Examen intermédiaire du 28/01/2010 (1h)

TOP: Techniques and tOols for Programming

Première année

Documents interdits, à l'exception d'une feuille A4 à rendre avec votre copie.

La notation tiendra compte de la présentation et de la clarté de la rédaction.

★ Questions de cours. (5pts)

▷ **Question 1:** (2pts) Décrivez en quelques mots chacun des algorithmes suivants : tri à bulle, tri par sélection, tri fusion et tri par insertion. *Describe each of the following algorithms : bubble sort, selection sort, merge sort and insertion sort.*

▷ **Question 2:** (1pt) À quelles classes de complexité (en notation Θ) appartiennent les algorithmes 1 et 2 suivants ? *To which complexity class (using Θ notation) do each of the following algorithm belong ?*

algorithme 1

```
1 pour i = 1 à n faire
2   pour j = 1 à n faire
3     x += 3
```

algorithme 2

```
1 pour i = 1 à n faire
2   pour j = 1 à n faire
3     x += 3
4 pour i = 1 à n faire
5   y = x + 5
```

▷ **Question 3:** (1pt) Quelle est la différence entre la complexité dans le meilleure cas et la borne inférieure de complexité ? *What's the difference between the complexity in best case and the lower bound on complexity ?*

▷ **Question 4:** (1pt) Définissez les types de récursivité suivants : terminale, générative, mutuelle (ou croisée) et structurelle. *Define the following types of recursion : terminal, generative, mutual and structural.*

★ Exercice 1: Code récursif mystère (5pts).

Considérez le code mystère suivant. *Consider the following mystery code.*

```
1 public int puzzle(int i, int j) {
2   if (i == 1)
3     return j;
4   if (i % 2 == 1)
5     return j+puzzle(i/2,j*2);
6   else
7     return   puzzle(i/2,j*2);
8 }
```

▷ **Question 1:** ($\frac{1}{2}$pt) Explicitez les appels récursifs effectués pour `puzzle(25,4)`. *Detail the recursive calls made for `puzzle(25,10)`*

▷ **Question 2:** (1pt) Calculez le résultat de la fonction puzzle pour les valeurs suivantes. *Compute the result of the puzzle function for the following values.* (10,1) (10,2) (10,3) (10,4) (10,6) (10,8) (10,10) Que semble calculer `puzzle()` ? *What seems to compute `puzzle()` ?*

▷ **Question 3:** ($\frac{1}{2}$pt) Montrez la terminaison de cet algorithme. *Show that this algorithm always terminate.*

▷ **Question 4:** ($\frac{1}{2}$pt) Quelle est la complexité algorithmique de `puzzle` (en nombre d'appels récursifs) ? *What is the algorithmic complexity of `puzzle` (in amount of recursive calls) ?*

▷ **Question 5:** ($\frac{1}{2}$pt) Est-il possible de dérécursiver directement cette fonction ? Pourquoi ? *Is it possible to transform this function directly into a non-recursive form ? Why ?*

▷ **Question 6:** (2pts) Dérécursivez cette fonction en appliquant les méthodes vues en cours (en une ou plusieurs étapes). Explicitez ce que vous faites et pourquoi. *Change this function into a non-recursive form (in one or more steps). Explicit what you are doing, and why.*

★ Exercice 2: Encore un code mystère (mais pas récursif) (d'après Maylis DELEST – 5pts).

On considère le tableau T012={1, 0, 2, 2, 0, 1, 0, 2, 1} et la fonction `swap(tab, a,b)`, qui inverse les valeurs des cases `tab[a]` et `tab[b]`. *We consider the array T012 above, and the function `swap(tab, a,b)` which swaps the values of the cells `tab[a]` and `tab[b]`*

```
1  void puzzle2(int tab[]) {
2    int i=0,j=0,k=tab.length-1;
3    while (i<=k) {
4      if (tab[i] == 0) {
5        swap(tab,i,j);
6        j=j+1;
7        i=i+1;
8      } else if (tab[i] == 2) {
9        swap(tab,i,k);
10       k=k-1;
11     } else {
12       i=i+1;
13     }
14   }
15 }
```

▷ **Question 1:** (1pt) Quel est le résultat de la fonction puzzle2() sur le tableau T012 ? *What is the result of this function onto T012 ?*

▷ **Question 2:** (2pt) Dans le cas général si T est un tableau d'entiers dont les valeurs sont des {0, 1 ou 2}, quel est le résultat de la fonction puzzle2() sur T ? Argumentez votre réponse. *In the general case, if T is an array of integers which values are in {0,1,2}, what is the result of this function onto T ? Justify your answer.*

▷ **Question 3:** (1pt) Quelle est la complexité de cette fonction ? Justifiez votre réponse. *What is the complexity of this function ? Justify.*

★ **Exercice 3:** (4pts) Écrivez les fonctions suivantes en utilisant le type `Chaine` muni des opérations suivantes. Si votre solution ne s'exécute pas en temps linéaire, vous serez pénalisé. *Write the following functions using the type* `Chaine` *which has the following operations. If the time complexity of your solution is not linear, you will loose points.*

$$\begin{cases} chvide : \emptyset \mapsto Chaine \\ estVide : Chaine \mapsto boolean \\ premier : Chaine \mapsto Caractère & \text{(défini ssi la chaîne n'est pas vide)} \\ reste : Chaine \mapsto Chaine & \text{(défini ssi la chaîne n'est pas vide)} \\ adj : Chaine \times Caractère \mapsto Chaine \end{cases}$$

▷ **Question 1:** (1pt) $est\_membre : \begin{cases} Chaine \times caractère \mapsto booléen \\ \text{retourne VRAI ssi le caractère fait partie de la chaîne} \end{cases}$

▷ **Question 2:** (1pt) $somme : \begin{cases} Chaine \mapsto entier \\ \text{retourne la somme de tous les éléments de la chaine (supposés être des chiffres)} \end{cases}$

▷ **Question 3:** (2pt) $retourne : \begin{cases} Chaine \mapsto Chaine \\ \text{retourne la chaine lue en sens inverse} \end{cases}$