Université de Béjaïa
Faculté des Sciences Exactes
Département d'Informatique

07 Juin 2012

Niveau : Master1

Corrigé EMD
Programmation Avancée

Durée : 02h00

Exercice N°1 (5 pts):

- Quelle est la différence entre un algorithme de programmation dynamique et un algorithme glouton ? **(0.5 pts)**

  Si un problème possède la propriété de sous structure optimale alors
  - la programmation dynamique permet de trouver la solution optimale.
  - cela ne garantit pas qu'un algorithme glouton trouvera la solution optimale (exp : problème du sac à dos) mais cela le favorise.

- Comment peut-on prouver la NP-complétude d'un problème ? **(1 pts)**

  Pour montrer qu'un nouveau problème P est NP-complet, on procède comme suit :
  1. Montrer que P est dans NP
  2. Choisir un problème, P2, NP-complet approprié
  3. Construire une réduction f, qui transformant P2 vers P.

- Donnez les étapes à suivre pour appliquer le Pattern Observer. **(1.25 pts)**
  1. Création des interfaces Observables et Observateurs.
  2. Tous les observateurs doivent implémenter l'interface Observateur.
  3. L'objet Observable doit implémenter l'interface Observable.
  4. L'observable doit gérer ses Observateurs (avoir une liste des observateurs pour les mettre à jours)
  5. Les observateurs doivent mettre à jours leurs interfaces graphiques.

- Quel est le concept qui permet de récupérer dynamiquement les informations propres à une classe Java ? **(0.5 pts)**

  La réflexivité (Introspection)

Exercice N°2 (6 pts) :

1) Les algorithmes :

a) La fonction puissance

```
Fonction Puissance (X, P : entier) : entier ;
Var      i : entier ;
Début
        I ← 1 ;                                    (1 Pts)
        Puissance ← 1 ;
        TQ (i<=P) Faire
            Puissance ← Puissance * X ;
            I ← I+1
        Fin.
Fin.
```

b) La fonction puissance

```
Fonction Polynôme (Tab : Tableau [0..n] de réel, X : réel, n: entier) : réel ;
Var      i : entier ;
Début
        Polynôme ← Tab [0] ;                       (2 Pts)
        i ←1 ;
        TQ (i<=n) Faire
            Polynôme ← Polynôme +Tab [i]* Puissance (X,i) ;
            i ←i+1;
        Fin.
Fin.
```

2) Complexité :

a) La fonction puissance

Instruction	Nbre d'opérations	Nbre de fois
I ← 1	1	1
Puissance ← 1	1	1
i<=P	1	(P-1)+1+1
Puissance ← Puissance * X	2	(P-1)+1
I ← I+1	2	(P-1)+1

(1 Pts)

$Coût_{puissance}(P) = 1+1+(P+1)+P+P$

$Coût_{puissance}(P) = 3(P+1)$

b) La fonction Polynôme

Instruction	Nbre d'opérations	Nbre de fois
Polynôme ← Tab [0]	1	1
i ←1	1	1
i<=n	1	(n-1)+1+1
Polynôme ← Polynôme +Tab [i]* Puissance (X,i)	4	(n-1)+1
I ← I+1	2	(n-1)+1

(1 Pts)

$Coût_{polynôme}(n) = 1+1+(n+1)+4n+2n+\sum_{i=1}^{n} Coût_{puissance}(i)$

$Coût_{puissance}(n) = 7n+3+\sum_{i=1}^{n} 3(i+1)$

$Coût_{puissance}(n) = 7n+3+3\sum_{i=1}^{n} i + \sum_{i=1}^{n} 1$ (1 Pts)

$Coût_{puissance}(n) = 8n+3+\frac{3n(n+1)}{2}$

$Coût_{puissance}(n) = \frac{16n+6+3n^2+3n}{2}$

$Coût_{puissance}(n) = \frac{3n^2+19n+6}{2}$

$Coût_{puissance}(n) = \frac{1}{2}(3n^2+19n+6)$

Exercice N°3 (5 pts) :

1) L'algorithme :

```
Fonction Somme (B : tableau [1..100] d'entiers, i, j : entier) : entier ;
Var      M, X, Y : entier ;
Début
    Si (i=j) Alors
        Somme ← B[i] ;
    Sinon
        M ← (i+j) div 2 ;                    (2 Pts)
        A ← somme (B, i, m) ;
        B ← somme (B, m+1, j);
        Somme ←A+B;
    Fin ;
Fin.
```

2) Complexité

Les trois étapes du paradigme diviser pour régner du calcul de la somme des éléments s'un tableau :

- **Diviser : Diviser le tableau de taille N en deux tableaux de taille N/2** (M ← (i+j) div 2)
- **Régner sur les sous-problèmes en les résolvants récursivement** (A ← somme (B, i, m) *et* B ← *somme* (B, m+1, j)).
- **Combiner : les solutions des sous-problèmes en additionner les résultats obtenus** (Somme ←A+B).

Ainsi :

(2 Pts)

a=2
b=2
C(n)= Θ(1)
D(n)= Θ(1)

⇒ T(n) = 2 T(n/2)+ Θ (1)

On a : f(n)= Θ(1)= $\Theta(n^{log_2^2-1})$ ⇒ c'est le 1er cas du théorème
⇒ T(n)= $\Theta(n^{log_2^2})$= Θ(n)

(1 Pts)

Exercice N°4 (5 pts) :

a) Trouver le plus court chemin entre chaque paire de sommets d'un graphe valué est un problème d'optimisation : ce problème possède les deux principales caractéristiques à savoir une sous-structure optimale et des sous problèmes superposés donc il est possible de résoudre ce problème par la programmation dynamique. **(2pts)**

b) L'algorithme :

```
ROCEDURE Floyd ( Var D: Tableau[1..n,1..n] d'entiers, n: entier )
        Var  k, i, j : entire;
Début
        Pour k  allant de 1 à  n  Faire
                Pour i  allant de 1 à  n Faire
                        Pour j allant de 1 à n Faire
                                D[i,j] := min ( D[i,j] , D[i,k] + D[k,j] )
                        Fin pour ;
                Fin pour ;
        Fin pour.
Fin.
```

(3 Pts)