với nhau để tạo thành file ban đầu bằng lệnh join. Hãy viết chương trình tách một file thành n file và nối các file đã tách thành file ban đầu.

CHƯƠNG 9: ĐỒ HỌA

9.1. MÀN HÌNH TRONG CHẾ ĐỘ ĐỒ HỌA

Hình ảnh trong chế độ đồ họa (Graphic) được tạo ra bằng các điểm ảnh (Pixel), số điểm ảnh của màn hình đồ họa tùy thuộc vào từng loại CARD màn hình và MODE qui định cho màn hình đó.

Việc lập trình trong chế độ đồ họa cần phải xác định được loại màn hình đang sử dụng và chương trình phải vận hành được trên nhiều loại màn hình khác nhau.

Tọa độ của một điểm ảnh trên màn hình đồ họa cũng giống như trong chế độ văn bản (TEXT) với điểm ảnh đầu tiên trên góc trái màn hình là (0,0), tọa độ đỉnh dưới phải tùy thuộc vào độ phân giải của màn hình, CARD màn hình và MODE màn hình.

Để dử dụng được chế độ đồ họa trên màn hình, ta cần phải có các File sau:

<**graphics.h**>	Chứa các lệnh đồ họa
***.BGI**	Chứa Font màn hình
***.CHR**	Chứa Font ký tư

9.2. KHỞI TẠO VÀ THOÁT KHỎI CHẾ ĐỘ ĐỒ HỌA

9.2.1. Khởi tạo chế độ đồ họa

```
void initgraph(int *gd,int *gm,char *Path);
```

trong đó:

- **gd**: Chỉ CARD màn hình.

Thông thường, một chương trình phải được chạy trên nhiều loại màn hình khác nhau nên ta có thể khai báo:

gd = DETECT (= 0)

Với hằng DETECT, máy sẽ tự động tìm CARD màn hình tương ứng để chạy chương trình.

- **gm**: Chỉ MODE màn hình.

Trong trường hợp khai báo gd = DETECT thì không cần thiết phải khai báo gm vì máy tính sẽ tự xác định loại CARD màn hình và thiết lập chế độ MODE màn hình tương ứng với CARD màn hình đó.

- **Path**: Đường dẫn đến nơi chứa các file *.BGI. Nếu Path = " " thì ta hiểu là các file *.BGI nằm trong thư mục hiện hành.

9.2.2. Thoát khỏi chế độ đồ họa

`void closegraph();`

Sau đây là cấu trúc chung của một chương trình đồ họa:.

```
#include <graphics.h>
void ThietLapDoHoa()
{
 int gd=0,gm;
 initGraph(&gd,&gm,"D:\\TC\\BGI");
}
void main()
{
  ThietLapDoHoa();
   . . .
  <Các lệnh vẽ>;
  …
  closegraph();
}
```

9.3. TỌA ĐỘ VÀ CON TRỎ TRÊN MÀN HÌNH ĐỒ HỌA

9.3.1. Lấy kích thước màn hình

`int getmaxx();` và **`int getmaxy();`**

Cho tọa độ cột lớn nhất và dòng lớn nhất của màn hình.

(0,0)

(getmaxx(),getmaxy())

9.3.2. Di chuyển con trỏ

`void moveto(int x,int y);`

Di chuyển con trỏ đến tọa độ (x,y).

`void moverel(int dx,int dy);`

Di chuyển con trỏ đến tọa độ mới cách tọa độ cũ khoảng cách là dx, dy.

9.3.3. Vẽ điểm

```
void putpixel(int x,int y,int color);
```

Vẽ một điểm tại toạn độ (x,y) với màu là color.

```
unsigned getpixel(int x,int y);
```

Lấy màu của một điểm tại tọa độ (x,y).

9.4. ĐẶT MÀU TRÊN MÀN HÌNH ĐỒ HỌA

9.4.1. Đặt màu cho đối tượng cần vẽ

```
void setcolor(int color);
```

9.4.2. Đặt màu nền

```
void setbkcolor(int color);
```

9.5. CỬA SỔ TRONG CHẾ ĐỘ ĐỒ HỌA

9.5.1. Đặt cửa sổ trên màn hình

```
void setviewport(int x1,int y1,int x2,int y2;int Clip);
```

trong đó:

(x1,y1): đỉnh trên trái của cửa sổ.

(x2,y2): đỉnh dưới phải của cửa sổ.

Nếu Clip ≠ 0 thì những gì vượt khỏi màn hình sẽ bị cắt bỏ.

Chú ý: Khi tạo cửa sổ thì tọa độ trên màn hình sẽ thay đổi theo.

Tọa độ mới = Tọa độ cũ - Tọa độ đỉnh trên trái.

9.5.2. Xóa hình ảnh trong cửa sổ

- Xóa hình ảnh trong cửa sổ, ta dùng hàm **clearviewport();**
- Xóa toàn bộ màn hình, ta dùng hàm **cleardevice();**

9.6. VIẾT CHỮ TRONG ĐỒ HỌA

9.6.1. Thiết lập font chữ

```
void settextstyle(int font,int dir,int size);
```

- Các font có thể chứa các hằng sau:

  DEFAULT_FONT = 0; TRIPLEX_FONT = 1; SMALL_FONT = 2;

  SANS_SERIF_FONT = 3; GOTHIC_FONT = 4;

- Dir có các hằng sau:

  HORIZ_DIR = 0 Từ trái qua phải.

VERT_DIR = 1 Từ dưới lên trên.

- size: độ lớn của chữ.

9.6.2. Thiết lập phân bố chữ

```
void settextjustify(int Hz,int Vt);
```

Chọn vị trí của chữ xung quanh tọa độ định sẵn.

- Hz là phân bố chữ theo trục ngang. Có các hằng sau:

LEFT_TEXT = 0 Chữ viết nằm bên phải trục đứng.

CENTER_TEXT = 1 Chữ viết nằm ở giữa trục đứng.

RIGHT_TEXT = 2 Chữ viết nằm bên trái trục đứng.

- Vt là bố trí chữ theo hướng dọc đối với tọa độ qui định xuất chuỗi. Các hằng liên quan:

BOTTOM_TEXT = 0 Chữ viết nằm bên trên trục ngang.

CENTER_TEXT = 1 Chữ viết nằm ở giữa trục ngang.

TOP_TEXT = 2 Chữ viết nằm bên dưới trục ngang.

9.6.3. Viết một xâu ký tự lên màn hình

- Xuất một xâu ký tự tại vị trí con trỏ:

Dùng hàm **`void outtext(char *st);`**

- Xuất một xâu ký tự tại tọa độ x,y:

Dùng hàm **`void outtextxy(int x,int y,char *st);`**

Chú ý: Cách xuất chuỗi của hai hàm trên được qui định trong các hàm **`settextstyle`** và **`settextjustify`**.

9.7. VẼ CÁC HÌNH CƠ BẢN

9.7.1. Chọn kiểu đường

```
void setlinestyle(int Ls,int Pt,int Tk);
```

Ls: kiểu đường vẽ, có các giá trị sau:

0: Đường liền nét

1: Nét đứt

2: Nét chấm gạch

3: Nét gạch

4: Đường do người thiết kế tạo ra.

Pt: xác định màu vẽ.

. Nếu Ls = 0..3 thì Pt=0 (Lấy giá trị Default)

. Nếu Ls = 4 thì Pt là số nguyên chỉ màu của kiểu đường.

Tk: xác định độ dày của đường.

Tk = 1: bình thường.

Tk = 3: đậm nét.

9.7.2. Vẽ đoạn thẳng

`void line(int x1,int y1,int x2,int y2);` vẽ từ điểm (x1,y1) đến điểm (x2,y2).

`void lineto(int x,int y);` vẽ từ vị trí con trỏ đến điểm (x,y).

`void linerel(int dx,int dy);` vẽ từ vị trí con trỏ đến điểm cách nó một khoảng dx,dy.

Ví dụ 1: Vẽ đồ thị hàm số: $f(x) = ax^2 + bx + c$ trên đoạn [-10,10].

Ý tưởng:

Bước 1: Xác định đoạn cần vẽ [Min,Max].

Bước 2: Đặt gốc tọa độ lên màn hình (x0,y0).

Chia tỉ lệ vẽ trên màn hình theo hệ sô k.

Chọn số gia dx trên đoạn cần vẽ.

Bước 3: Chọn điểm xuất phát: x = Min, tính f(x).

