ORDRE DES INGÉNIEURS DU QUÉBEC

SESSION DE MAI 2017

Toute documentation permise
Calculatrices : modèles autorisés seulement
Durée de l'examen : 3 heures

14-IF-A4 Conception de programmes et structures de données

Les langages de programmation sont le C, C++ et Java.

1. [10 points]
   Pour chacun des segments de code suivants, donnez la fonction $f(n)$ dont le nombre d'instructions exécutées dans le segment est $O(f(n))$, Big-Oh (l'ordre de grandeur en fonction de n) Les fonctions $f(n)$ doivent être simplifiées le plus possible.

   — Segment 1 :

   ```
   for(int i = 0, Sum = 0; i < n; i++)
   Sum++;
   ```

   — Segment 2 :

   ```
   for (int i = 0, Sum = 0; i < n; i++)
   for (int j = 0; j < n; j++)
   Sum++;
   ```

   — Segment 3 :

   ```
   for (int i = 0, Sum = 0; i < n; i++)
   Sum++;
   for (int j = 0; j < n * n; j++)
   Sum++;
   ```

   — Segment 4 :

   ```
   for (int i = 0, Sum = 0; i < n; i++)
   for (int j = 0; j < i; j++)
   Sum++;
   ```

— Segment 5 :

```
for (int i = 0, Sum = 0; i < n; i++)
for (int j = 0; j < i * i; j++)
for (int k = 0; k < j ; k+)
Sum++;
```

2. [10 points]
   Indiquez vrai ou faux et justifiez.
   (a) $2^{n+1} = O(2^n)$?
   (b) $2^{2n} = O(2^n)$?
3. [10 points] Donnez toutes les préconditions, les post-conditions et les invariantes de boucle de la fonction (méthode) `factorielle`.

```
//-----------------------------------
// Calcule factorielle pour un entier.
// Precondition:
// Postcondition:
// Loop invariant:
//-----------------------------------
public static int factorielle (int n) {
  int result  = 1;

  for (int i = n; i > 1; --i) {
   result *= i;
  } // end for
  return result
} // end factorielle
```

4. [10 points]
   Supposez que vous avez un paquet non trié de 52 cartes, 13 cartes de chaque figure ; et 4 figures (trèfle, carreau, cœur et pique).Vous devez trier le paquet pour obtenir un paquet par ordre : de 4 petits paquets de figures de 13 cartes chacun par ordre croissant. Quelle est la méthode la plus rapide parmi les suivantes ? Indiquez pourquoi.
   (a) Prenez tous les trèfles et faites le tri ; faites de même pour les carreaux, les cœurs et les piques.
   (b) Faites 13 tas selon le numéro de la carte, prenez ensuite une carte pour former 4 piles avec les figures.
   (c) En une seule passe dans les cartes, chaque carte est mise dans une position relative aux cartes qui ont déjà été triées.
5. [10 points]
   La couverture d'arêtes (*edges*) d'un graphe $G = (V, E)$ est un sous-ensemble de sommets $V' \in V$ tel que chaque arête en E contient au moins une arête de V'. Si toutes les feuilles, de n'importe quelle recherche en profondeur de l'arbre

de G sont détruites. Est-ce que les arêtes qui restent forment une couverture d'arêtes ? Donnez une preuve ou un contre-exemple.

6. [10 points]
   Il y a huit îles dans un lac, on veut construire sept ponts afin de connecter toutes les îles de telle manière qu'une île puisse être accessible à partir d'une autre en passant par un ou plusieurs ponts. Pas de connexion avec le bord du lac. Le coût d'un pont est proportionnel à sa longueur. Les distances entre les îles sont spécifiées dans le tableau suivant :

	1	2	3	4	5	6	7	8
1	-	248	210	340	280	200	345	120
2	-	-	265	175	215	180	185	155
3	-	-	-	260	115	350	435	195
4	-	-	-	-	160	330	295	230
5	-	-	-	-	-	360	400	170
6	-	-	-	-	-	-	175	205
7	-	-	-	-	-	-	-	305
8	-	-	-	-	-	-	-	-

   Trouvez les ponts à construire de telle manière que le coût soit minimal.

   Indiquez l'algorithme que vous avez utilisé et indiquez comment vous avez trouvé la solution.

7. [10 points]
   Donnez le code pour implémenter une liste chaînée, puis donnez le code pour les méthodes (fonctions) pour `eliminerDupliquées` de la liste et pour `inverser` la liste.

   Pour la première méthode, la liste (a, b, a, a, b, c, a) retourne (a, b, c).

   Pour la deuxième méthode, la liste (a, b, c, d, e) retourne (e, d, c, b, a).

8. [10 points]
   Écrivez une méthode (fonction) récursive pour calculer la somme de n premiers entiers dans un tableau contenant au moins n entiers.

   ```
   int somme( int arr[], int n ) { ... }
   ```

9. [10 points]
   Donnez un algorithme pour trouver les k plus petits éléments d'un ensemble pas trié de n entiers en $O(n + k \log n)$.

10. [10 points]
    Utilisez l'idée de partitionnement du *quicksort* pour écrire un algorithme qui trouve l'élément médian d'un tableau de n entiers dans un temps $O(n)$.

Fin de l'examen.