CHƯƠNG 6 : ĐỒ HOẠ TRONG C

§1. KHÁI NIỆM CHUNG

Turbo C có khoảng 100 hàm đồ hoạ . Các hàm này được chia làm hai kiểu :

- Loại theo kiểu văn bản (ví dụ hàm tạo cửa sổ)
- Loại theo kiểu đồ hoạ

§2. HÀM THEO KIỂU VĂN BẢN

Các hàm này được dùng với màn hình đơn sắc hay màn hình đồ hoạ . Ta phải đặt vào đầu chương trình dòng #include <conio.h> .

1. Cửa sổ : Mục đích của các hàm đồ hoạ theo kiểu văn bản là tạo ra các cửa sổ . Cửa sổ là vubgf hình chữ nhật trên màn hình dùng để giới hạn vùng xuất dữ liệu . Nếu ta soạn thảo văn bản trong cửa sổ thì con nháy chỉ di chuyển trong phạm vi của cửa sổ chứ không phải toàn bộ màn hình . Ta xét một chương trình tạo ra cửa sổ và điền đầy vào đó dòng " Xin chao "

Chương trình 6-1 :

```
#include <conio.h>
#include <dos.h>
#define left 10
#define top 8
#define right 52
#define bot 21

void main()
  {
    int i;
    clrscr();
    window(left,top,right,bot);
    textcolor(RED);
    textbackground(GREEN);
    for (i=0;i<100;i++)
      {
          cputs(" Xin chao ");
          delay(100);
      }
    gotoxy(15,8);
    cputs("Ket thuc");
    getche();
  }
```

Trong chương trình ta có hàm :

window(x1,y1,x2,y2) dùng để ấn định một cửa sổ có toạ độ góc trên trái là x1,y1 và góc dưới phải là x2,y2

textcolor(RED) để ấn định màu chữ là đỏ

textbackcolor(GREEN) để ấn định màu nền văn bản là xanh lá cây

gotoxy(x,y) để di chuyển con nháy về toạ độ x,y

cputs(string) để đặt chuỗi string trong một cửa sổ . Khi gặp biên của cửa sổ chuỗi sẽ được xuống dòng . Màu trong chế độ đồ hoạ được quy định như sau :

Số	Màu
0	BLACK
1	BLUE
2	GREEN
3	CYAN
4	RED
5	MAGENTA
6	BROWN
7	LIGHTGRAY
8	DARKGRAY
9	LIGHTBLUE
10	LIGHTGREEN
11	LIGHTCYAN
12	LIGHTRED
13	LIGHTMAGENTA
14	YELLOW
15	WHITE

2. Dời chỗ văn bản : Muốn dời chỗ một vùng hình chữ nhật của văn bản từ nơi này sang nơi khác ta dùng hàm movetext() . Ta xét chương trình sau tạo ra một cửa sổ , điền đầy cửa sổ bằng một đoạn văn bản và dời cửa sổ sang vị trí khác trên màn hình

Chương trình 6-2 :

```
#include <conio.h>
#include <dos.h>
#define left 26
#define top 7
#define right 65
#define bot 20
#define desleft 1
#define destop 1
#define numcolor 16

void main()
  {
    int i;
    clrscr();
    window(left,top,right,bot);
    textbackground(GREEN);
    for (i=0;i<100;i++)
      {
          textcolor(i%numcolor);
          cputs(" Xin chao ");
          delay(200);
      }
    delay(2000);
    movetext(left,top,right,bot,desleft,destop);
```

```
  getche();
 }
```

Hàm movetext(x1,y1,x2,y2,x0,y0) dùng di chuyển cửa sổ x1,y1,x2,y2 đến vị trí mới mà toạ độ góc trên trái bây giờ là x0,y0 .