Đổi qua tọa độ màn hình và làm tròn:

x1=x0 + Round(x*k);

y1=y0 - Round(f(x)*k);

Di chuyển đến (x1,y1): moveto(x1,y1);

Bước 4: Tăng x lên: x=x + dx;

Đổi qua tọa độ màn hình và làm tròn:

x2=x0 + Round(x*k);

y2=y0 - Round(f(x)*k);

Vẽ đến (x2,y2): lineto(x2,y2);

Bước 5: Lặp lại bước 4 cho đến khi x ≥ Max thì dừng.

```
#include<conio.h>
#include<graphics.h>

float a,b,c,d,min,max;
```

```
int round(float x)
{
  if (x>0) return int (x+0.5);
  else  return int (x-0.5);
}

void khoitaodohoa()
{
  int gd=0,gm=0;
  initgraph(&gd,&gm,"d:\\tc\\bgi");
}

float f(float x)
{
  return(a*x*x*x+b*x*x+c*x+d);
}

void vedothi(float min,float max)
{
  int x0,y0,x1,y1,x2,y2;
  float x,dx,k;
  x0=getmaxx()/2;
  y0=getmaxy()/2;
  k=(float)getmaxx()/50;
  dx=0.001;
  setcolor(12);
  //Vẽ trục tọa độ
  line(0,y0,2*x0,y0);
  line(x0,0,x0,2*y0);
  //Vẽ đồ thị
  x=min;
  setcolor(14);
  x1=x0+round(x*k);
```

```
  y1=y0-round(f(x)*k);
  moveto(x1,y1);
  while (x<max)
  {
    x=x+dx;
    x2=x0+round(x*k);
    y2=y0-round(f(x)*k);
    lineto(x2,y2);
  }
}

void main()
{
  khoitaodohoa();
  min=-10;max=10;
  a=1;b=-1;c=-1;d=2;
  vedothi(min,max);
  getch();
  closegraph();
}
```

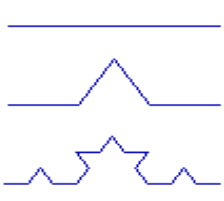
Ví dụ 2: Viết chương trình vẽ cung Koch. Các bước phát sinh của cung Koch được thực hiện trong hình sau:

- Bắt đầu từ đường ngang K_0 có độ dài bằng 1.
- Để tạo cung bậc-1(gọi là K_1), chia đường thành ba phần và thay đoạn giữa bằng tam giác đều có cạnh dài 1/3. Bây giờ, toàn bộ đường cong có độ dài 4/3.
- Cung bậc-2 K_2 có được bằng cánh dựng tiếp các tam giác đều từ 4 đoạn của K1. Vì mỗi đoạn có độ dài tăng 4/3 lần nên toàn bộ cung dài ra 4/3 lần.

(a) K_0

(b) K_1

(c) K_2



Ý tưởng:

Từ hình (b) ta thấy rằng, đầu tiên hướng vẽ quay trái 60^0, rồi quay phải 120^0, cuối cùng quay trái 60^0 để trở về hướng ban đầu.

```
#include <graphics.h>
#include <conio.h>
```

```
#include <math.h>

#define PI 3.1416

void ThietLapDoHoa()
{
 int gd=0,gm;
 initgraph(&gd,&gm,"D:\\TC\\bgi");
}

int Round(float x)
{
 if (x>0) return int (x+0.5);
 else return int (x-0.5);
}

void Koch(float dir,float len,int n)
{
 if(n>0)
 {
   Koch(dir,len/3,n-1);
   dir=dir+60; //Quay phai 60 do
   Koch(dir,len/3,n-1);
   dir=dir-120; //Quay trai 120 do
   Koch(dir,len/3,n-1);
   dir=dir+60; //Quay phai 60 do
   Koch(dir,len/3,n-1);
  }
  else
linerel(Round(len*cos(dir*PI/180)),Round(len*sin(dir*PI/180)));
}

void main()
{
```

```
  float Goc=180,Length=250;
  int n=4;
  ThietLapDoHoa();
  moveto(350,200);
  Koch(Goc,Length,n);
  getch();
  closegraph();
}
```

9.7.3. Vẽ hình chữ nhật

`void rectangle(int x1,int y1,int x2,int y2);`

Vẽ hình chữ nhật với đỉnh trên trái là (x1,y1) và đỉnh dưới phải là (x2,y2).

Ví dụ: Vẽ các hình chữ nhật ngẫu nhiên trên màn hình.

```
#include <graphics.h>
#include <stdlib.h>
#include <conio.h>
#include <dos.h>

void ThietLapDoHoa()
{
 int gd = DETECT, gm;
 initgraph(&gd, &gm, "d:\\tc\\bgi");
}

void Demo()
{
 int x1,y1,x2,y2;
 randomize();
 do
 {
   x1=random(getmaxx());
   y1=random(getmaxy());
   x2=random(getmaxx()-x1)+x1;
   y2=random(getmaxy()-y1)+y1;
```

```
   setcolor(random(14)+1);
   rectangle(x1,y1,x2,y2);
   delay(100);
 }
 while (kbhit()==0);
}

void main()
{
   ThietLapDoHoa();
   Demo();
   getch();
   closegraph();
}
```

9.7.4. Vẽ cung tròn

`void arc(int x,int y,int g1,int g2,int R);`

Vẽ cung tròn có tâm (x,y) bán kính R, góc bắt đầu là g1 và góc kết thúc là g2.

9.7.5. Vẽ đường tròn - Ellipse

Vẽ đường tròn: **`void circle(int x,int y,int R);`**

Vẽ ellipse: **`void ellipse(x,y:integer; g1,g2,Rx,Ry:Word);`**

Vẽ Ellipse có tâm (x,y) bán kính ngang Rx, bán kính dọc Ry, góc bắt đầu là g1 và góc kết thúc là g2.

9.7.6. Định MODE vẽ cho đoạn thẳng

`void setwritemode(int Mode);`

Ta có thể chọn Mode bằng các hằng sau:

. COPY_PUT = 0; đây là Mode chèn, đường mới sẽ không xóa đường cũ.

. XOR_PUT = 1; đây là Mode xóa, đường mới sẽ xóa đường cũ.

Ví dụ: Vẽ một kim đồng hồ quay quanh tâm O(x0,y0).

```
#include <graphics.h>
#include <conio.h>
#include <math.h>
#include <dos.h>
```

```
#define PI 3.1416

void ThietLapDoHoa()
{
 int gd = DETECT, gm;
 initgraph(&gd, &gm, "d:\\tc\\bgi");
}

int Round(float x)
{
 if(x>0) return int(x+0.5);
 else return int(x-0.5);
}

void VeKim(int x0,int y0,float R,float Alpha)
{
 line(x0,y0,x0+Round(R*cos(PI*Alpha/180)),
      y0-Round(R*sin(PI*Alpha/180)));
}

void Demo()
{
 int x0=getmaxx()/2;
 int y0=getmaxy()/2;
 int R=100;
 float Alpha=90, Beta=6;
 setwritemode(XOR_PUT);
 VeKim(x0,y0,R,Alpha);
 do
 {
  delay(500);
  VeKim(x0,y0,R,Alpha);
```

```
  Alpha=Alpha-Beta;
  VeKim(x0,y0,R,Alpha);
 }
 while (kbhit()==0);
}

void main()
{
 ThietLapDoHoa();
 Demo();
 getch();
 closegraph();
}
```

9.8. TÔ MÀU CÁC HÌNH

9.8.1. Chọn kiểu tô

```
void setfillstyle(int Pt,int Color);
```

Với:

- Pt: Mẫu tô của hình. Có các hằng từ 0 đến 12.
  - 0: Tô bằng màu nền.
  - 1: Tô bằng màu viền.
  - 2: Tô bằng các dấu ---
  - .....
- Color: Màu tô của hình.

9.8.2. Vẽ hình chữ nhật có tô màu ở bên trong

```
void bar(int x1,int y1,int x2,int y2);
```

Vẽ hình chữ nhật có tô màu và mẫu tô được xác định bởi hàm setfillstyle.

