IFT 2015 : Devoir 2

Professeure : Esma Aïmeur
Démonstrateurs : Alexis Roger et Dorsaf Sallami

Hiver 2023

1 Consignes générales

- Date de rendu : dimanche 19 février à 23h59.
- Ce travail est à faire seul(e) ou à deux.
- Aucun plagiat ne sera accepté.
- Il est possible que des précisions ou des modifications de l'énoncé soient envoyées par courriel et mises sur Studium.
- Chaque jour de retard pour la remise de votre travail entraînera une pénalité de 5 points.
- Tout code est à faire avec Java JDK 17
- Vous devez soumettre les fichiers Java demandés et un unique PDF avec les réponses écrites.
- Le PDF peut être des photos/scans de réponses écrites mains. Attention à la lisibilité!
- Si vous êtes en équipe : l'un soumet les fichiers demandés et l'autre soumet uniquement un PDF avec le nom de son partenaire.
- L'évaluation du code peut être automatisée, assurez-vous d'avoir les même signatures de fonctions.
- Vous avez le droit de créer autant de fonctions intermédiaires que vous voulez.
- Si vous le souhaitez, un squelette du code est disponible ici : git
- Pour toutes questions contactez : alexis.roger@umontreal.ca
- Bon travail!

2 Bonus (1 point)

Pour ce point bonus il faut utiliser git avec son partenaire de devoir.
Forkez le repo squelette dans votre GitHub/GitLab perso.
Mettre le répo en privé!
Ajoutez moi (@Alexis-BX) et votre coéquipié.
Il faut au moins 1 commit par exercice, avec chaque membre de l'équipe qu'en fait au moins un.

3 Liste chainée

Aucune opération de liste ou méthode Java n'est autorisée.
Partez du squelette de code donné dans Node.java
Les fonctions décrites comme "itératives" doivent être faites avec au moins une boucle (*for* ou *while*).

3.1 Mise en place des bases (9 points)

Liste des fonctions à implémenter dans la classe Node pour faire une liste chainée.

Fonction	Signature	Processus	Points
Constructeur	public Node(int value)		0.5
Constructeur	public Node(int value, Node next)		0.5
Ajouter une valeur	public void addValue(int value)	récursif	1
Concatener une liste	public void addNode(Node next)	récursif	1
Retirer le dernier élément	public void removeLast()	récursif	1
Retirer tous les éléments avec une certaine valeur	public void removeValue(int value)	récursif	1
Renvoie la longueur de la liste	public int length_iteratif()	itératif	1
Renvoie la longueur de la liste	public int length_recurssion()	récursif	1
Renvoie le n-ième élément en partant de la fin en O(n)	public int returnNlast(int nLast)	itératif	2

3.2 Implémentation d'une fonction de tri (3 points)

Les 2 fonctions suivantes serviront à faire des listes chainées triées. L'algorithme de tri a implementer est un insert sort.
Cet algorithme trie une liste d'éléments en prenant à plusieurs reprises un élément de la partie non triée de la liste et en l'insérant a la bonne position dans la partie triée.
Le plus petit élément vient en premier et le plus grand à la fin.

Fonction	Signature	Processus	Points
Ajouter un élément dans une liste ordonnée	public void addValue_ordered(int value)	récursif	1
Trier la liste	public void insertSort()	itératif	2

4 File circulaire

4.1 Les bases (4 points)

Le but de cet exercice est d'implémenter une file circulaire dans un tableau dans le fichier File.java
Aucune opération de liste n'est autorisée.
La taille maximale de votre file circulaire est de 100 éléments, et elle ne contiendra que des entiers (int).
Rappeles :

- Les éléments d'une file circulaire sont "first in, first out".
- On ne peut voir que le premier et le dernier élément d'une file circulaire.

Fonction	Signature	Points
Ajouter un élément a la file circulaire	public void push(int element)	0.5
Retirer un élément de la file circulaire	public int pop()	0.5
Renvoie la longueur de la file circulaire	public int length()	0.5
Affiche la liste circulaire de la sorte : (1, 3, 2, 3)	public void print()	0.5
Renvoie vrai si l'élément est dans la liste, faux sinon	public boolean search()	1
Retire la première apparition de la valeur	public void remove(int value)	1

5 Liste quadruplement chainée

Ces classes nous permettent de simuler le jeu “Change Change”.
Ce jeu peut être résumé comme suit : Nous avons 11 pièces dont quatre ont la valeur 1; deux ont la valeur 5; quatre ont la valeur 10; une à la valeur 25. La position avec la croix (X) représente un espace vide.

25	10	1	10
1	5	5	10
1	10	1	X

Completer le fichier Grid.java

5.1 La classe Cell et Grid (1 point)

Implémenter une classe Cell dans Grid, qui représente une case du tableau, qui stock sa valeur et un pointeur a chacun de ses voisins: à gauche, à droite, en haut et en bas.
On donnera la valeur de -1 a la case vide.
Dans votre classe Grid, implémenter une variable grid, qui est une liste des 12 Cell du jeu.

5.2 Bouger des cases (1 point)

Faites une fonction move qui bouge une case sur l’espace vide, et renvoie true. Tout mouvement illégal sera ignoré et renvoie false.
Signature: public boolean move(Cell case)

5.3 Résoudre le jeu (2 points)

Le but du jeu est de déplacer les pièces de telle sorte que la première et la dernière rangée soient identiques. Par exemple, la configuration suivante est une solution possible :

5	10	1	10
1	1	X	25
5	10	1	10

Les taches à faire:
Une fonction pour vérifier si le jeu est résolu. public boolean check_complete()
Une fonction qui va bouger les pièces pour résoudre le jeu. public void solve_game()