles structures intermédiaires. Evaluer la complexité de votre dispositif.
- Ecrire l'algorithme permettant de supprimer un nœud `i` quelconque de votre `ABR`
- Ecrire l'algorithme de recherche d'une valeur donnée dans votre `ABR` et d'en estimer la complexité.

Exercice 3 (3,5 points)

On considère la suite de clés `S= (10,2,14,12,7,8,5,18)`.

1. Construire le tas max correspondant à l'insertion **successive** des clés de `S`.
2. Dérouler l'exécution de la fonction `supprimer` sur le tas max obtenu.
3. Ecrire une fonction `heap2List` qui construit une liste simplement chaînée `L` contenant tous les éléments d'un tas max dans l'ordre croissant. Quelle est sa complexité ?

Exercice 4. (8 points)

On souhaite programmer le spooler d'impression qui va gérer des requêtes de jobs d'impression envoyées à une imprimante partagée dans une imprimerie industrielle. Chaque requête comportera un identifiant ID, le nom du document, le nom de la machine, le nombre de pages à imprimer ainsi que le statut (attente, en cours, bloqué, rejeté). A noter que les documents à imprimer ont des nombres de pages différents.

PARTIE 1.

1. Dans un premier temps on souhaite implémenter une structure dynamique de type liste chainée qui prendra en charge cette gestion, la liste ne doit pas excéder 128 jobs en en attente. Proposer les structures de données ainsi que les algorithmes relatifs à la gestion de votre file d'attente en terme d'ajout d'un nouveau job et extraction d'un de votre file. Discuter ensuite cette solution en terme de complexité algorithmique.
2. On remarque que les requêtes de jobs ayant un grand nombre de pages pénalisent les autres requêtes. Discutez deux variantes de solutions possibles et Implémentez la solution adéquate la plus efficace pour palier au problème en commençant par les jobs minimaux en termes de nombre de pages.

PARTIE 2.

Pour des raisons matérielles et de complexité, on voudra tout de même maintenir l'approche dynamique tout en réduisant la complexité algorithmique lors des traitements des jobs (ajout et suppression) en se basant sur les plus petites requêtes en termes de nombre de pages.

1. Proposez la structure de données appropriées
2. Proposez une implémentation adéquate des primitives Ajouter Job et Supprimer Job de la file d'attente.
3. Quelle tendance à redouter pour les opérations multiples (de l'ordre de centaines de milliers) d'ajout et de suppression ? Proposez une solution fiable et l'implémenter en indiquant la structure de données et les algorithmes d'insertion et de suppression

PARTIE 3.

Après plusieurs mois de fonctionnement, les problèmes de performance impactent considérablement les délais de livraison de l'imprimerie. La direction décide donc de revoir fondamentalement les mécanismes de gestion et impose que cette gestion soit implémentée sans faire appel à un mécanisme d'allocation dynamique de la mémoire, c'est-à-dire uniquement à l'aide d'un (ou plusieurs) tableau(x) déclaré(s) statiquement au début du programme.

Les requêtes d'impression sont déposées par les utilisateurs dans une structure de donnée dans laquelle l'imprimante viendra extraire une tâche à exécuter.

1. On souhaite dans un premier temps que les requêtes soient traitées selon une politique « premier arrivé, premier servi ». Implémentez la solution
2. On souhaite maintenant associer une priorité à chaque requête : lors de l'exécution de la primitive Extraire c'est une requête (quelconque) de priorité maximale (parmi les requêtes présentes) qui doit être extraite. Le nombre de priorité sera supposé fini.