Ví dụ: Viết chương trình tạo Menu cho phép chọn và thực hiện các chức năng bằng cách di chuyển mũi tên trên các hộp sáng, các thủ tục thực hiện xong quay trở lại Menu chính. Nhấn ESC để thoát khỏi chương trình.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
```

```
#include<graphics.h>

#define TRUE 0
#define FALSE 1

const  mau1=15;
const  mau2=8;
const  maumn=7;
const  mauchu=1;
const  XTop=180;
const  YTop=100;
const  Dy=32;
const  Dx=270;

void ThietLapDoHoa()
{
 int gd=0,gm;
 initgraph(&gd,&gm,"d:\\tc\\bgi");
}

void Baitap1()
{
  cleardevice();
  outtext("\nBAI TAP TINH TONG CAC CHU SO CUA N");
  getch();
}

void Baitap2()
{
  cleardevice();
  outtext("\nBAI TAP VE MANG MOT CHIEU\n");
  getch();
}
```

```
void Baitap3()
{
  cleardevice();
  outtext("\nXin loi! Bai tap nay tui chua lam!");
  getch();
}

void Baitap4()
{
  cleardevice();
  outtext("\nBAI TAP TIM UOC CHUNG LON NHAT");
  getch();
}

//Vẽ nút menu
void Box(int x1,int y1,int x2,int y2,
         char MauVienTren,char MauVienDuoi,char MauNen)
{
  int i;
  setfillstyle(1,MauNen);
  bar(x1,y1,x2,y2);
  setcolor(MauVienTren);
  for(i=0;i<=1;i++)
  {
   moveto(x1-i,y2+i);
   lineto(x1-i,y1-i);
   lineto(x2+i,y1-i);
  }
  setcolor(MauVienDuoi);
  for(i=0;i<=1;i++)
  {
   moveto(x2+i,y1-i);
```

```
   lineto(x2+i,y2+i);
   lineto(x1-i,y2+i);
  }
}

//Viết xâu st ra màn hình tại dòng thứ n voi mau la color
void Write(char *st,int n,int color)
{
  setcolor(color);
  outtextxy(XTop+20,YTop+15+n*Dy,st);
}

void Ve_menu(int Xdau,int Ydau,int chon,int SoDong,
                char *DongMN[])
{
  int i;
  cleardevice();
  for(i=0;i<SoDong;i++)
  {
    if(i==chon)
      Box(Xdau,Ydau+i*Dy+6,Xdau+Dx,Ydau+i*Dy+Dy,
            mau2,mau1,maumn);
    else
      Box(Xdau,Ydau+i*Dy+6,Xdau+Dx,Ydau+i*Dy+Dy,
            mau1,mau2,maumn);
    Write(DongMN[i],i,mauchu);
  }
}

void main()
{
  char ch,*st[10];
  st[0]="Bai tap tinh tong cac chu so";
```

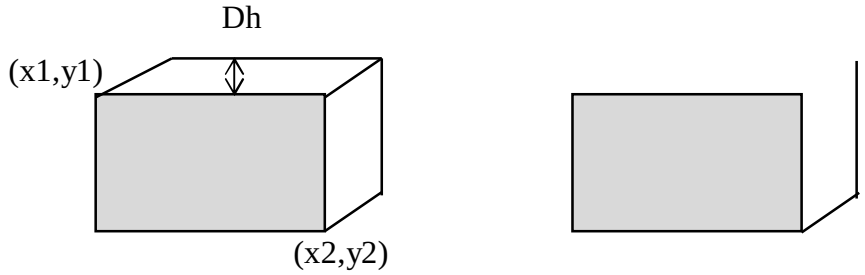
```
st[1]="Bai tap ve mang mot chieu";
st[2]="Bai tap thiet ke ham De quy";
st[3]="Bai tap tim Uoc chung lon nhat";
st[4]="<ESC> Ket thuc chuong trinh!";
int chon=0,luuchon,sodong=5,ok=FALSE;
ThietLapDoHoa();
Ve_menu(XTop,YTop,chon,sodong,st);
do
{
  ch=getch(); //Nhan mot phim
  switch (ch)
  {
    case 72: //phim len
            luuchon=chon;
            chon--;
            if(chon<0) chon=sodong-1;
            Box(XTop,YTop+luuchon*Dy+6,XTop+Dx,
                YTop+luuchon*Dy+Dy,mau1,mau2,maumn);
            Write(st[luuchon],luuchon,mauchu);
            Box(XTop,YTop+chon*Dy+6,XTop+Dx,
                YTop+chon*Dy+Dy,mau2,mau1,maumn);
            Write(st[chon],chon,mauchu);
            break;
    case 80://phim xuong
            luuchon=chon;
            chon++;
            if(chon==sodong) chon=0;
            Box(XTop,YTop+luuchon*Dy+6,XTop+Dx,
                YTop+luuchon*Dy+Dy,mau1,mau2,maumn);
            Write(st[luuchon],luuchon,mauchu);
            Box(XTop,YTop+chon*Dy+6,XTop+Dx,
                YTop+chon*Dy+Dy,mau2,mau1,maumn);
            Write(st[chon],chon,mauchu);
```

```
            break;
      case 13: //phim ENTER
            ok=TRUE; break;
  }
  if (ok==TRUE) //Neu phim ENTER duoc nhan
  {
   switch (chon)
    {
      case 0:
           Baitap1();
           Ve_menu(XTop,YTop,chon,sodong,st);
           break;
      case 1:
           Baitap2();
           Ve_menu(XTop,YTop,chon,sodong,st);
           break;
      case 2:
           Baitap3();
           Ve_menu(XTop,YTop,chon,sodong,st);
           break;
      case 3:
           Baitap4();
           Ve_menu(XTop,YTop,chon,sodong,st);
           break;
      case 4: exit(0);
    }
   ok=FALSE; //tra lai trang thai ENTER chua duoc nhan
  }
 }
 while (ch!=27);//Nhan phim ESC de thoat
 closegraph();
}
```

9.8.3. Vẽ hình hộp chữ nhật

```
void bar3d(int x1,int y1,int x2,int y2,int Dh,int Top);
```

Vẽ hình hộp chữ nhật có tọa độ đỉnh trên là (x1,y1), đỉnh dưới là (x2,y2) và bề dày là Dh.



Có nắp Không có nắp

Top ≠ 0: Hình hộp có nắp.

Top = 0: Hình hộp không có nắp.

9.8.4.Vẽ và tô màu Ellipse

```
void fillellipse(int x,int y,int Rx,int Ry);
```

Vẽ hình và tô màu cho ellipse có tâm (x,y), bán kính theo 2 trục là Rx và Ry với màu và mẫu tô được xác định bởi hàm setfillstyle.

9.8.5. Vẽ hình quạt tròn

```
void pieslice(int x,int y,int g1,int g2,int R);
```

Vẽ hình quạt tròn có tâm (x,y), góc đầu g1, góc cuối g2, bán kính R.

9.8.6. Vẽ hình quạt Ellipse

```
void sector(int x,int y,int g1,int g2,int Rx,int Ry);
```

9.8.7. Làm loang màu một vùng kín

```
void floodfill(int x,int y,int Color);
```

Trong đó:

(x,y): điểm nằm trong vùng kín.

Color: màu muốn tô.

9.8.8. Vẽ đa giác

Đối với một đa giác bất kỳ có N đỉnh, ta phải khai báo N+1 đỉnh để vẽ đường gấp khúc với tọa độ điểm đầu trùng với tọa độ điểm cuối. Để vẽ đa giác, ta dùng hàm:

```
void drawpoly(int N,int P[]);
```

trong đó:

- N: số đỉnh của đa giác + 1
- P: chứa tọa độ các đỉnh, cứ hai phần tử liên tiếp trong mảng sẽ tạo thành một điểm (x,y).

Ví dụ: Viết chương trình để vẽ đa giác đều có n đỉnh.

Ý tưởng:

Khi vẽ một đa giác đều N đỉnh, các đỉnh này nằm trên một đường tròn tâm O bán kính R đồng thời khoảng cách giữa hai đỉnh và tâm tạo thành một góc không đổi là 2*Pi/N.

Giả sử đỉnh thứ nhất của đa giác nằm trên đường thẳng tạo với tâm một góc 0^0, đỉnh thứ hai tạo một góc 2*Pi/N và đỉnh thứ i sẽ tạo một góc là 2*Pi*(i-1)/N.

Giả sử P0 là tọa độ tâm của đa giác, đỉnh thứ i của đa giác sẽ tạo một góc là:

Angle=2*Pi*(i-1)/N

Nhưng nếu đa giác này có đỉnh đầu tiên tạo một góc bằng A0 thì:

Angle:=2*Pi*((i-1)/N + A0/360)

Và tọa độ các đỉnh này trên màn hình sẽ là:

P[i].x = P0.x + R*cos(Angle)

P[i].y = P0.y - R*sin(Angle)

Ta xây dựng một hàm để tự động lưu các đỉnh của đa giác đều vào mảng P. Trong đó: P0 là tọa độ tâm, A0 là góc bắt đầu, R là bán kính, N là số đỉnh của đa giác đều (N≥3).