3.Lưu trữ và phục hồi màn hình văn bản : Ta có thể lưu trữ một vùng văn bản hình chữ nhật trên màn hình và sau đó phục hồi lại tại một vị trí nào đó trên màn hình . Nhờ vậy ta có thể tạo một cửa sổ nhỏ trên đầu văn bản hiện hành . Ta xét ví dụ sau

Chương trình 6-3 :

```
#include <conio.h>
#include <dos.h>
#define left 1
#define top 1
#define right 80
#define bot 25
int buf[80][25];
void main()
 {
  int i,x,y;

  clrscr();
  for (i=0;i<300;i++)
    cputs(" Turbo C ");
  getche();
  gettext(left,top,right,bot,buf);
  clrscr();
  getch();
  puttext(left,top,right,bot,buf);
  getch();
 }
```

Chương trình lưu toàn bộ màn hình vào vùng đệm có tên là buf nhớ hàm gettext(x1,y1,x2,y2,buf) lưu vn trong hình chữ nhật x1,y1,x2,y2 vào biến buf . Hàm puttext(x1,y1,x2,y2,buf) đặt lại văn bản trong hình chữ nhật x1,y1,x2,y2 lưu bởi biến buf ra màn hình .

3. Một số hàm đồ hoạ văn bản khác :

- void clreol(void) : xoá đến cuối dòng
- int cprintf(const char *format) đưa kí tự ra một cửa sổ
- void textattr(int newattr) ấn định màu cùng lúc cho văn bản và nền
- void gettextinfo(struct text_info *r) : đọc các thông tin như kiểu màn hình , vị trí và kích thước cửa sổ , màu nền và văn bản ,vị trí con nháy
- void normvideo(void) trả lại độ sáng cũ
- void insline(void) : chèn thêm một dòng
- void delline(void) xoá một dòng
- void hightvideo(void) tăng độ sáng
- void lowvideo(void) : giảm độ sáng
- void textmode(int newmode) chuyển đổi giữa các kiểu văn bản . Hàm dùng các đối số sau :

Trị	Hằng	Ý nghĩa

-1	LASTMODE	Kiểu văn bản trước đó
0	BW40	Đen trắng 40 cột
1	C40	Màu 40 cột
2	BW80	Đen trắng 80 cột
3	C80	Màu 80 cột
7	MONO	Đơn sắc 80 cột

§3. CÁC HÀM ĐỒ HOẠ

1. Khởi tạo kiểu đồ hoạ : Để khởi tạo đồ hoạ ta dùng hàm initgraph() được khai báo trong graphics.h với cú pháp :

void far initgraph(int *graphdrive , int *graphmode , char *path);

với các biến graphdrive chứa trình điều khiển đồ hoạ

graphmode kiểu đồ hoạ

path đường dẫn đến thư mục chứa các drive đồ hoạ . Trong phần này ta phải dùng hai dấu \\ vì dấu \ đã được dùng cho kí tự escape .

Để thuận tiện ta khởi tạo đồ hoạ tự động bằng cách viết :

graphdrive = detect;

initgraph(graphdrive , graphmode , path);

Ta có chương trình vẽ đường thẳng và đường tròn như sau :

Chương trình 6-4 :

```
#include <graphics.h>
#include <conio.h>
void main()
  {
   int gd,gm;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   line(0,0,100,100);
   circle(100,100,50);
   getche();
   closegraph();
  }
```

2. Lỗi đồ hoạ : Để biết lỗi đồ hoạ ta dùng hàm int far graphresult(void) . Sau khi biết mã lỗi ta chuyển nó sang cho hàm grapherrormsg() . Hàm này trả về con trỏ chỉ đến lỗi . Sau đây là chương trình minh hoạ

Chương trình 6-5 :

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
void main()
  {
   int gd,gm,ge;
   char *ep;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   ge=graphresult();
```

