NF01 - Automne 2007

Examen Final - 2 heures

ATTENTION !
Utilisez trois copies séparées, une par problème

Problème n°1 (6 points) : Récursivité

MYSTERE

1. (2 pts) Ecrire un programme Pascal qui réalise la fonction récursive suivante:
   MYSTERE(X,Y) *vaut X si Y=1*
   vaut X + MYSTERE(X,Y-1) sinon

2. (2 pts) Préciser ce que fait cette fonction (X et Y entiers naturels). Pour confirmer, une simulation effectuée sur un exemple simple sera la bienvenue.

AFFICHAGE

(2 pts) Ecrire une procédure récursive *AfficheChiffres()* qui affiche les chiffres d'un nombre entier séparés par une barre de division (barre oblique). Par exemple *AfficheChiffres(123)* affichera : *1 / 2 / 3*

Problème n°2 (6 points) : fichiers

L'objet de ce travail est de calculer la moyenne d'une classe d'étudiants. Un étudiant sera représenté par son nom, prénom et le tableau des notes qu'il a obtenues :

```
Type etudiant = record
                    Nom, Prénom : string ;
                    Notes : array [1..3] of real
                End ;
```

1. (2 pts) Ecrire la procédure permettant de créer un fichier d'étudiants et de saisir les étudiants (nom et les 3 notes obtenues) d'une classe.

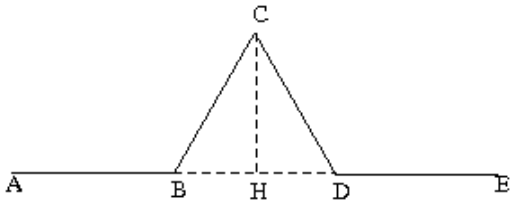
2. (2 pts) Ecrire une fonction *calcul* à un paramètre (un tableau de 3 réels) qui retourne la moyenne calculée à partir des notes du tableau, pour un étudiant donné.

3. (2 pts) A partir du fichier initial et de cette fonction calcul, écrire une procédure permettant d'afficher à l'écran les élèves pour lesquels la moyenne est supérieure ou égale à dix.

Problème n°3 (10 points) : Procédures, fonctions, tableaux et enregistrements

partie I : décomposition d'un segment [A,E] en 4 segments [A,B], [B,C], [C,D], [D,E]

Nous allons dans un premier temps découvrir une décomposition d'un segment de droite initial [A, E]. Le point A est de coordonnées (x_A, y_A) et E de coordonnées (x_E, y_E). Ce segment de droite est coupé en 3 parties égales, [A, B], [B, D] et [D, E]. Nous retirons le segment [B, D]. Nous ajoutons les segments [B, C] et [C, D] tels qu'ils soient les côtés d'un triangle équilatéral. La figure ci-dessous donne le résultat.



Les coordonnées d'un point P quelconque de la droite passant par les points A et E sont données par l'équation paramétrique : P = (1-t).A + t.E où t est un réel.
C'est à dire : $x_P = (1-t) * x_A + t* x_E$ **et** $y_P = (1-t) * y_A + t* y_E$.

- t=0 donne le point A
- t=1 donne le point E
- t=1/3 donne le point B
- t=2/3 donne le point D
- t=1/2 donne le point H, milieu du segment [A, E]

Nous pouvons donc calculer facilement les coordonnées des points B, D et H en connaissant uniquement les coordonnées de A et E.

La projection orthogonale du point C sur le segment [A, E] donne le point H. Soit N= (y_E-y_A, x_A -x_E) un vecteur orthogonal au segment [A, E]. Le point C est obtenu par C = H + ($\sqrt{}$3/6) N.

Question 1 (1 pt) : Définir le type **point** qui contient deux coordonnées réelles.

Question 2 (1 pt) : Ecrire la procédure **calcule_point** qui calcule un point **P** avec **t** et les points **A** et **E.**

Question 3 (1,5 pt) : Ecrire une procédure **calcule_N** qui calcule le vecteur **N** (vu comme un point) à partir de deux points **A** et **E**.

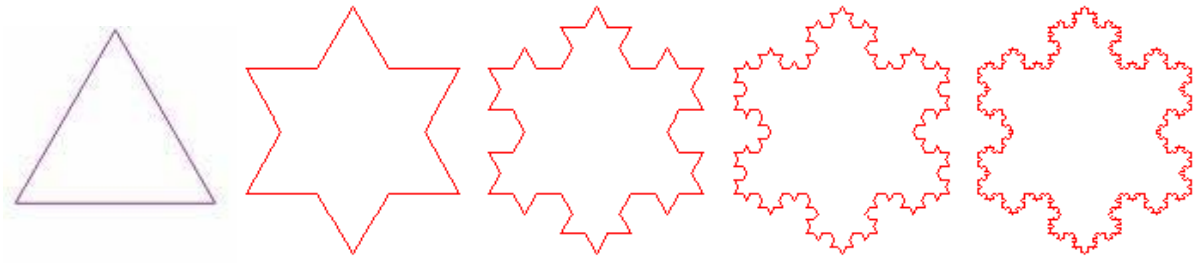
Question 4 (1,5 pt) : Ecrire une procédure **calcule_C** qui détermine le point **C**. Vous utiliserez **obligatoirement** les deux procédures précédentes.