```
#include <graphics.h>
#include <conio.h>
#include <math.h>

#define Max 10
#define PI 3.1416

struct ToaDo
{
  int x,y;
};

void ThietLapDoHoa()
{
 int gd=0,gm;
```

```
 initgraph(&gd,&gm,"D:\\TC\\bgi");
}

int Round(float x)
{
 if (x>0) return int (x+0.5);
 else return int (x-0.5);
}

void TaoDinh(float R,float A0,int n,ToaDo P0,int P[])
{
 for(int i=0;i<n;i++)
 {
   float Angle=2*PI*(float(i)/n + A0/360);
   P[2*i]  =P0.x + Round(R*cos(Angle));
   P[2*i+1]=P0.y - Round(R*sin(Angle));
 }
 P[2*n]=P[0];
 P[2*n+1]=P[1];
}

void main()
{
 ThietLapDoHoa();
 int n=5, P[Max];
 ToaDo P0;
 P0.x=getmaxx()/2;
 P0.y=getmaxy()/2;
 float A0=90,R=getmaxy()/4;
 TaoDinh(R,A0,n,P0,P);
 drawpoly(n+1,P);
 getch();
 closegraph();
```

```
}
```

9.9. CÁC KỸ THUẬT TẠO HÌNH CHUYỂN ĐỘNG

9.9.1. Kỹ thuật lật trang màn hình

CARD màn hình có nhiều trang, mỗi trang được đánh số 0,1,2,...

Để vẽ hình lên một ***trang màn hình*** (trong vùng đệm), ta dùng hàm:

```
void setactivepage(int Page);
```

Trong đó, Page là số của trang màn hình. Hàm này được đặt trước khi có lệnh vẽ ra màn hình.

Để đưa trang màn hình ra màn hình, ta dùng hàm:

```
void setvisualpage(int Page);
```

Page: trang màn hình muốn xem.

Thông thường, màn hình sẽ làm việc và hiện ra trên trang 0. Do đó, để vừa xem màn hình vừa vẽ lên trang màn hình khác, ta thường dùng hai hàm trên đi kèm với nhau.

Để thực hiện một cách tự động khi sử dụng kỹ thuật lật hình này, ta thường theo một giải thuật sau:

- Tạo 2 biến: `int page1,page2;`
- Tạo vòng lặp:

```
...
do
{
  setvisualpage(page1);   // Xem trang màn hình page1
  setactivepage(page2);   // Vẽ hình lên trang page2
  .......
  < Các lệnh vẽ hình lên trang page2>;
  .......
  // Hoán vị 2 biến page1, page2
  Swap(page1,page2);
}
while <Điều kiện lặp>;
```

Ví dụ 1: Vẽ hai hình

Sau đó, viết chương trình thực hiện chuyển động của miệng cá.

```
#include <graphics.h>
#include <conio.h>
#include <dos.h>

int Xc,Yc,R;

void VeHinhCa1()
{
   setcolor(15);
   pieslice(Xc,Yc,30,330,R); //Vẽ bụng cá
   setcolor(0);
   circle(Xc + R/2,Yc - R/2,4); //Vẽ mắt cá
}

void VeHinhCa2()
{
   setcolor(15);
   pieslice(Xc,Yc,15,345,R); //Vẽ bụng cá
   setcolor(0);
   circle(Xc + R/2,Yc - R/2,4); //Vẽ mắt cá
}

void main()
{
  int gd=5,gm;
  initgraph(&gd,&gm,"D:\\TC\\BGI");
  Xc=getmaxx()/2;
  Yc=getmaxy()/2;
  R=50;
```

```
  int i=0;
  int page1=0, page2=1;
  do
  {
    setvisualpage(page1); //xem trang page1
    setactivepage(page2); //ve len trang page2
    i=1-i;
    if(i==0) VeHinhCa1();
    else VeHinhCa2();
    delay(200);
    //hoan doi 2 trang
    page1=1-page1;
    page2=1-page2;
  }
  while (kbhit()==0);
  closegraph();
}
```

Ví dụ 2: Viết chương trình tạo một dòng chữ chạy ngang qua màn hình.

```
#include <graphics.h>
#include <conio.h>
#include <dos.h>

void Run(char *s)
{
  int page=0;
  int x=getmaxx(),y=getmaxy()/3;
  settextjustify(0,1);
  setwritemode(XOR_PUT);
  setactivepage(page);
  do
  {
      outtextxy(x,y,s);
      setvisualpage(page);
```

```
      page=1-page;
      setactivepage(page);
     // delay(10);
      outtextxy(x+1,y,s);
      x=x-1;
      if(x<-textwidth(s)) x=getmaxx();
   }
   while (kbhit()==0);
}

void main()
{
  int gd=5,gm;
  initgraph(&gd,&gm,"D:\\TC\\bgi");
  setcolor(14);
  settextstyle(1,0,5);
  Run("Pham Anh Phuong");
  closegraph();
}
```

9.9.2. Lưu và di chuyển một vùng màn hình

Ta có thể lưu một vùng màn hình vào bộ nhớ rồi sau đó dán nó lên màn hình ở một vị trí khác.

Việc lưu một vùng màn hình vào bộ nhớ được thực hiện bằng hàm:

```
getimage(x1,y1,x2,y2:Integer; void far *Pointer);
```

trong đó Pointer là con trỏ trỏ đến vùng lưu nội dung của vùng (x1,y1,x2,y2).

Việc đăng ký một vùng nhớ động phải được khai báo dung lượng cần thiết. Dung lượng vùng nhớ được thực hiện bằng hàm:

```
unsigned imagesize(int x1,int y1,int x2,int y2);
```

Để hiện hình ảnh từ bộ nhớ ra màn hình, ta dùng hàm:

```
void putimage(int x,int y,void far *Pointer,int  Mode);
```

trong đó:

(x,y): Tọa độ đỉnh trái hình chữ nhật mà ta muốn đưa ra.

Pointer: Con trỏ trỏ đến vùng lưu hình chữ nhật.

Mode: Hằng số chỉ phương thức hiện ra màn hình. Mode chứa một trong các hằng sau:

COPY_PUT = 0: Xuất ra như đã lưu (phép MOV)

XOR_PUT = 1: Phép XOR, xóa hình cũ nếu hai hình giao nhau.

OR_PUT = 2: Phép OR, lấy cả hai hình nếu hai hình giao nhau.

AND_PUT = 3: Phép AND, nếu hai hình giao nhau thì lấy phần chung.

NOT_PUT = 4: Phép NOT, cho ra âm bản.

Về việc thực hiện được tiến hành như sau:

- Khai báo một biến con trỏ P;
- Đăng ký một vùng nhớ động do P quản lý bằng hàm
  `farmalloc(imagesize(x1,y1,x2,y2));`
- Lưu hình ảnh bằng hàm `getimage(x1,y1,x2,y2,P);`
- Xuất ra màn hình bằng hàm `putimage(x,y,P,Mode);`
- Xóa vùng nhớ động bằng hàm `farfree(size);`

Ví dụ: Viết chương trình vẽ chiếc đĩa bay chuyển động ngẫu nhiên trên màn hình.

