LIFAP1 – TP6 : Les structures

Objectifs : Manipulation des structures

Petit exercice d'échauffement

Une fleur est définie par

- ✓ son nom,
- ✓ sa couleur,
- ✓ **un tableau** comparatif de son prix chez 3 commerçants sélectionnés.

Une liste de fleurs est définie par

- ✓ le nombre de fleurs qu'elle contient,
- ✓ un tableau de fleurs.

1. Définir en langage C/C++ une constante CHMAX ayant pour valeur 50 qui sera utilisée comme taille maximale des chaines de caractères.
2. Définir en langage C/C++ les structures de données `fleur` et `liste_fleurs` permettant de stocker toutes les informations concernant une fleur et une liste de fleurs.
3. Écrire en langage C/C++ un sous-programme `saisir_fleur` permettant de saisir toutes les informations relatives à une fleur. Attention on recommencera la saisie tant que les données de prix ne sont pas strictement positives.
4. Écrire en langage C/C++ un sous-programme `saisir_liste_fleurs` permettant de saisir toutes les informations relatives à une liste de fleurs. On utilisera pour cela le sous-programme de la question c.
5. Afin d'effectuer des comparaisons tarifaires entre les 3 commerçants sélectionnés, on souhaite connaître pour une liste de fleurs, le prix total de toutes les fleurs chez chacun des 3 commerçants. Écrire en langage C/C++ **un seul** sous-programme `prix_liste_fleurs` permettant d'extraire ces 3 informations. On s'assurera de ne parcourir qu'une seule fois le tableau de fleurs.
6. Écrire en langage C/C++ un sous-programme `affiche_fleurs_rouges` permettant d'afficher le nom de toutes les fleurs de couleur rouge de la liste de fleurs.
7. Écrire en langage C/C++ le programme principal permettant de remplir un tableau avec les caractéristiques de 5 fleurs, d'afficher le nom de toutes les fleurs rouges de cette liste et d'afficher le numéro du commerçant le moins cher des 3 pour cette liste.

```
#include <iostream>
#include <string.h>
using namespace std;
const int CHMAX =50;

struct fleur
{
   char nom[CHMAX];
   char couleur[CHMAX];
   float prix[3];
};
```

```
struct bouquet
{
    int nb;
    struct fleur tab_fleur[15];
};

struct fleur saisir_fleur (void)
{
   struct fleur f;
   int i;

   cout<<"donnez le nom de la fleur"<<endl;
   cin>>f.nom;
   cout<<"donnez la couleur de la fleur"<<endl;
   cin>>f.couleur;
   for (i=0;i<3;i++)
   {
      do
      {
          cout<<"donnez le prix de la fleur chez le commercant "<<i<<endl;
          cin>>f.prix[i];

      }
      while (f.prix[i]<=0);
   }

   return f;
}

void saisir_liste_fleurs (struct bouquet &b)
{
    b.tab_fleur[b.nb]=saisir_fleur();
    b.nb++;
}

void prix_liste (struct bouquet b, float &com1, float &com2, float &com3)
{
    int i;

    com1=0;
    com2=0;
    com3=0;

    for (i=0;i<b.nb;i++)
    {
        com1+=b.tab_fleur[i].prix[0];
        com2+=b.tab_fleur[i].prix[1];
        com3+=b.tab_fleur[i].prix[2];
    }

}

void affiche_fleurs_rouges (struct bouquet b)
{   int i;
    cout<<"Voila la liste des fleurs de couleur rouge : "<<endl;
    for (i=0;i<b.nb;i++)
    {
        if (strcmp(b.tab_fleur[i].couleur,"rouge")==0)
            cout<<b.tab_fleur[i].nom<<endl;
```

```
    }
}
int main (void)
{
    struct bouquet bq;
    float c1,c2,c3;
    bq.nb=0;
    for (int i=0;i<5;i++)
    {
        saisir_liste_fleurs(bq);
    }
    affiche_fleurs_rouges (bq);
    prix_liste(bq,c1,c2,c3);
    cout<<"Prix chez le commercant 1 : "<<c1<<endl;
    cout<<"Prix chez le commercant 2 : "<<c2<<endl;
    cout<<"Prix chez le commercant 3 : "<<c3<<endl;

    return 0;
}
```

Des images en mode texte

1. Représentation des images à l'aide d'une structure
   a. Définissez en langage C la structure *Image* représentant une image de caractères. Voici sa définition en langage algorithmique :

```
Constantes : DIMMAX : Entier = 100

Structure Image
     dimx, dimy : Entier // les dimensions de l'image <= DIMMAX
     im : tableau[DIMMAX][DIMMAX] de caractères
Fin Structure
```

   b. Ecrivez une procédure permettant d'initialiser une image *im* avec le caractère *c* ?
   c. Écrivez la procédure *imAff* qui efface l'écran et affiche l'image
      Attention : il faut afficher l'image de haut en bas car l'écran se remplit de haut en bas!
      Rappel : pour effacer l'écran l'instruction est `system("cls");`

2. Dessins (boucle avec bornes évoluées)
   a. Écrivez la procédure *imDessineRect* qui remplit un rectangle avec le caractère *c* dans l'image. Cette procédure n'affiche pas l'image !

```
// En algorithmique
Procédure imDessineRect(im : donnée-résultat Image; c : donnée
Caractère; xmin, ymin, xmax, ymax : donnée Entier)
```

