Examen de test et vérification

EISTI

Décembre 2019

Exercice de Test (10 points)

Soit le code suivant :

```
1  public static int[] triBicolore(int[] t) {
2          int i;  // start iterator
3          int izero; // last index for next zero (last box)
4          i = 1;
5          izero = t.length - 1;
6          while (i < izero) {
7              // if read a zero, swap with last index
8              if (t[i] == 0) {
9                  t[i] = t[izero];
10                 t[izero] = 0;
11                 izero--;
12             }
13             i++;
14         }
15         return t;
16 }
```

Ce code réalise le tri suivant : pour un tableau composé aléatoirement de 0 et de 1, il place les 1 de manière contigüe, au début et les zéros à la fin.

1. Dessinez le graphe de flot de contrôle de la fonction `triBicolore`. (1pt)
2. Déterminez **les cas de test** minimaux afin de couvrir toutes les instructions. (1pt)
3. Ajouter les cas de test qui : (1pt)
   — n'exécute pas la boucle `while`,
   — exécute **exactement une fois** la boucle `while`.
4. Indiquez la faute grossière mise en avant par la question précédente, la corriger. (1pt)
5. Écrivez dans une classe de test JUnit (4 ou 5), l'ensemble de vos méthodes de tests pour les cas de tests que vous avez identifiés. (1,5pt)
6. Certains de vos tests ne devraient pas passer, malgré la modification réalisée en question 4. Identifier la faute et la ligne. (0,5pt)
7. Quel est ce cas de test, pourquoi provoque-t-il une faute qui se voit ? Si vous ne l'aviez pas dans vos tests, proposez un cas de test qui exécute la faute et ne passe pas. (1pt)
8. Proposez un cas de test qui n'exécute pas la faute. (1pt)
9. Proposez un cas de test qui exécute la faute mais qui ne se voit pas. (1pt)
10. Corrigez la faute et vérifiez que tous vos tests passent. (1pt)

Exercice de Vérification (10 points)

Soit code suivant :

```
1  public class Verif2019 {
2
3          /*@ requires..(1)..;
4          @ ensures (0 <= \result < a.length)
5          @ && \forall integer i; ..(2).. ;
6          @*/
7      public static int max(int[] a) {
8                  int x = 0;
9                  int y = a.length-1;
10
11                 /*@ loop_invariant (0 <= x <= y < a.length)
12                 @ && (\forall integer i; ..(3).. ;
13                 @*/
14                 while (x!=y) {
15                         if (a[x] <= a[y]) x++;
16                         else y--;
17                 }
18                 return x;
19     }
20 }
```

1. Déterminer en une phrase ce que fait ce code (2pt)
2. Complétez la précondition ..(1).. . (1pt)
3. Complétez la postcondition ..(2).. . (2pt)
4. Complétez l'invariant de boucle ..(3).. . (3pt)
5. Utilisez le logiciel Why (commande krakatoa) pour démontrer que le programme ci dessus est partiellement correct. (1pt)
6. Trouver une solution pour le rendre totalement correct. (1pt)