```
#include <graphics.h>
#include <alloc.h>
#include <stdlib.h>
#include <conio.h>
#include <dos.h>

void ThietLapDoHoa()
{
 int gd=0,gm;
 initgraph(&gd,&gm,"D:\\TC\\bgi");
}

int r = 20, StartX = 100, StartY = 50;

void Move(int &x,int &y)
{
  int Step=random(2*r);
  if(Step%2==0) Step=-Step;
```

```
  x=x+Step;
  Step=random(r);
  if(Step%2==1) Step=-Step;
  y=y+Step;
}

void VeDiaBay()
{
  ellipse(StartX,StartY,0,360,r,r/3+2);
  ellipse(StartX,StartY-4,190,357,r,r/3);
  line(StartX+7,StartY-6,StartX+10,StartY-12);
  line(StartX-7,StartY-6,StartX-10,StartY-12);
  circle(StartX+10,StartY-12,2);
  circle(StartX-10,StartY-12,2);
}

void Play()
{
  unsigned int x1,y1,x2,y2,size;
  int x,y;
  void far *buff;
  VeDiaBay();
  //lấy vùng hình chữ nhật chứa hình đĩa bay
  x1=StartX - (r+1);
  y1=StartY - 14;
  x2=StartX + r + 1;
  y2=StartY + r/3 + 3;
  // Luu va xoa anh
  size=imagesize(x1,y1,x2,y2);
  buff=farmalloc(size);
  getimage(x1,y1,x2,y2,buff);
  putimage(x1,y1,buff,XOR_PUT); //Xoa anh
  x=getmaxx()/2;
```

```
  y=getmaxy()/2;
  //Di chuyen dia bay
  do
  {
    putimage(x,y,buff,XOR_PUT); //Ve dia bay
    delay(200);
    putimage(x,y,buff,XOR_PUT); // Xoa dia bay
    Move(x,y);
  }
  while (kbhit()==0);
  farfree(buff);  //Giai phong vung nho
}

