POLYTECHNIQUE MONTRÉAL

Questionnaire
Contrôle Périodique 3

LOG3430

Sigle du cours

Identification de l'étudiant(e)		
Nom :	**Prénom :**	
Signature :	**Matricule :**	**Groupe :**

Sigle et titre du cours	*Groupe*	*Trimestre*
LOG3430 - Méthodes de test et de validation du logiciel	**Tous**	**20153**
Professeur	***Local***	***Téléphone***
Giuliano Antoniol	**L-2708**	

Jour	*Date*	*Durée*	*Heures*
Mercredi	**21 octobre 2015**	**1 heure**	

Documentation	*Calculatrice*	
☐ Aucune ☒ Toute ☒ Voir directives particulières	☐ Aucune ☒ Toutes ☐ Non programmable	**Les cellulaires, agendas électroniques ou téléavertisseurs sont interdits.**

Directives particulières

Toute documentation est permise, ainsi que les calculatrices, à l'exception toutefois de tout dispositif connecté à Internet.

Important

Cet examen contient **1** exercice et ☐ question sur un total de **11** pages **(excluant cette page)**

La pondération de cet examen est de **5** **%**

Vous devez répondre sur : ☒ le questionnaire ☐ le cahier ☐ les deux

Vous devez remettre le questionnaire : ☒ oui ☐ non

L'étudiant doit honorer l'engagement pris lors de la signature du code de conduite.

Exercice 1 – 20 points

Considérez les classes SetOfInt, SetOfSortedInt, SetOfBigInt données en annexe et le client SetClient :

```
public class SetClient{
public static void main(String args[]){
    SetOfInt si= new SetOfInt();

    for (int i=0;i<4;i++){ // test set  1
       int n=(int)(Math.random()*100*(i+1)+1);
       System.out.println("Adding: " + n);
       si.add(n);

    }

    for (int i=-2;i<2;i++){ // test set 2
       System.out.println("Adding: " + i);
       si.add(i);

    }

    System.out.println("Set of int:");
    si.printSet();
}
}
```

Q1.1 Pour la classe **SetOfInt** complétez le tableau suivant pour la méthode MaDUM (ajoutez des reporteurs (reporters) si nécessaire). Utilisez les numéros dans le commentaire des méthodes au lieu de leurs noms (c-a-d numérotez les méthodes si nécessaire) ; considérez aussi les abréviations suivantes : T : transformateur, C : constructeur, O : autre, R reporteur (reporter). Voir l'exemple pour les constructeurs (3 points)

	1	2	3	4	5	6	7	8	9	10	11	
data	C	C	T	O	T	O	T		O	R		
manyItems			T	O	O		T	R	O			
INIT.											R	

Justification : We add a reported for data say getDate number 10 and then we may also add a reported for initial capacity say 11 even though this is actually a final and thus cannot be changed.

Q1.2 Identifiez toutes les tranches de la classe **SetOfInt** dans le tableau suivant -- Utilisez les numéros dans le commentaire des méthodes au lieu de leurs noms : (1 point)

data	1, 2, 3,4,5,6,7,9,10
manyItems	1,2,3,4,5,7,8,9
INIT.	1,2,11

Justification :

Q1.3 Pour data et manyItems, donnez les séquences de tests dans les tableaux suivants : (2 points) (Il n'est pas nécessaire de donner les séquences pour les deux constructeurs !) -- Utilisez les numéros dans le commentaire des méthodes au lieu de leurs noms

Séquences de tests pour tranche data :

1	10	3	10	5	10	7	10
1	10	3	10	7	10	5	10
1	10	5	10	3	10	7	10
1	10	5	10	7	10	3	10
1	10	7	10	5	10	3	10
1	10	7	10	3	10	5	10

We just need 6 sequences the second set if obtained replacing 1 with 2.

Séquences de tests pour tranche manyItems :

1	8	3	8	7	8	
1	8	7	8	3	8	

We have just 2 transformers 3 and 7 plus the reporter aka the 8.

Q1.4 Pour la classe **`SetOfSortedInt`** Faut-il changer des colonnes de la MaDUM de la question Q1.1 et/ou ajouter des lignes? Justifiez votre réponse. (1 point)

We do not need to add rows but we must add a column as the overidden method add changes the behaviour sorting the data. This also imply we have to retest the slices of data and manyItems with the overidden method add. Of course we must also add a column for the new method sort which in turn change data thus is a transformer.

Q1.5 Pour la classe **`SetOfBigInt`**, quelles tranche(s) faut-il re-tester ? Justifiez votre réponse. (1 point)

SetOfBigInt is a bit different as it adds two attributes : filled and big. This means we must add a line in the MADUM for filled and a line for big. The overidden method add is a transformer for manyItems, filled, big and data and thus we must retest the slices of data and manyItems plus we now have to test for filled and big. The slice of filled (big) just contains the constructors, add and getFilled (getBig). In other words each just add a test sequence per constructor.

Q1.6 Pour la classe **`SetOfSortedInt`**, selon le critère de Harrold McGregor, quelles méthodes doivent être re-testée(s)? Justifiez votre réponse et comparez la réponse avec les réponses des questions Q1.4 et Q1.5. (2 points)

Since the veridden methods interacts with manyItems and data we are forced to retest all methods that in a way of the other interact with manyItems and data. This is similar to the MADUM as we are foced to retest completely the two slices of data and manyItems.

Q1.7 La classe SetOfInt a été testée avec le client SetClient, on veut obtenir 100% de la couverture des branches pour les méthodes surchargés de la classe **SetOfSortedInt;** Proposez les séquences de test à utiliser. Justifiez votre réponse. (1 point)

We observe that the data user to test SetClient are all ordered and thus the sorting algorithm actually does nothing. More pecisely the if data[i] >data[i+i] is never true and thus we miss the if block and we do not swap. To get 100% coverage we just need to give any seuquence of integers not ordered for example 4, 1.

Q1.8 Pour la classe **SetOfBigInt ,** faut-il changer des colonnes de la MaDUM de la question Q1.1 et/ou ajouter des lignes ? Justifiez votre réponse. (1 point)

See the answer to question Q1.5

Q1.9 Pour les classes SetOfInt, SetOfSortedInt, SetOfBigInt, est-ce que le principe de Liskov est respecté? Justifiez votre réponse. -- svp identifiez le/les invariants. (4 points)

We first observe that the invarant is manyItems : it reprent the number of valid data and what is after manyItems is not important. The class SetOfBigInt breaks the condition as manyItems is incremented even if the data is not actually added and thus the set makes no sense.

Q1.10 Si le principe de Liskov n'est pas respecté, proposez les modifications à faire pour le faire respecter. (4 points)

There are two possible fix. The first and the esiest is to remove the else i.e., the manyItems++
In this way manyItems point to the currect array position and the set makes sense. However this path somehow breaks the meaning of set. A better fixt would be

1) remove the else the manyItems++
2) extract in a different if the update of big this imply removing the element >big and the line big=element and add the if
   if (element >b) big=element

Finally, we must notice that also the method remove must be overidden and modified as in fact we must keep consistent the class SetOfBigInt if we remove the biggest element and thus we have to change the set contained biggest element.