```
  if (ge)
    {
       printf("Ma loi %d",ge);
       ep=grapherrormsg(ge);
       puts(ep);
       getch();
       exit(1);
    }
  line(0,0,100,100);
  circle(100,100,50);
  getche();
  closegraph();
}
```

3. Đường thẳng và màu sắc : Để thiết lập dạng , mẫu và bề dày của đường thẳng ta dùng hàm void far setlinestyle(int style,int pattern, int thickness) . Tham biến style có thể là :

Trị	Hằng	Y nghĩa
0	SOLID_LINE	Đường đặc
1	DOTTED_LINE	Đường chấm
2	CENTER_LINE	Đường gạch
3	DASHED_LINE	Đường gạch dài
4	USERBIT_LINE	Đường tự tạo

Tham biến thickness có thể nhân một trong hai giá trị sau :

Trị	Hằng	Y nghĩa
1	NORM_WIDTH	dãy 1 điểm ảnh
2	THICK_WIDTH	dãy 3 điểm ảnh

Để xác định màu cho đường thẳng ta dùng hàm void setcolor(int color) . Ta có chương trình sau

Chương trình 6-6 :

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

void main()
  {
   int gd,gm,ge;
   int x1=0,y1=0;
   int x2=199,y2=199;
   int xc=100,yc=100;
   int r=90;
   char *ep;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   ge=graphresult();
   if (ge)
```

```
    {
        printf("Ma loi %d",ge);
        ep=grapherrormsg(ge);
        puts(ep);
        getch();
        exit(1);
    }
  setlinestyle(1,1,1);
  setcolor(LIGHTGREEN);
  line(x1,y1,x2,y2);
  circle(xc,yc,r);
  getche();
  closegraph();
 }
```

4. Ellipse và đa giác : Để vẽ ellipse ta dùng hàm

void far ellipse(int x,int y , int gd,int gc,int xr , int yr)

x,y - toạ độ tâm ellipse

gd,gc - góc bắt đầu vẽ và góc kết thúc vẽ

xr,yr - toạ độ tâm ellipse

Chương trình 6-7 : Vẽ một loạt ellipse

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
void main()
 {
  int gd,gm,ge;
  int x=150,y=150;
  int g1=0,g2=360;
  int xr=150,yr;
  char *ep;
  gd=DETECT;
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  ge=graphresult();
  if (ge)
    {
        printf("Ma loi %d",ge);
        ep=grapherrormsg(ge);
        puts(ep);
        getch();
        exit(1);
    }
  setcolor(RED);
  for (yr=0;yr<100;yr+=10)
    ellipse(x,y,g1,g2,xr,yr);
  getche();
  closegraph();
 }
```

Để vẽ đa giác ta dùng hàm

void far drawpoly(int number , int far *addrlist)
number - số đỉnh đa giác cộng thêm 1
addrlist - mảng chứa toạ độ các đỉnh , toạ độ điểm đầu và cuối phải trùng nhau

Chương trình 6-8 : Vẽ một hình hộp chữ nhật

```
#include <graphics.h>
#include <conio.h>
#define left 50
#define top 50
#define right 150
#define bot 180
int a[]={150,50,180,20,180,135,150,180};
int b[]={50,47,150,47,180,17,95,17,50,47};

void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    rectangle(left,top,right,bot);
    setcolor(2);
    drawpoly(4,a);
    drawpoly(5,b);
    getche();
    closegraph();
  }
```

5. Tô màu và mẫu tô : Turbo C có nhiều hàm để tô màu . Hàm thông dụng nhất để tô bên trong một đa giác và mẫu tô hiện hành là void far fillpoly(int number , int far * addlist) . Màu và mẫu tô được thiết lập nhờ hàm void far setfillstyle(int pattern , int color) . Biến pattern có thể nhận một trong các trị sau :

Trị	Hằng	Ý nghĩa
0	EMPTY_FILL	Rỗng
1	SOLID_FILL	Màu đặc
2	LINE_FILL	Đường ngang
3	LTSLASH_FILL	//// chéo mảnh
4	SLASH_FILL	//// chéo dày
5	BKSLASH_FILL	\\\\ chéo ngược
6	LTBKSLASH_FILL	\\\\ chéo ngược mảnh
7	HATCH_FILL	Sọc dưa thưa
8	XHATCH_FILL	Sọc dưa dày
9	INTERLEAVE_FILL	Đường xen kẽ
10	WIDE_DOT_FILL	Chấm thưa
11	CLOSE_DOT_FILL	Chấm dày
12	USER_FILL	Mẫu tự do

Biến color được chọn theo danh sách đã liệt kê trong phần setcolor(). Nếu dùng giá trị không hợp lệ cho pattern và color thì hàm graphresult() sẽ trả về mã lỗi là -11 . Hàm floodfill() dùng để to màu một hình kín . Nó cần biết điểm bắt đầu tô . Hàm sẽ tô cho đến khi gặp đường biên có màu xác định bằng biến border . Có thể tô bên trong hay ngoài hình vẽ tuỳ điểm bắt đầu . Nếu tô một vùng không kín thì màu tô sẽ lan ra trong lẫn ngoài vật thể . Sau đây là chương trình tô vòng tròn .

Chương trình 6-9 :

