EISTI Ecole Internationale des Sciences du Traitement de l'Information	ING1 : EXAMEN D'ALGORITHMIQUE PROCEDURALE
Rédigé par : l'équipe pédagogique du module Algo. Proc.	Ref : *ING1-ALG-PRO-EXAMEN*
A l'intention de : Etudiants des ING1	Créé le : 10/01/2012

Préambule

Cet examen dure 2h00. L'examen se fait sur feuilles. Aucun document n'est autorisé. L'ordinateur est interdit.

1. Tris et structures

1.1 Date

Soit la structure Date suivante :

```
Type Date = structure
        jour : entier
        mois : entier
        annee : entier
FinType
```

- Ecrire une fonction dont la signature est :

  datePlaceeAvant(date1 : Date, date2 : Date) : booleen

  et qui renvoie VRAI si la date date1 est antérieure à la date date2 et faux sinon.

1.2 Heure

Soit la structure Heure suivante :

```
Type Heure = structure
        heure : entier
        minute : entier
        seconde : entier
FinType
```

- Ecrire une fonction dont la signature est :

  heurePlaceeAvant(heure1 : Heure, heure2 : Heure) : booléen

  et qui renvoie vrai si l'heure *heure1* est antérieure à l'heure *heure2* et faux sinon.

1.3 DateHeure

- Définir la structure DateHeure qui est composée d'une date et d'une heure.
- Ecrire une fonction dont la signature est :

  dateHeurePlaceeAvant(dateHeure1 : DateHeure, dateHeure2 : DateHeure) : booléen

  et qui renvoie VRAI si la date et heure *dateHeure1* est antérieure à la date et heure *dateHeure2*. On comparera d'abord les dates puis s'il y a lieu les heures.

Remarque : On considère que les dates et les heures sont toujours valides.

1.4 Tris Rapides

- Ecrire une procédure qui trie un tableau de DateHeure selon un algorithme de tri en n.log(n) (tri fusion ou tri rapide). On triera d'abord par la date, ensuite par l'heure.

2. Matrices, Invariant et complexité

L'algorithme ci-dessous reçoit dans un tableau les coefficients d'un polynôme et son degré. Il renvoie dans une matrice les différentes dérivations du polynôme. La première ligne contient les coefficients du polynôme P, la deuxième les coefficients du polynôme P', la troisième les coefficients du polynôme P'', etc. Les cases inutiles ont pour valeur 0.0. Les coefficients sont rangés par ordre croissant de degré.
Dans cet exercice, on suppose qu'il existe comme pour les tableaux, une procédure qui permet d'allouer dynamiquement une matrice. Sa signature est *creerMatrice(S m : Matrice, E n : Entier, E p : Entier).*

```
Fonction derivationsPolynome(E p ; tableau() : Reel, E n : Entier) : matrice() : Reel
Variables locales
d : matrice() : Reel
i, j : Entier

creerMatrice(d, n+1,n+1)
Pour i ← 1 à n+1 pas 1
        d(1,i) ← p(i)
Fin Pour
Pour i ← 2 à n+1
        Pour j ← 1 à n+1
                d(i,j) ← 0.0
        Fin Pour
Fin Pour
Pour i ← 2 à n+1 pas 1
        Pour j ← 2 à (n-i+3) pas 1
                d(i ,j-1) ← d(i-1,j) *(j-1)
        Fin Pour
Fin Pour
Retourner d
Fin Fonction
```

- Calculer la complexité exacte de cette fonction. On ne tiendra pas compte de la complexité de la procédure creerMatrice. On rappelle que dans une instruction *Pour ...*, il y au moins une addition et une affectation pour l'incrémentation de l'indice et une comparaison pour tester à chaque début d'itération, la fin de boucle.
- Donner son équivalent avec la notation asymptotique O(???).
- On s'intéresse à la boucle *Pour i ← 2 à n+1 pas 1...* . Montrer en utilisant un invariant de boucle, qu'à la fin de cette boucle, la matrice d contient bien les coefficients décrits dans l'énoncé.
  Remarque : au début de chaque itération de la boucle, la ligne i-1 est remplie avec les bons coefficients.

3. Récursivité

Dans cet exercice nous manipulons des matrices carrées de réels. Nous vous rappelons les éléments suivants :

- Une matrice carrée est une matrice dont le nombre de lignes et le nombre de colonnes sont égaux.
- Nous appelons taille d'une matrice carrée son nombre de lignes/colonnes.
- Nous noterons $M(i, j)$ l'élément de la ligne i et la colonne j de la matrice M.

- Nous noterons $M_{i,j}$ la matrice obtenue à partir d'une matrice carrée M en supprimant la ligne i et la colonne j.
- Le déterminant d'une matrice carrée de taille n (n≥2) est donné par la formule :

$$Det(M)=\sum_{i=1}^{n}\sum_{j=1}^{n}(-1)^{i+j}Det(M_{i,j})$$

- Le déterminant d'une matrice de taille 1 est égal à son élément unique. Autrement dit dans ce cas nous avons : $Det(M)=M(1,1)$

Nous supposons qu'il existe une opération creerMatriceCarree qui prend en entrée un entier n et retourne une matrice carrée de taille n dont tous les éléments sont égaux à 0.0.
Pour la suite, on dispose d'une fonction nommée *creerMIJ* qui prend en entrée la matrice M et deux entiers i et j et qui retourne la matrice $M_{i,j}$. Vous n'avez donc pas à écrire l'algorithme de cette fonction.

- Ecrire une opération permettant de calculer de déterminant d'une matrice carrée.
- Donnez la formule permettant de calculer la complexité exacte de cette dernière opération.

4. Analyse d'algorithme

On considère la fonction et la procédure suivante :

```
fonction mystere1(n : Entier) : Booleen
 variables
 cpt : Entier
 cpt ← 2
 tant que NON(n mod cpt =0)
   cpt ← cpt+1
 fin tant que
 retourner (cpt=n)
finfonction

procedure mystere2(n : Entier)
 variables
 i, j : Entier
 pour i ← 4 à n pas 2   // de 4 à n avec des pas de 2
   pour j ← 2 à i-1
     si mystere1(j) ET mystere1(i-j)      alors
       écrire(j)
       écrire('+')
       écrire(i-j)
       écrire(' ')
   finsi
  finpour
 écrire('\n')
finpour
```

- Donner les valeurs de **mystere1** pour tous les entiers de 1 à 10. En déduire ce que fait **mystere1**.
- Donner le résultat de **mystere2** pour n=20 ?
- Quelle conjecture mathématique est illustrée par **mystere2** ? (question bonus)