Nom et prénom :

Institut Paul Lambin Session de juin 2022

Examen de Structures de données

Christophe Damas, José Vander Meulen

Année(s) d'études : 2ème année de baccalauréat en informatique de gestion

Date et heure : vendredi 3 juin à 13h45

Durée : 2h ; pas de sortie durant les 60 premières minutes

Contenu

Total : **/20**

Nom et prénom :

1. Questions sur machine

Dans votre archive d'examen, vous devez avoir les répertoires suivants :

- **graphe_NOM_PRENOM**
- **arbre_NOM_PRENOM**
- **huffman_NOM_PRENOM**
- **programme_NOM_PRENOM**

Pour ces parties, voici les consignes principales:

1. Nous vous conseillons de créer 4 nouveaux projets et d'y copier-coller les différents fichiers
2. A la fin de l'examen, vérifiez bien que vos productions apparaissent bien sur le disque U :
3. A la fin de l'examen, il faut que vos noms apparaissent dans les différents répertoires. Ex : graphe_DAMAS_CHRISTOPHE.

Nom et prénom :

a) Graphe [4 pts]

Dans le dossier graphe, vous trouverez une implémentation de graphe. Cette implémentation est basée sur une liste d'arc. Nous vous demandons d'implémenter la méthode `bfs(Airport a)` dans la classe `Graph.`
Cette méthode affiche à la sortie standard les codes `iata` des différents aéroports qu'il est possible d'atteindre dans l'ordre d'un parcours en largeur (BFS) depuis l'aéroport de départ.
La méthode main fournie devrait afficher le résultat suivant :

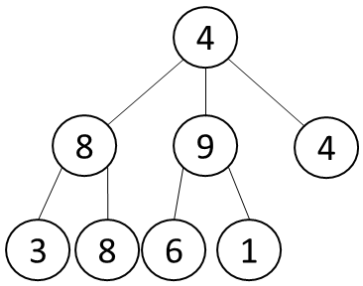
```
JFK BCN CDG DFW FCO LAX LHR MAD ORD ATL FRA DEN DXB PEK AMS MUC IST
LGW STN DME EWR IAH
```

(Attention le fichier xml ne doit pas se trouver dans le répertoire src mais à la racine du projet)

b) Récursion [5 pts]

Dans le projet **arbres**, on vous fournit la classe `Tree` qui implémente des méthodes basiques sur les arbres

1) Dans `Tree`, implémentez la méthode `contains(int x)` qui renverra true si l'entier en paramètre est contenu dans l'arbre. [2pts]
2) Dans `Tree`, implémentez la méthode `toMap()` qui renverra renvoie une map dont les clés sont les entiers présents dans l'arbre et les valeurs sont le nombre de fois qu'apparaissent ces entiers dans l'arbre [3pts]. Pour l'arbre ci-dessous, le résultat est le suivant : {1=1, 3=1, 4=2, 6=1, 8=2, 9=1}. En effet, les chiffres 1,3,6 et 9 apparaissent une fois dans l'arbre alors que les chiffres 4 et 8 apparaissent deux fois.



c) Huffman [4 pts]

Dans le répertoire `Huffman`, vous trouverez un projet presque complet qui implémente la méthode de Huffmann pour compresser et décompresser un fichier. Implémentez la seule méthode manquante : `expand(Node root, String t).` Cette méthode décode une chaine de 0 et de 1 codée à l'aide de l'algorithme de Huffman.

Nom et prénom :

d) Programme [4 pts]

Dans le répertoire `programmes`, nous fournissons un squelette de code qui a pour but de gérer les validations d'unités d'enseignement par les étudiants. Ce programme calcule pour chaque étudiant le nombre d'ects déjà validé et peut afficher la liste des étudiants triés par nombre d'ects validés. Chaque étudiant est identifié par son numéro de registre national. Une unité d'enseignement est identifiée par son nom.

L'objectif de cette question est de compléter les deux méthodes manquantes de la classe `ProgrammesEtudiants` qui enregistre les programmes de tous les étudiants :

- `valider(Etudiant e, UniteEnseignement ue)` : Enregistre la validation de l'unité d'enseignement par l'étudiant et met à jour le nombre d'ects validé par l'étudiant. Si l'unité d'enseignement a déjà été validée par l'étudiant, la méthode lance une `RuntimeException` avec le message 'ue déjà validée'
- `afficherEtudiantsTriesParEcts()` : affiche la liste des étudiants triée par nombre d'ects validés

Vous devez garantir une efficacité maximale pour ces méthodes.

Vous pouvez ajouter.modifier des attributs dans la classe `ProgrammesEtudiants` et vous pouvez également compléter le constructeur de cette classe. Vous pouvez également rajouter des méthodes/attributs dans `Etudiant` et `UniteEnseignement` si nécessaire.

Dans les cadres ci-dessous, donnez les complexités de vos deux méthodes en fonction de n (nombre d'étudiants) et m (nombre d'unité d'enseignement) :

`valider(Etudiant e, UniteEnseignement ue):`

`afficherEtudiantsTriesParEcts() :`

Une méthode main est fournie. L'output attendu est le suivant :

```
Alain Delcourt 10
Pol Durant 8
Jean Michel 0
Exception in thread "main" java.lang.RuntimeException: UE deja validée
      at ProgrammesEtudiants.valider(ProgrammesEtudiants.java:42)
      at ProgrammesEtudiants.main(ProgrammesEtudiants.java:82)
```

Nom et prénom :

2. Question sur papier

Minimax [3pts]

Soit le jeu des 4 nombres. Il a les mêmes règles que le jeu des 10 nombres présenté dans les transparents du cours théorique sur l'algorithme Minimax.
Supposons que l'on commence avec les 4 nombres suivants : 3 5 1 2
Sur la feuille A3 du questionnaire d'examen, nous avons dessiné l'arbre du jeu.
Répondez aux deux questions suivantes sur la feuille A3 présentant l'arbre du jeu :

1) Pour chaque noeud de l'arbre, calculez sa valeur Minimax.
2) Déterminez quel est le meilleur coup à jouer pour le premier joueur : le premier joueur devra-t-il commencer par prendre le 3 ou le 2 ?