```
#include <graphics.h>
#include <conio.h>
#define x 200
#define y 200
#define r 150

void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    circle(x,y,r);
    setfillstyle(CLOSE_DOT_FILL,BLUE);
    floodfill(x,y,RED);
    getche();
    closegraph();
  }
```

Màu dùng để tô có thể giống hay khác với màu dùng cho đường viền của vùng . Tuy vậy màu dùng cho tham biến border của floodfill() phải giống màu vè vật thể (trong chương trình là màu RED)

6. Đồ thị : Turbo C có nhiều hàm giúp đơn giản hoá việc vẽ đồ thị các hàm là bar() , bar3d() và pieslice() .

void bar (int top , int left , int right , int bottom)

void far bar3d(int left , int top , int right , int right , int bottom , int depth , int topflag)

topflag = 0 - có nắp , topflag = 1 - không có nắp

void far pieslice(int x , int y , int startangle , int endangle , int r)

Ta có chương trình minh hoạ

Chương trình 6-10 :

```
#include <graphics.h>
#include <conio.h>
#define n 10
#define bwidth 10
#define sep 12
#define di (bwidth+sep)
#define shft 15
#define width ((n+1)*di)
#define left 5
#define depth 3
#define topflag 1
#define bot 170
```

```
#define top 5
#define ppd (float)(bot-top)/100

void main()
  {
    int gd,gm,i;
    int data[n]={41,47,54,62,63,59,75,83,89,96};
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    rectangle(top,left,left+width,bot);
    for (i=0;i<n;i++)
      {
          setfillstyle(SOLID_FILL,1+i%3);
          bar3d(left+shft+i*di,bot-data[i]*ppd,left+shft+i*di+bwidth,bot,depth,topflag);
      }
    getche();
    closegraph();
  }
```

Sau đây là chương trình dùng pieslice()

Chương trình 6-11 :

```
#include <graphics.h>
#include <conio.h>
#define n 6
#define r 90
int data[n]={11,19,44,32,15,7};

void main()
  {
    int gd,gm,i,x,y;
    float datasum,startangle,endangle,relangle;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    x=getmaxx()/2;
    y=getmaxy()/2;
    setcolor(RED);
    for (i=0,datasum=0;i<n;i++)
      datasum+=data[i];
    endangle=0;
    relangle=10;
    for (i=0;i<n;i++)
      {
          startangle=(i+1)*relangle;
          setfillstyle(SOLID_FILL,i%4);
          pieslice(x,y,startangle,endangle,r);
          getch();
      }
```

```
  getche();
  closegraph();
 }
```

7. Viewport : Viewport là một vùng nhì thấy được của màn hình . Khi mới khởi động viewport là toàn bộ màn hình . Để xác định một viewport ta dùng hàm setviewport() có cú pháp :

```
        void far setviewport(int left , int top , int  right , int bot , int clip)
```

Tham biến clip cho biết hình vẽ có hiện ra ngoài viewport hay không . Nếu clip <>0 thì không thấy được hình bên ngoài viewport . Để xoá một viewport ta dùng hàm void far clearviewport(void)

Chương trình 6-12 :