void main()
{
  ThietLapDoHoa();
  Play();
  closegraph();
}
```

BÀI TẬP

Bài tập 9.1: Viết chương trình vẽ bàn cờ quốc tế lên màn hình.

Bài tập 9.2: Viết chương trình vẽ một chiếc xe ô tô (theo hình dung của bạn) và cho nó chạy ngang qua màn hình.

Gợi ý:

Dùng kỹ thuật lật trang màn hình hoặc di chuyển vùng màn hình.

Bài tập 9.3: Viết chương trình vẽ lá cờ tổ quốc đang tung bay.

Gợi ý:

Dùng kỹ thuật lật trang màn hình.

Bài tập 9.4: Viết chương trình nhập vào n học sinh của một lớp học bao gồm 2 trường sau: Họ tên, điểm trung bình.

a/ Hãy thống kê số lượng học sinh giỏi, khá, trung bình và yếu.

b/ Vẽ biểu đồ thống kê số lượng học sinh giỏi, khá, trung bình và yếu theo 2 dạng: biểu đồ cột (column) và biểu đồ bánh tròn (Pie).

Bài tập 9.5: Viết chương trình để vẽ đồ thị của các hàm số sau:

a/ $y = ax^2 + bx + c$

b/ $y = ax^4 + bx^3 + cx^2 + dx + e$

c/ $y = \frac{ax+b}{cx+d}$

d/ $y = \frac{ax^2+bx+c}{dx+e}$

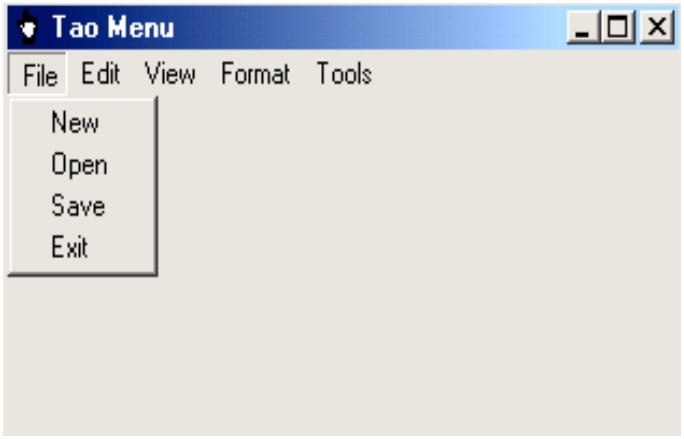
Bài tập 9.6: Viết chương trình vẽ cái đồng hồ đang hoạt động.

Bài tập 9.7: Viết chương trình mô phỏng chuyển động của trái đất xung quanh mặt trời và đồng thời chuyển động của mặt trăng xung quanh trái đất.

Gợi ý:

Dùng ma trận của phép quay.

Bài tập 9.8: Viết chương trình tạo Menu đồ họa giống như các Menu trong môi trường WINDOWS (xem hình).



Bài tập 9.9: Xây dựng một chương trình chứa tất cả các bài tập trong chương này.

PHỤ LỤC A: HƯỚNG DẪN SỬ DỤNG TURBO C++ 3.0

1. Một số phím nóng thông dụng

- **F2**: Lưu chương trình đang soạn thảo vào đĩa.
- **F3**: Mở file mới hoặc file đã tồn tại trên đĩa để soạn thảo.
- **Alt-F3**: Đóng file đang soạn thảo.
- **Alt-F5**: Xem kết quả chạy chương trình.
- **F8**: Chạy từng câu lệnh một trong chương trình.
- **Alt-X**: Thoát khỏi Turbo C.
- **Alt-<Số thứ tự của file đang mở>**: Dịch chuyển qua lại giữa các file đang mở.
- **F10**: Vào hệ thống Menu của Turbo C.

2. Các phím thông dụng khi soạn thảo chương trình

2.1. Các phím soạn thảo

- **Insert**: Chuyển qua lại giữa chế độ đè và chế độ chèn.
- **Home**: Đưa con trỏ về đầu dòng.
- **End**: Đưa con trỏ về cuối dòng.
- **Page Up**: Đưa con trỏ lên một trang màn hình.
- **Page Down**: Đưa con trỏ xuống một trang màn hình.
- **Del**: Xoá ký tự ngay tại vị trí con trỏ.
- **Back Space (←)**: Xóa ký tự bên trái con trỏ.
- **Ctrl-PgUp**: Đưa con trỏ về đầu văn bản.
- **Ctrl-PgDn**: Đưa con trỏ về cuối văn bản.
- **Ctrl-Y**: Xóa dòng tại vị trí con trỏ.

2.2. Các thao tác trên khối văn bản

- Chọn khối văn bản: **Shift + <Các phím ←↑→↓>**
- **Ctrl-KY**: Xoá khối văn bản đang chọn
- **Ctrl-Insert**: Đưa khối văn bản đang chọn vào Clipboard
- **Shift-Insert**: Dán khối văn từ Clipboard xuống vị trí con trỏ.
- **Alt-Back Space**: tương đương với **Undo**

PHỤ LỤC B: HƯỚNG DẪN SỬ DỤNG DEV C++ 5.0

1. GIỚI THIỆU

Dev C++ là một công cụ hỗ trợ cho các lập trình viên biên dịch các chương trình viết bằng ngôn ngữ C/C++ trên cả hệ thốngWindows lẫn Linux. Phiên bản hiện tại của Dev C++ là 4.9.9.2. Bạn đọc có thể download tại địa chỉ: http://www.bloodshed.net/.

Dev C++ thường được sử dụng trong việc học tập và giảng dạy tại các trường học cũng như trong các kỳ thi Olympic Tin học vì giao diện dễ sử dụng và dung lượng nhỏ gọn. Mã nguồn viết trên Dev C++ có thể biên dịch được trên các môi trường khác nhưng mã nguồn viết trên các môi trường khác có thể không biên dịch được trên Dev C++.

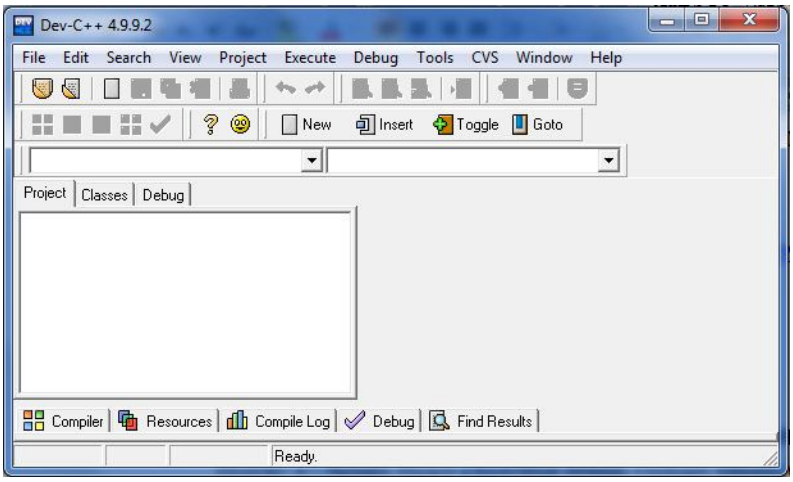
2. CÀI ĐẶT VÀ THIẾT LẬP CẤU HÌNH

2.1. Cài đặt

Tải file cài đặt tại http://www.bloodshed.net/dev/devcpp.html và tiến hành cài đặt theo các bước hướng dẫn có trên màn hình.