Partie II :

L'objectif est désormais de :

1) calculer les coordonnées géométriques d'un flocon de Koch qui est une fractale.
2) de les afficher à l'écran

La figure ci-dessous illustre notre explication. Le flocon de rang 0 est un triangle équilatéral (à gauche). Ce flocon initial est formé de trois points P1, P2, P3. Initialement nous avons P1 = (0,0), P2 = (1, 0) et P3 = (1/2, ($\sqrt{}$3/2). Nous appliquons la décomposition décrite en partie 1 à chacun des segments [P1, P2] [P2, P3] et [P3, P1]. Nous obtenons ainsi le flocon de rang 1 à droite du triangle. En réappliquant la décomposition aux segments obtenus, nous obtenons successivement les flocons de rang 2, 3 et 4 (de la gauche vers la droite du dessin de la page suivante).



Nous définissons le début du programme :

const MAX_POINT = 10000 ; (*MAX_POINT est le nombre maximum de points *)

```
type floc = record
              rang, nb_point : integer;
              t_point : array[1..MAX_POINT] of point;
            end;
```

Deux points ***consécutifs*** d'indice **i** et **i+1** du tableau **t_point** définissent un segment. Un champ **nb_point** permet de gérer explicitement le nombre de points du contour (3 initialement). Le **rang** du flocon est un entier initialisé à 0. Nous recopions dans la case **nb_point+1** du tableau le point de la case d'indice **1** pour « fermer » le contour. Ainsi tous les segments sont consécutivement codés dans le tableau. La procédure suivante initialise le flocon de rang 0 :

```
procedure init_flocon(var un_floc : floc);
begin
      un_floc.rang := 0;
      un_floc.nb_point := 3;
      with un_floc do
            begin
                  t_point[1].x := 0.0;
                  t_point[1].y := 0.0;
                  t_point[2].x := 1.0;
                  t_point[2].y := 0.0;
                  t_point[3].x := 0.5;
                  t_point[3].y := 0.5*sqrt(3);
                  t_point[4]   := t_point[1];
            end;
end;
```

Question 5 (1,5 pt) : Chaque segment décomposé génère **3 nouveaux points**. Si l'on décompose le segment entre les points indices **i** et **i+1** du tableau, il faut générer 3 places pour 3 nouveaux points **entre** ces deux indices. La fonction suivante "fait" de la place à partir d'un indice **i** et retourne le nouvel emplacement de l'ancien point de numéro i+1 qui est « décalé ».

```
function faire_place(var un_floc : floc; i : integer) : integer;

var k : integer;

begin
  with un_floc do
      begin
            for k := nb_point + 1 downto i+1 do
               begin
                  t_point[k+3]:= t_point[k];
               {donner la valeur de k et dessiner le contenu tableau t_point ici}
               end ;
            nb_point := nb_point+3;
      end;
  faire_place := i+4;
end;
```

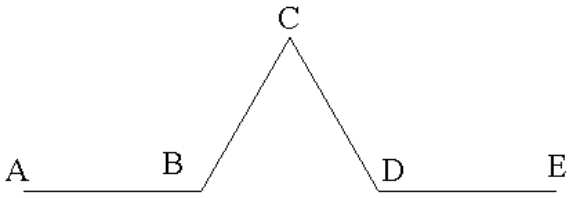
Soit flocon de **rang** 0, avec **nb_point** = 3, et le tableau **t_point** initialisé et schématisé comme ci-dessous :

1	2	3	4	5	6	7	8	9	10	11	12	13
P1	P2	P3	P1									

Faire fonctionner « a mano » la fonction pour i = 1 en suivant l'indication en commentaire dans le code, donner la nouvelle valeur du champ **nb_point** et la valeur retournée par la fonction.

Question 6 (1,5 pt) : Concevoir une fonction **ajoute_points** qui crée et met à jour les nouveaux points dans une variable de type floc pour un segment défini par l'indice i de son premier point. Cette fonction retournera l'indice du premier point du prochain segment à partir duquel de nouveaux points devront être ajoutés. On utilisera **obligatoirement** les variables locales A, B, C, D et E de type point pour que le code soit clair (voir première figure avec le segment décomposé) ainsi que la fonction **`faire_place`** ci-dessus.

Rappel de la figure :



Question 7 (2 pts) : Concevoir une procédure **`floc_r_2_r_plus_1`** qui calcule et met à jour les données quand un flocon passe d'un rang r au rang r+1. Vous utiliserez **obligatoirement** une boucle repeat until.