```
#include <graphics.h>
#include <conio.h>
void main()
 {
  int gd,gm,i;
  int left=0,top=0,right=150,bot=150;
  int x1=0,y1=0,x2=199,y2=199;
  int x=100,y=100;
  int clip=1;
  int r=90;
  gd=DETECT;
  clrscr();
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  setviewport(left,top,right,bot,clip);
  setcolor(RED);
  rectangle(left,top,right,bot);
  line(x1,y1,x2,y2);
  circle(x,y,r);
  getch();
  closegraph();
 }
```

8. Vẽ theo toạ độ tương đối : Trong C ta có thể dùng toạ độ tương đối so với điểm hiện hành CP-current point . Để vẽ đường thẳng ta dùng hàm void far lineto(int x, int y) . Hàm này vẽ đường thẳng từ điểm CP đến điểm mới có toạ độ là x,y . Hàm void far linerel(int dx , int dy) vẽ đường thẳng từ CP(xc,yc) đến điểm có toạ độ (xc+dx,yc+dy) . Thường ta hay kết hợp với hàm void far moveto(int x, int y) để di chuyển điểm hiện hành tới điểm mới có toạ độ (x,y)

Chương trình 6-13 : Vẽ một bàn cờ

```
#include <graphics.h>
#include <conio.h>
#define max 160
#define grid 20
#define size 18
void main()
 {
  int gd,gm,i,j;
  void square(int );
```

```
   gd=DETECT;
   clrscr();
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   for (i=0;i<max;i+=grid)
     for (j=0;j<max;j+=grid)
          {
           moveto(j,i);
           square(size);
          }
   getch();
   closegraph();
  }

void square(int side)
  {
   linerel(side,0);
   linerel(0,side);
   linerel(-side,0);
   linerel(0,-side);
  }
```

9. Điểm ảnh : Để đặt một điểm ảnh lên màn hình ta dùng hàm :

void far putpixel(int x , in y, int color)

Chương trình 6-14 : Lập chương trình vẽ hình sin bằng putpixel

```
#include <graphics.h>
#include <conio.h>
#include <math.h>
void main()
  {
   int gd,gm,x,y;
   double g,sg;
   gd=DETECT;
   clrscr();
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   line (1,100,200,100);
   for (x=0;x<200;x++)
     {
          g=((double)x/200)*(2*3.14159);
          sg=sin(g);
          y=100-100*sg;
          putpixel(x,y,RED);
     }
   getch();
   closegraph();
  }
```

Để xác định màu của một điểm ta dùng hàm int getpixel(int x,int y)

10 . Ảnh bit và làm ảnh chuyển động : Để cất gữi một hình ảnh vào bộ nhớ ta dùng hàm :

void far getimage(int left , int top , int right , int bot , void far * addbuf)

left , top , right , bot - các góc của hình chữ nhật chứa ảnh
addbuf - địa chỉ bộ nhớ dùng chứa ảnh

Hàm này cần biết kích thước của hình . Kích thước này được xác định theo hàm :

unsigned far imagesize(int left , int top , int right , int bot)

Giá trị của hàm được truyền cho hàm malloc() để cấp phát bộ nhớ . Con trỏ do hàm malloc() trả về được truyền cho hàm putimage để khôi phục lại hình đã cất . Cú pháp của putimage() là :

void far putimage(int left , int top , void far * addbuf,int putop)
left,top là góc trên trái của vùng sẽ đưa ảnh ra
addbuf - địa chỉ bộ nhớ dùng chứa ảnh
putop là các đưa ảnh ra . Các hằng putop là :

Trị	Hằng	Ý nghĩa
0	COPY_PUT	Thay hình cũ bằng hình mới
1	XOR_PUT	XOR hình cũ bằng hình mới
2	OR_PUT	OR hình cũ bằng hình mới
3	AND_PUT	AND hình cũ bằng hình mới
5	NOT_PUT	Thay hình cũ bằng đảo hình mới

Chương trình 6-15 : Lập chương trình thể hiện quả bóng dội

```
#include <graphics.h>
#include <conio.h>
#include <alloc.h>
#define left 0
#define top 0
#define right 639
#define bottom 479
#define r 8

void main()
  {
    int gd,gm,x,y;
    int dx,dy,oldx,oldy;
    void far *buf;
    unsigned size;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    rectangle(left,top,right,bottom);
    x=y=r+10;
    setcolor(LIGHTGREEN);
    setfillstyle(SOLID_FILL,LIGHTGREEN);
    circle(x,y,r);
    floodfill(x,y,LIGHTGREEN);
    size=imagesize(x-r,y-r,x+r,y+r);
    buf=malloc(size);
    getimage(x-r,y-r,x+r,y+r,buf);
    dx=1;
    dy=1;
```

