Enseirb Semestre 1
Algorithmes et Structures de Données

Epreuve proposée par Robert Cori

23 Janvier 2007, durée 1 heure 30

Les notes de cours et de travaux dirigés sont autorisées, à l'exclusion de tout autre document.

Exercice 1 : Piles

On vous demande d'écrire un algorithme qui transforme des expressions arithmétiques usuelles en expressions postfixées. Pour cela on procède en plusieurs étapes.

Question 1. Dans un premier temps on considère des expressions qui sont formées d'une somme de termes, où chaque terme est un produit de nombres, une telle expression est par exemple : `2*5*7 + 3*2 + 8*9`, qui se transforme en : `2 5 7 * * 3 2 * 8 9 * + +`

On vous demande d'écrire un algorithme qui utilisera une pile et lira les symboles de l'expression. On supposera l'expression écrite dans un tableau `exp`, un symbole (nombre ou opérateur par case). L'action de l'algorithme variera suivant que le symbole lu est un nombre, une opération `+` ou une opération `*`.

La structuration générale de l'algorithme sera donc :

```
for (i = 0 ; i < n; ++i) {
  if (nature(exp[i]) == nombre)  {....}
  if (nature(exp[i]) == addition) {....}
  if (nature(exp[i]) == multiplication) {....}
  }
....
```

On vous demande de rempalcer les `...` par des instructions permettant d'obtenir l'expression postfixée.

Question 2. On considère dans cette question des expressions contenant des nombres, additions, multiplications mais aussi des parenthèses. On vous demande d'écrire dans ce cas un algorithme ayant la même structure que le précédent qui effectue la transformation. Par exemple : `2*(5*7 + 3*2)*3 + 8*(9+1)` sera transformée en :

`2 5 7 * 3 2 * + 3 * * 8 9 1 + * +`

Exercice 2 : Récursivité

Pour tester des circuits électroniques on a besoin d'engendrer toutes les suites de longueur n composées de 0 et de 1, par exemple pour $n = 3$ on doit engendrer les 8 suites :

$$000, 001, 010, 011, 100, 101, 110, 111$$

Question 1. Un informaticien étourdi propose l'algorithme suivant :

```
genere(int k) {
    if(k < 0) affiche(u);
    else{
        u[k] = 0;
        genere (k);
        u[k] = 1;
        genere (k);
    }
}
```

Qu'il appelle par `genere (n-1)`. Il se rend compte que le programme correspondant à cet algorithme boucle indéfiniment. Comment doit-il modifier l'algorithme pour que le programme affiche bien toutes les valeurs souhaitées ?

Question 2. Donner un algorithme qui affiche toutes les suites de longueur n composées de trois symboles : 0, 1, 2.

Question 3. On demande à l'informaticien de trouver un algorithme qui engendre toutes les suites de longueur n composées de 0 et de 1, mais dans lesquelles 1 apparaît au plus m fois. L'informaticien propose l'algorithme suivant :

```
genere(int k, int m, int m1){
       if (k  < 0 || m == m1) affiche(u);
       else {
             u[k] = 0;
             genere (k-1, m, m1);
             if (m1 < m) {
                  u[k] = 1;
                  genere(k-1, m, m1+1);
                  }
     }
   }
```

Il applique cet algorithme avec `genere(3,2,0)` pour afficher toutes les suites de longueur 4 contenant au plus 2 fois le symbole 1. Il voit alors s'afficher les suites :

$$0000, 1000, 0100, 1100, 0010, 1010, 1110, 0001, 1001, 1101, 1111$$

Ce qui manifestement n'est pas correct. Un expert lui dit alors, c'est presque bon tu dois tout juste ajouter une seule instruction à ton algorithme pour le rendre correct. Quelle est cette instruction ? Où doit-il la placer ?

Exercice 3 : Programmation dynamique

Dans cet exercice, tiré d'une question de bio-informatique, on considère une suite $u_1, u_2, \ldots, u_n$ de nombres positifs ou négatifs et on cherche l'intervalle $[p, q]$ tel que la somme

$$\sigma_{p,q} \quad = \quad \sum_{k=p}^{k=q} u_k$$

soit maximale. Ainsi si tous les u_i sont positifs l'intervalle est $[1, n]$. S'ils sont tous négatifs, l'intervalle est vide ($p > q$) car $\sigma_{p,q}$ est alors égale à 0. On se propose de procéder par la technique de la programmation dynamique : on note s_i la valeur maximale de $\sigma_{p,q}$ pour la suite $u_1, u_2, \ldots u_i$ ($p, q \leq i$) et on calcule itérativement les s_i. Par exemple on aura $s_1 = u_1$ si $u_1 > 0$ et $s_1 = 0$ sinon.

Question 1. On suppose que l'on a déterminé la valeur maximale s_{i-1} de $\sigma_{p,q}$ pour $p, q \leq i-1$ et on considère la suite $u_1, u_2, \ldots u_i$, montrer sur un exemple que l'on ne peut pas se contenter de la connaissance de s_{i-1} et de u_i pour déterminer s_i. On prendra un exemple de deux suites pour lesquelles les valeurs des deux s_{i-1} et les valeurs des deux u_i sont identiques mais qui ont des valeurs différentes pour s_i.

Question 2. On introduit alors une nouvelle somme x_i qui pour chaque i est la plus grande valeur possible des $\sigma_{p,i}$ (intervalles d'extrémité i) soit :

$$x_i \quad = \quad Max_{p \in [1, i+1]} \sum_{k=p}^{k=i} u_k$$

(noter que $x_i \geq 0$ car pour $p = i + 1$ l'intervalle est vide et la somme correspondante est nulle).

Donner une formule qui permet de déterminer s_i et x_i en connaissant s_{i-1}, x_{i-1}, u_i.

Question 3. Transformer votre formule en un algorithme de programmation dynamique qui permet de déterminer la valeur maximale s_n.

Question 4. Modifier cet algorithme pour qu'il donne aussi les bornes de l'intervalle $[p, q]$ pour lequel :

$$s_n = \sum_{k=p}^{k=q} u_k$$

est maximale.