2.2. Thiết lập cấu hình Dev C++ sau khi cài đặt

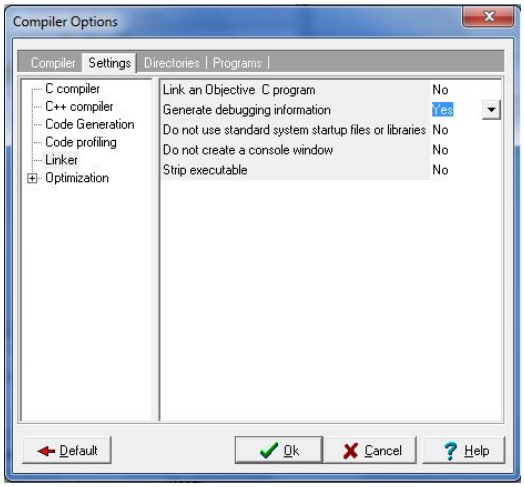
- Sau khi cài đặt xong, chạy chương trình Dev C++, màn hình sẽ xuất hiện như sau:



- Chỉ cần thiết lập cấu hình một lần duy nhất vào lần đầu tiên sau khi cài đặt chương trình:

  - Vào menu **Tools**, chọn **Compiler Options**, chọn tab **Settings**, chọn mục **Linker**.
  - Thay đổi thông số trong mục **Generate Debugging Information** thành **Yes.**

- Kích **OK** để hoàn tất việc cấu hình.

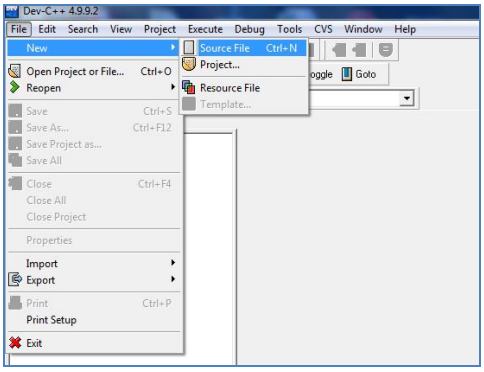


3. TẠO MỘT CHƯƠNG TRÌNH HOẶC DỰ ÁN MỚI

3.1. Tạo một chương trình mới

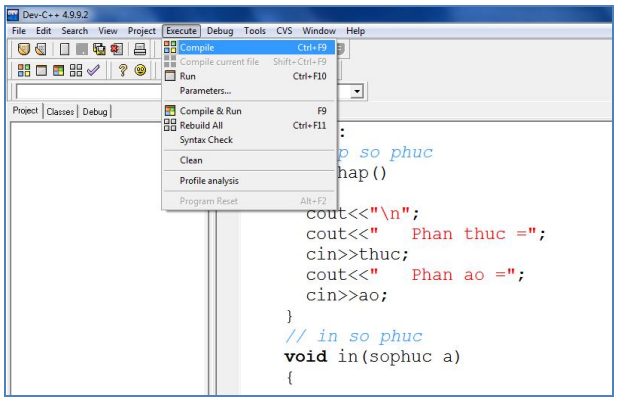
Các bước cơ bản khi tạo một chương trình Dev C++ đơn giản (chỉ có một file mã nguồn) được thực hiện như sau:

Bước 1: Vào menu File → New → Source File (xem hình) hoặc nhấn tổ hợp phím Ctrl_N:

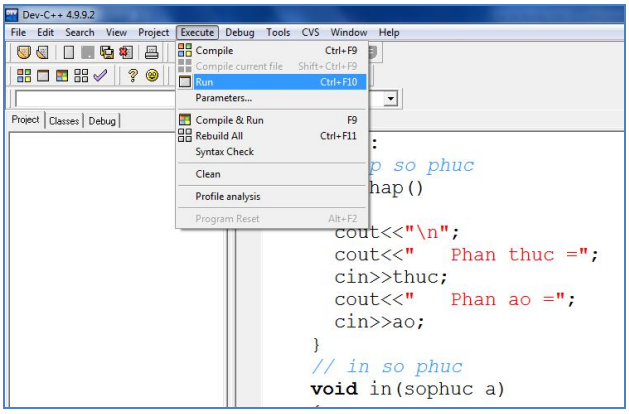


Bước 2: Soạn thảo chương trình.

Bước 3: Dịch chương trình: Vào menu Execute → Compile (hoặc nhấn tổ hợp phím Ctrl_F9), nếu có lỗi thì phải sửa lỗi.



Bước 4: Chạy chương trình: Vào menu Execute → Run (hoặc nhấn tổ hợp phím Ctrl_F10) và kiểm tra kết quả:

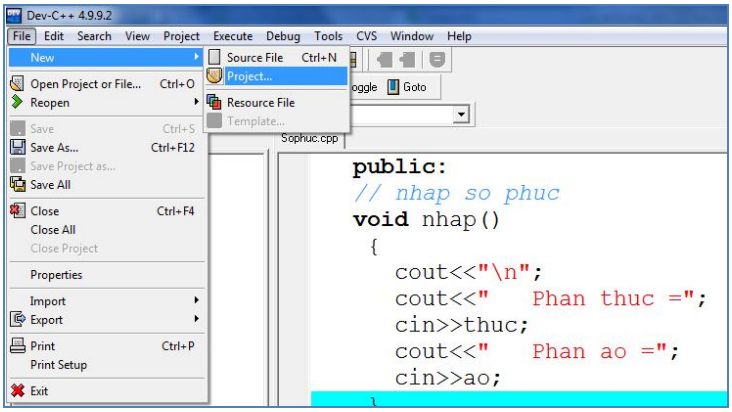


Bước 5: Debug (nếu cần): Vào menu Debug → Debug hoặc nhấn phím F8.

3.2. Tạo một dự án mới

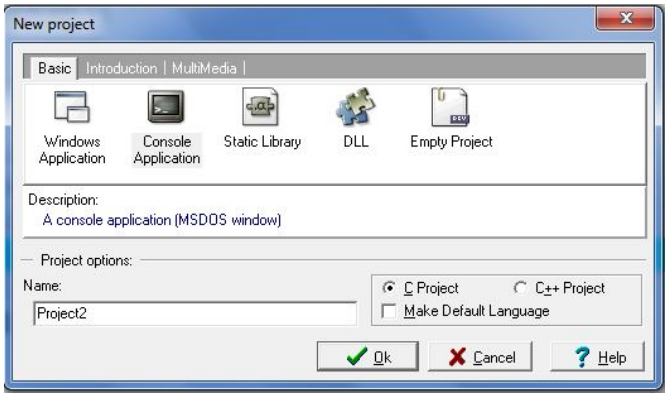
Đối với những chương trình lớn và phức tạp thì ta có thể tạo một dự án (project) bao gồm nhiều file mã nguồn và các file tài nguyên...

Bước 1: Tạo project mới: Vào menu File → New → Project:



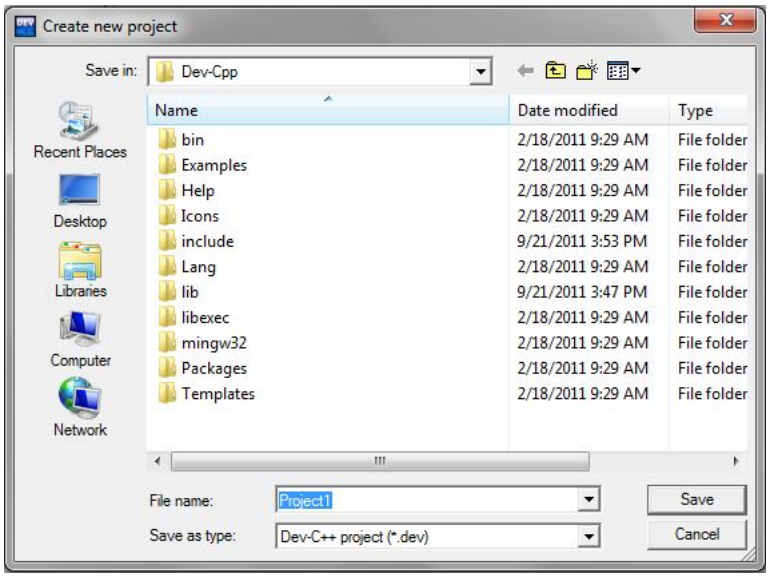
Bước 2:

- Chọn loại ứng dụng: **Console Aplication** (đối với các chương trình đơn giản), ứng dụng WinForm (Windows Application), thư viện liên kết động (DLL),…
- Chọn **C Project** nếu muốn viết bằng ngôn ngữ C, chọn **C++ Project** nếu viết bằng ngôn ngữ C++.
- Đặt tên cho dự án ở mục Name (tên Project chính là tên của file thực thi .exe khi biên dịch):

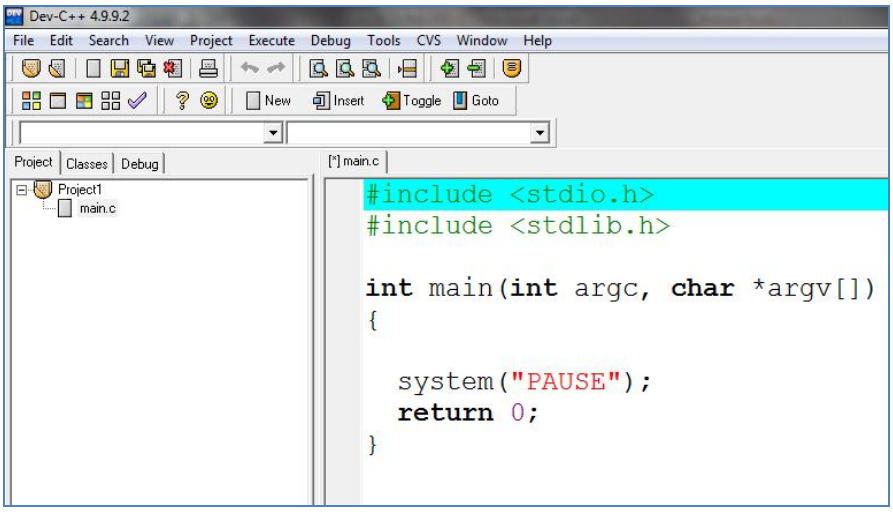


- Chọn OK.

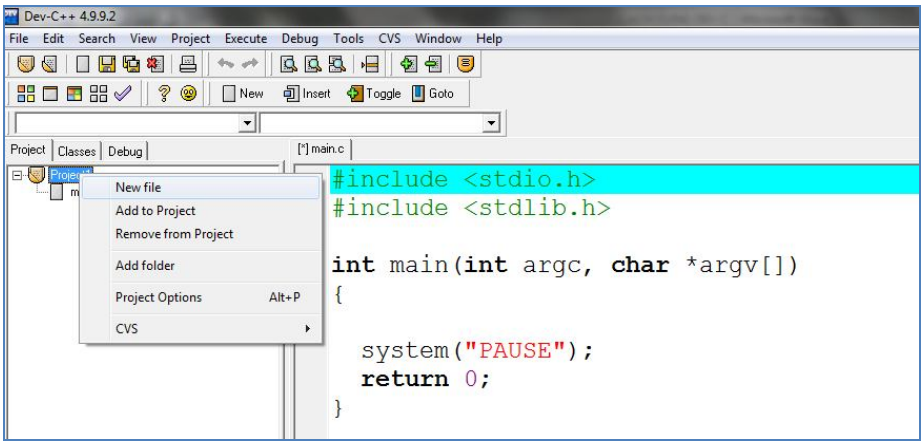
Bước 3: Chọn nơi lưu các file dự án (Nên tạo thư mục mới để lưu).



Sau khi nhập tên dự án và nhấn nút Save thì giao diện Dev C++ sẽ xuất hiện như sau:



Để tạo thêm các file nguồn cho dự án, vào thư mục **Project,** kích chuột phải và chọn **New file** như hình sau:



Bước 4: Dịch chương trình: Vào menu **Execute** → **Compile** (hoặc nhấn tổ hợp phím Ctrl_F9), nếu sai thì sửa lỗi.

Bước 4: Chạy chương trình: Vào menu **Execute** → **Run** (hoặc nhấn tổ hợp phím Ctrl_F10) và kiểm tra kết quả.

4. LẬP TRÌNH ĐỒ HỌA TRONG DEV C++

4.1. Thiết lập cấu hình cho Dev C++ để lập trình đồ hoạ

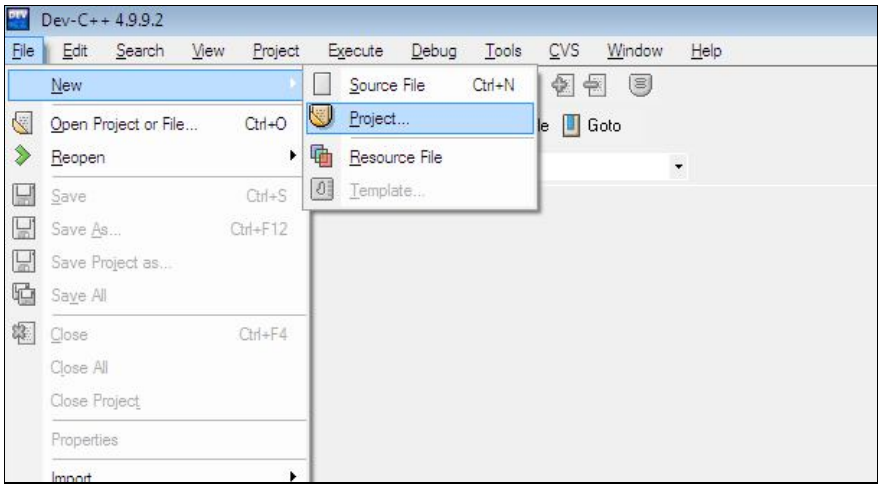
Để có thể lập trình đồ hoạ trong DeV C++, cần phải thiết lập thêm cấu hình theo các bước sau:

Bước 1: Tải về hai file sau:

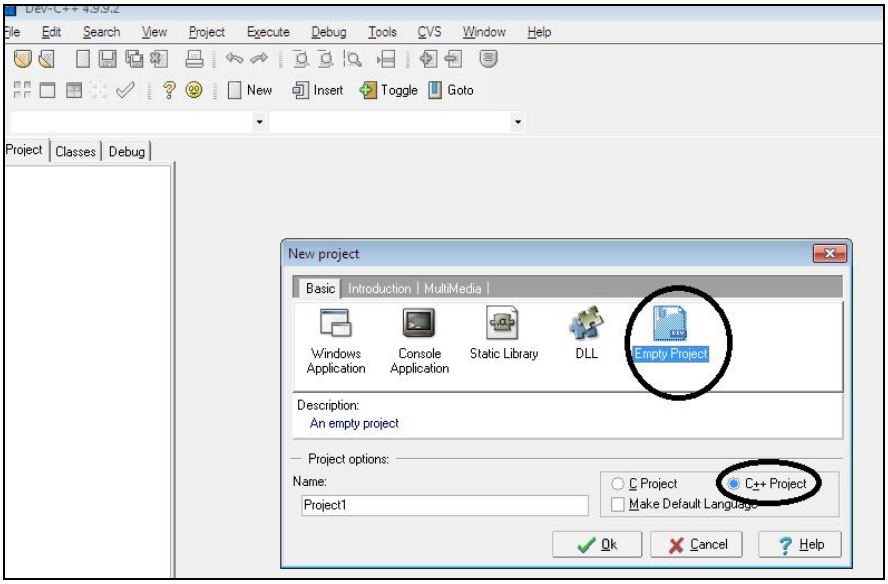
graphics.h: Lưu vào thư mục \Dev-Cpp\include.

libbgi.a: Lưu vào thư mục \Dev-Cpp\lib

Bước 2: Khởi động Dev C++, vào menu File → New → Project

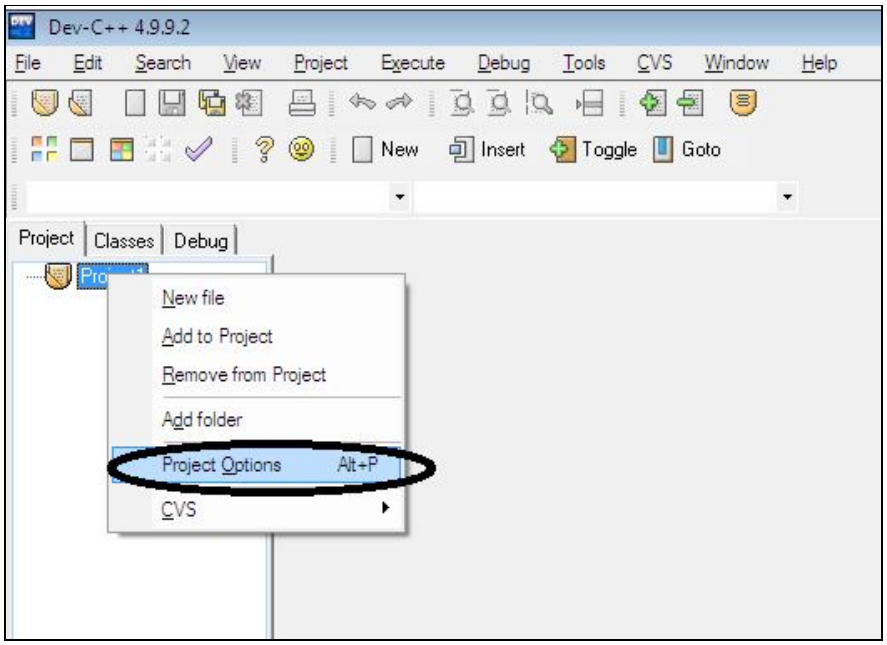


Cửa sổ New Project xuất hiện, chọn **Empty Project** và chọn **C++ Project** (như hình sau) → Chọn **OK.**

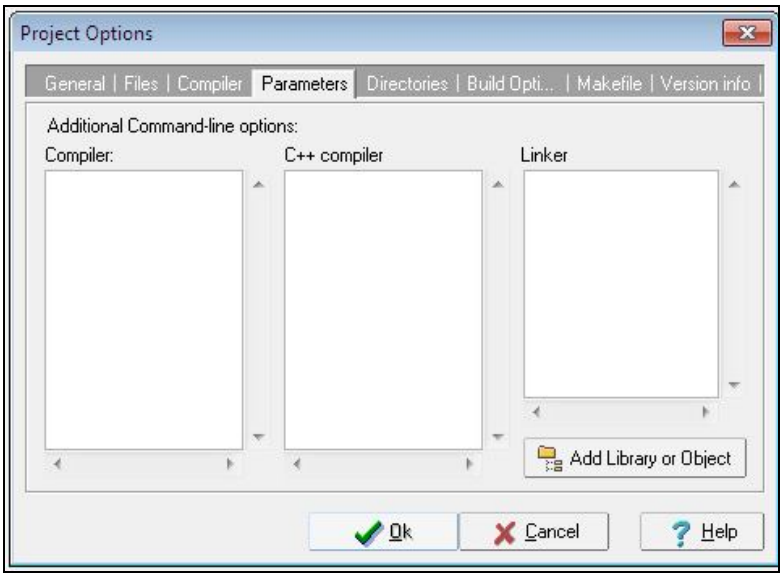


Bước 3: Đặt tên cho project vừa tạo và lưu vào thư mục bất kỳ trên máy của bạn.

Bước 4: Chọn procject vừa tạo, **kích chuột phải** và chọn **Project Option** (hoặc nhấn tổ hợp phím **Alt_P**).

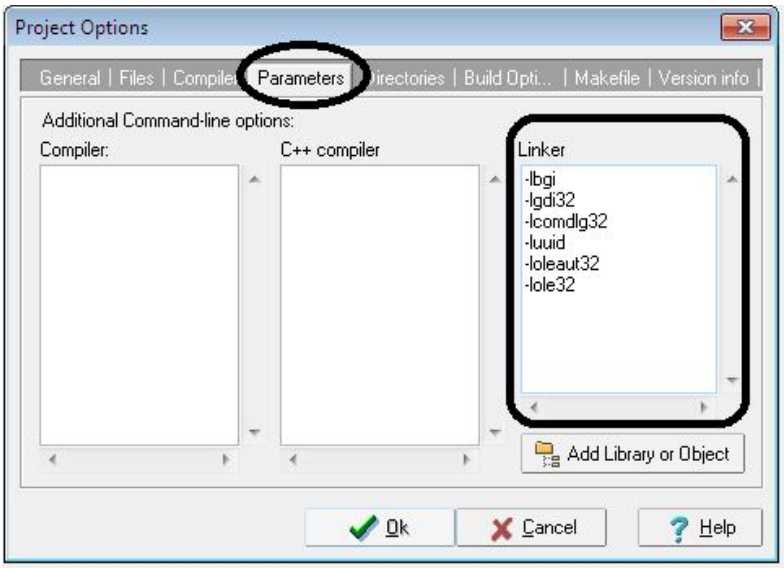


Cửa sổ **Project Option** xuất hiện như sau:



Chọn tab **Parameters**. Ở cửa sổ **Linker** nhập nội dung sau vào khung **Linker:**