```
  while (!kbhit())
    {
       putimage(x-r,y-r,buf,COPY_PUT);
       delay(5);
       oldx=x;
       oldy=y;
       x=x+dx;
       y=y+dy;
       if (x<=left+r+2||x>=right-r-2)
         dx=-dx;
       if (y<=top+r+1||y>=bottom-r-2)
         dy=-dy;
       putimage(oldx-r,oldy-r,buf,XOR_PUT);
       delay(5);
    }
  closegraph();
}
```

§4. VĂN BẢN TRONG ĐỒ HOẠ

1. Các fonts : Để chọn fonts chữ ta dùng hàm :

void far settextstyle(int font , int direction , int charsize)

Các fonts chứa trong các tập tin trong bảng sau

Trị	Hằng	Tập tin
0	DEFAULT_FONT	Cài sẵn
1	TRIPLEX_FONT	trip.chr
2	SMALL_FONT	litt.chr
3	SANSERIF_FONT	sans.chr
4	GOTHIC_FONT	goth.chr
5	SCRIPT_FONT	scrip.chr
6	SIMPLEX_FONT	simp.chr
7	TRIPLEX_SCR_FONT	tscr.chr
8	COMPLEX_FONT	lcom.chr
9	EUROPEAN_FONT	euro.chr
10	BOLD_FONT	bold.chr

Đối direction có thể nhận một trong hai trị :

0 (HORIZ_DIR) - từ trái sang phải

1 (VERT_DIR) - từ trên xuống dưới

Khi đối charsize có trị là 1 , kích thước chữ là nhỏ nhất . Khi kích thước là 2 , chữ sẽ tăng gấp đôi v.v. Để in chuỗi ra màn hình trong chế độ đồ hoạ ta dùng các hàm :

void far outtext(char far * string);

void far outtextxy(int x , int y , char far *string);

Chương trình 6-16 : Dùng hàm settextstyle() để viết chữ

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

```
#define FONTSIZE 4
void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    settextstyle(GOTHIC_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,0,"Gothic");
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,40,"Triplex");
    settextstyle(SMALL_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,80,"Small");
    settextstyle(SANS_SERIF_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,100,"Sanserif");
    settextstyle(DEFAULT_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,160,"Default");
    settextstyle(EUROPEAN_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,200,"Euro");
    settextstyle(BOLD_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,240,"Bold");
    settextstyle(COMPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,300,"Complex");
    settextstyle(SCRIPT_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,340,"Script");
    settextstyle(SIMPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,370,"Simplex");
    settextstyle(TRIPLEX_SCR_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,420,"Triplex script");
    getch();
    closegraph();
  }
```

2. Justify và định kích thước văn bản : Hàm định vị trí văn bản là ;

void far settextjustify(int horiz , int vert);

Đối horiz nhận các biến trong bảng sau

Trị	Hằng	Ý nghĩa
0	LEFT_TEXT	CP nằm bên trái văn bản
1	CENTER_TEXT	CP nằm bên chính giữa văn bản
2	RIGHT_TEXT	CP nằm bên phải văn bản

Đối vert nhận một trong các giá trị sau :

Trị	Hằng	Ý nghĩa
0	BOTTOM_TEXT	CP nằm ở đáy văn bản
1	CENTER_TEXT	CP nằm bên chính giữa văn bản
2	TOP_TEXT	CP nằm ở đỉnh văn bản

Chương trình 6-17 :