      Remarque : L'image n'est pas initialisée entre chaque dessin. Deux figures tracées à la suite dans une image vont se superposer.
   b. Écrivez la procédure `imDessineCercle` qui dessine un cercle plein dans l'image.

```
// En algorithmique
Procédure imDessineCercle(im : donnée-résultat Image; c : donnée
Caractère; x,y,r : donnée Entier)
```

      Version simple de l'algorithme : parcourez toutes les cases de l'image, si la case considérée a une distance au centre du cercle inférieure au rayon, la case est dans le cercle.

```
#include <iostream>
#include <math.h> // Pour sqrt();
using namespace std;

// Taille maximum des tableaux (variable globale constante)
const int DIMMAX=100;

// Structure d'une image de caracteres
struct Image
{
  int dimx,dimy;
  char tab_im[DIMMAX][DIMMAX];
};


// Initialisation des elements d'une image
void ImInit(struct Image &im, char c)
{
  int x,y;
  for(y=0;y<im.dimy;y++)
  {
    for(x=0;x<im.dimx;x++)
    {
      im.tab_im[x][y]=c;
    }
  }
}


// Affichage d'une image de caracteres sur la sortie standard
void ImAff(struct Image im)
{
  int x,y;
  for(y=0;y<im.dimy;y++)
  {
    for(x=0;x<im.dimx;x++)
    {
      cout<<im.tab_im[x][y];
    }
    cout<<endl;
  }
}


// Ecriture d'un rectangle dont les coordonnées des angles sont (xmin,ymin) et //
(xmax,ymax), avec le caractere c
void ImDessineRect(struct Image &im, char c, int xmin, int ymin, int xmax, int ymax)
{
// Controle d'erreurs
  if(xmin<0 || xmin>=im.dimx || xmax<0 || xmax>=im.dimx || ymin<0 || ymin>=im.dimy ||
ymax<0 || ymax>=im.dimy)
  {
    cout<<"**ERREUR**: Impossible d'ecrire un rectangle a ces coordonnees\n";
  } else
  {
  // Detection du "sens" du rectangle
    int stepx = (xmin<xmax)?1:-1;
    int stepy = (ymin<ymax)?1:-1;

    int x,y;
```

```
   // Les "horizontales"
      for(x=xmin; x!=(xmax+stepx); x+=stepx)
      {
         im.tab_im[x][ymin]=c;
         im.tab_im[x][ymax]=c;
      }

   // Les "verticales" (on évite de refaire les coins)
      for(y=ymin+stepy; y!=ymax; y+=stepy)
      {
         im.tab_im[xmin][y]=c;
         im.tab_im[xmax][y]=c;
      }
   }
}

// Ecriture d'un cercle de centre (x0,y0), de rayon r, avec le caractere c
void ImDessineCercle(struct Image &im, char c, int x0, int y0, int r)
{
//Controle d'erreurs
   if( r<0 || (x0-r<0) || (x0+r>=im.dimx) || (y0-r<0) || (y0+r>=im.dimy))
   {
      cout<<"**ERREUR**: Impossible d'ecrire un cercle a ces coordonnees\n";
   }else
   {
      int x,y;
      for(x=x0-r;x<=x0+r;x++)
      {
         y = (int)sqrt((float)(r*r - (x0-x)*(x0-x)));
         im.tab_im[x][y0+y] = c;
         im.tab_im[x][y0-y] = c;
      }
   }
}

// Ecriture d'un cercle de centre (x0,y0), de rayon r, avec le caractere c
void ImDessineCercle(struct Image &im, char c, int x0, int y0, int r)
{
//Controle d'erreurs
   if( r<0 || (x0-r<0) || (x0+r>=im.dimx) || (y0-r<0) || (y0+r>=im.dimy))
   {
      cout<<"**ERREUR**: Impossible d'ecrire un cercle a ces coordonnees\n";
   }else
   {
      int x,y;
      for(x=x0-r;x<=x0+r;x++)
      {
         y = (int)sqrt((float)(r*r - (x0-x)*(x0-x)));
         im.tab_im[x][y0+y] = c;
         im.tab_im[x][y0-y] = c;
      }
   }
}
int main(void)
{
   struct Image im;
   im.dimx      = 10;
   im.dimy      = 10;
   char plein   = 'o';
   char vide    = '.';
```

```
    cout<<"Bienvenue dans le TP6!\n";
    cout<<"L'image de travail est de dimension: "<<im.dimx<<"x"<<im.dimy<<endl;

    cout<<"\nInitialisation de l'image avec le caractere: "<<vide<<endl;
    ImInit(im,vide);

    cout<<"\nAffichage de l'image\n";
    ImAff(im);

    cout<<"\nTrace d'un rectangle de (2,8) a (7,1), avec le caractere: "<<plein<<endl;
    ImDessineRect(im,plein,2,8,7,1);

    cout<<"\nAffichage de l'image\n";
    ImAff(im);

    cout<<"\nEffacement de l'image avec le caractere: "<<vide<<endl;
    ImInit(im,vide);

    cout<<"\nTrace d'un Cercle de centre (5,5) et de rayon 4, avec le caractere x\n";
    ImDessineCercle(im,'x',5,5,4);

    cout<<"\nAffichage de l'image\n";
    ImAff(im);

    cout<<"\nFin\n";
    return 0;
}
```