```
-lbgi
-lgdi32
-lcomdlg32
-luuid
-loleaut32
-lole32
```



Chọn **OK**.

Bây giờ bạn hoàn toàn có thể sử dụng thư viện **graphics.h** trong DevC++.

4.2. Ví dụ minh họa

Bước 1: Tạo một project mới đặt tên là Project1. Lưu Project1 vào một thư mục bất kỳ trên máy.

Bước 2: Tạo một file mới có tên là **THUDOHOA.CPP**

Bước 3: Gõ toàn bộ nội dung đoạn mã lệnh sau vào file **THUDOHOA.CPP**

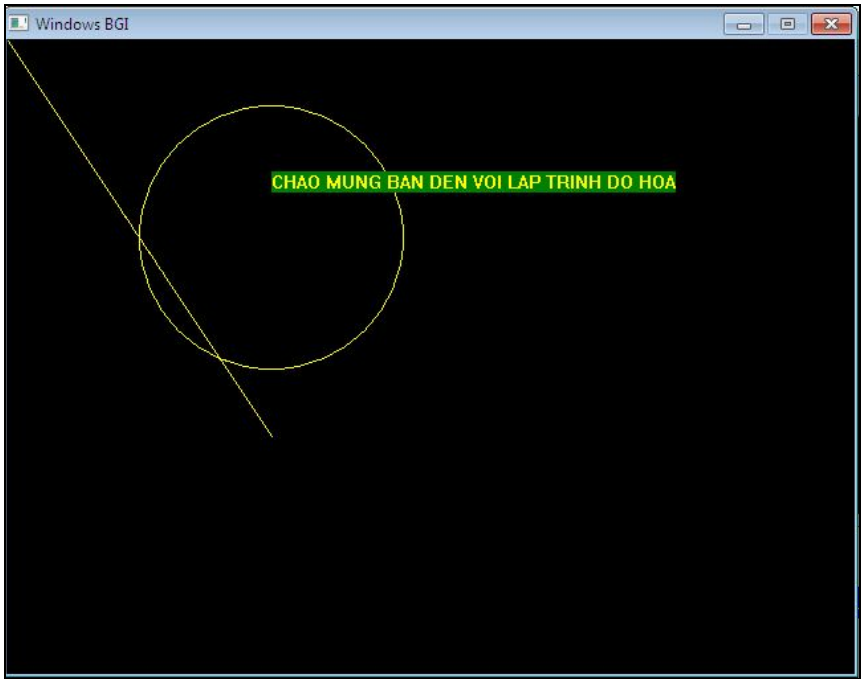
```
#include <cstdlib>
#include <iostream>
#include <graphics.h>
#include <stdio.h>
#include <conio.h>
#include "windows.h"
using namespace std;
void KhoiTaoDohoa()
{
    int driver=0,mode;
    initgraph(&driver,&mode,"");
}
int main(int argc, char *argv[])
{
    KhoiTaoDohoa();
    setbkcolor(GREEN);
    setcolor(YELLOW);
    line(1,1,200,300);
```

```
        circle(200,150,100);
        outtextxy(200,100,"CHAO MUNG BAN DEN VOI LAP TRINH DO HOA");
        system("PAUSE");
        return EXIT_SUCCESS;
}
```

Bước 4: Biên dịch chương trình: Vào menu **Execute** chọn **Compile** hoặc nhấn tổ hợp phím **Ctrl_ F9.**

Bước 5: Chạy chương trình**:** Vào menu **Execute** chọn **Run** hoặc nhấn tổ hợp phím **Ctrl_ F10**.

Trên màn hình xuất hiện kết quả như sau:



Chú ý: Khi khai báo các thư viện **iostream.h** và **string.h** bạn cần thực hiện khai báo như sau:

```
#include <iostream>
#include <string>
using namespace std;
```

TÀI LIỆU THAM KHẢO

[1] Phạm Văn Ất, ***Kỹ thuật lập trình C cơ bản và nâng cao***, NXB Khoa học và Kỹ thuật, 1999.

[2] Gerald Leblanc, ***Turbo C***, NXB Khoa học và Kỹ thuật, 1995.

[3] Nguyễn Xuân Huy, ***Sáng tạo trong thuật toán và lập trình***, NXB Thông tin và Truyền thông, 2011.