```
#include <graphics.h>
#include <conio.h>
#define cl 150
#define lead 40
#define fontsize 3
void main()
  {
    int gd,gm,i;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,fontsize);
    line(cl,0,cl,200);
    for (i=0;i<lead*5;i+=lead)
      line(0,i,300,i);
    moveto(cl,0);
    outtext("Default");
    moveto(cl,lead);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    outtext("Left-top");
    moveto(cl,lead*2);
    settextjustify(RIGHT_TEXT,TOP_TEXT);
    outtext("Right-top");
    moveto(cl,lead*3);
    settextjustify(CENTER_TEXT,TOP_TEXT);
    outtext("Center-top");
    moveto(cl,lead*4);
    settextjustify(CENTER_TEXT,BOTTOM_TEXT);
    outtext("Center-bottom");
    getch();
    closegraph();
  }
```

Để thay đổi kích thước và tỉ lệ chữ ta dùng hàm :

void far setusercharsize(int multx , int divx , int multy , int divy);

multx - nhân chiều rộng của kí tự

divx - chia chiều rộng của kí tự

multy - nhân chiều cao của kí tự

divx - chia chiều cao của kí tự

Chương trình 6-18 : Tạo một đồ thị có ghi chú

```
#include <graphics.h>
#include <conio.h>
#include <stdlib.h>
#define n 12
#define bwidth 20
#define sep 24
#define shft 30
#define left 5
```

```
#define depth 6
#define topflag 1
#define bot 300
#define top 5
#define ticks 10
#define twidth 10
#define maxdata 100
#define xtitle 40
#define ytitle 40
#define font SANS_SERIF_FONT
#define di (bwidth+sep)
#define width (((n+1)*di))
#define pbt ((float)(bot-top))
#define ppd ((float)(bot-top)/maxdata)

void main()
  {
    int gd,gm,i;
    float a,b,c,d;
    int data[n]={41,47,54,62,63,59,75,83,89,96,55,2};
    char month[12][4]={"Jan","Feb","Mar","Apr","May","Jun","Jul",
                       "Aug","Sep","Oct","Nov","Dec"};
    char string[40];
    clrscr();
    gd=DETECT;
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    rectangle(left,top,left+width,bot);
    setusercharsize(4,3,4,3);
    settextstyle(font,HORIZ_DIR,0);
    moveto(xtitle,ytitle);
    outtext("1998 Sales");
    setusercharsize(2,3,2,2);
    settextstyle(font,HORIZ_DIR,0);
    for (i=0;i<ticks;i++)
      {
          line(left,bot-i*pbt/10,left+twidth,bot-i*pbt/10);
          line(left+width-twidth,bot-i*pbt/10,left+width,bot-i*pbt/10);
          moveto(left+width+sep,bot-(i+1)*pbt/10);
          itoa(i*(maxdata/ticks),string,10);
          outtext(string);
      }
    setusercharsize(2,3,2,2);
    settextstyle(font,VERT_DIR,0);
    for (i=0;i<n;i++)
      {
          setfillstyle(SOLID_FILL,i);
          bar3d(left+shft+i*di,bot-data[i]*ppd,left+shft+i*di+bwidth,bot,depth,topflag);
          moveto(left+i*di+bwidth-1,bot+5);
          outtext(month[i]);
      }
```

```
  getch();
  closegraph();
 }
```

§5. VÍ DỤ KẾT THÚC

Chương trình 6-19 : Lập chương trình vẽ một mặt Mallbrot

```
#include <graphics.h>
#include <conio.h>
#include <stdlib.h>
#define ymax 400
#define xmax 400
#define maxcount 16
void main()
 {
  int gd,gm;
  int x,y,count;
  float xscale,yscale;
  float left,top,xside,yside,zx,zy,cx,cy,tempx;
  clrscr();
  gd=DETECT;
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  left=-2.0;
  top=1.25;
  xside=2.5;
  yside=-2.5;
  xscale=xside/xmax;
  yscale=yside/ymax;
  rectangle(0,0,xmax+1,ymax+1);
  for (y=1;y<=ymax;y++)
   {
      for (x=1;x<=xmax;x++)
       {
        cx=x*xscale+left;
        cy=y*yscale+top;
        zx=zy=0;
        count=0;
        while((zx*zx+zy*zy<4) && (count<maxcount))
         {
            tempx=zx*zx-zy*zy+cx;
            zy=2*zx*zy+cy;
            zx=tempx;
            count++;
         }
        putpixel(x,y,count);
        if (kbhit())
         exit(0);
       }
```

```
    }
  getch();
  closegraph();
}
```