**Write a program for implementation of rotation transformation.**

Code:-

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#include<dos.h>

#include<stdlib.h>

#include<math.h>

#include<stdio.h>

void main()

{

int gd=DETECT,gm;

void drawaxis(void);

void dda(int,int,int,int);

void polygon(int,int,int,int);

void rotation(void);

initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");

rotation();

closegraph();

}

void drawaxis()

{

int j;

char s[25];

settextstyle(0,0,0);

line(0,240,640,240);

outtextxy(570,228,"X-AXIS");

settextstyle(0,0,0);

line(320,0,320,480);

outtextxy(332,6,"Y-AXIS");

settextstyle(0,0,0);

for(j=240;j>-240;j-=20)

{

outtextxy(318,237-1,"-");

if(j==0)

continue;

itoa(j,s,10);

outtextxy(286,240-j,s);

}

for(j=-300;j<300;j+=20)

{

outtextxy(j+300+5,236,"-");

if(j==0)

continue;

itoa(j,s,10);

settextstyle(0,1,0);

outtextxy(j+325,245,s);

}

}

void dda(int x1,int y1,int x2,int y2)

{

int k,dx,dy,steps;

float x,y,incrx,incry;

dx=x2-x1;

dy=y2-y1;

if(abs(dx)>abs(dy))

steps=abs(dx);

else

steps=abs(dy);

incrx=(float)dx/steps;

incry=(float)dy/steps;

x=x1;

y=y1;

putpixel(320+x,240-y,WHITE);

for(k=0;k<steps;k++)

{

x+=incrx;y+=incry;

putpixel(320+x,240-y,WHITE);

}

}

void polygon(int cx[],int cy[],int n)

{

int i;

for(i=0;i<n-1;i++)

{

dda(cx[i],cy[i],cx[i+1],cy[i+1]);

dda(cx[n-1],cy[n-1],cx[0],cy[0]);

}

}

void rotation(void)

{

int ax[15],ay[15],n;

float A,C,S,h[3][2],temp=0;

int tx[15]={0},ty[15]={0},i,j;p

cout<<"Enter size of polygon:";

cin>>n;

if(n<3)

{

cout<<"Invalid size";

closegraph();

exit(1);

}

cout<<"Enter the coordinates of polygon :";

for(i=0;i<n;i++)

{

cout<<"Enter for side" <<i+1<<"\n";

cin>>ax[i]>>ay[i];

}

cout<<"Enter the rotation value for A" ;

cin>>A;

cleardevice();

drawaxis();

polygon(ax,ay,n);

for(i=0;i<3;i++)

{

for(j=0;j<2;j++)

{

if(i==j)

h[i][j]=1;

else

h[i][j]=0;

}

}

float PI=3.142;

C=cos((A\*PI)/180);

S=sin((A\*PI)/180);

for(i=0;i<3;i++)

{

temp=h[i][0]\*C-h[i][1]\*S;

h[i][1]=h[i][0]\*S+h[i][1]\*C;

h[i][0]=temp;

}

temp=0;

for(i=0;i<n;i++)

{

temp=ax[i]\*h[0][0]+ay[i]\*h[1][0]+h[2][0];

ty[i]=(int)(ax[i]\*h[0][1]+ay[i]\*h[1][1]+h[2][1]);

tx[i]=(int)temp;

}

sleep(2);

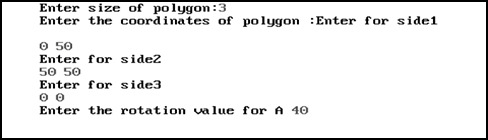
cleardevice();

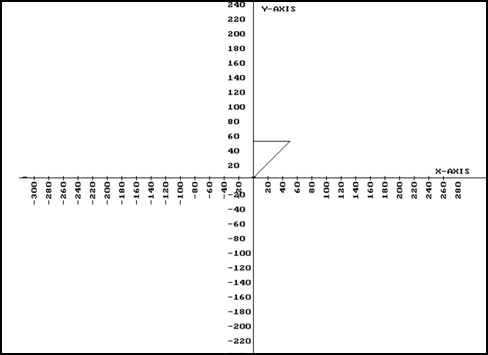
drawaxis();

polygon(tx,ty,n);

}

Output:-





**Write a program for implementation of Circle Drawing Algorithm.**

Code:-

#include<iostream.h>

#include<conio.h>

#include<dos.h>

#include<graphics.h>

#include<stdlib.h>

#include<math.h>

void main()

{

int gd=DETECT,gm;

void drawaxis(void);//funct1

void drawcircle(int,int,int,int,int);//funct2

int x1,y1,r,x,y;

initgraph(&gd,&gm,"C:\TURBOC3\BGI");

cout<<"Enter the value of Origin : ";

cin>>x>>y;

cout<<"Enter the value of Radius : ";

cin>>r;

cleardevice();

drawaxis();

drawcircle(x1,y1,r,x,y);

getch();

closegraph();

}

void drawaxis()

{

int i;

char s[25];

line(320,0,320,480);

line(0,240,640,240);

outtextxy(5,230,"X-AXIS ");

settextstyle(0,1,1);

outtextxy(336,6,"Y-AXIS");

settextstyle(0,0,1);

for(i=240;i>-240;i-=20)

{

outtextxy(318,237-1,"-");

if(i==0)

continue;

itoa(i,s,10);

outtextxy(286,240-i,s);

}

for(i=-300;i<300;i+=20)

{

itoa(i,s,10);

settextstyle(0,1,0);

outtextxy(i+300+5,236,"-");

outtextxy(i+325,245,s);

}

}

void drawcircle(int x1,int y1,int r,int x,int y)

{

int lim,d1,del,del2,z;

x1=0;

y1=r;

d1=2\*(1-r);

lim=0;

while(y1>=lim)

{

putpixel(x1+x+320,240-y1-y,RED);

putpixel(x1+x+320,y1-y+240,RED);

putpixel(320-x1+x,240+y1-y,RED);

putpixel(320-x1+x,240-y1-y,RED);

if(d1<0)

{

del=(2\*d1)+(2\*y1)-1;

if(del<=0)

{

x1=x1+1;

d1=d1+(2\*x1)+1;

}

else

{

x1=x1+1;

y1=y1-1;

d1=d1+(2\*x1)-(2\*y1)+2;

}

}

else if(d1>0)

{

del2=(2\*d1)-(2\*x1)-1;

if(del2<=0)

{

x1=x1+1;

y1=y1-1;

d1=d1+(2\*x1)-(2\*y1)+2;

}

else

{

y1=y1-1;

d1=d1-(2\*y1)+1;

}

}

if(d1==0)

{

x1=x1+1;

y1=y1-1;

d1=d1+(2\*x1)-(2\*y1)+2;

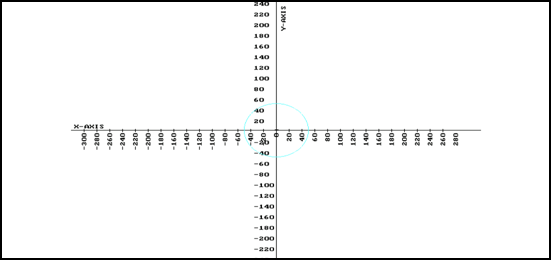
}

}

}

Output:-





**Write a program for implementation of Scale Transformation.**

Code:-

#include<iostream.h>

#include<conio.h>

#include<dos.h>

#include<graphics.h>

#include<stdlib.h>

void main()

{

int gd=DETECT,gm;

void drawaxis(void);

void polygon(int,int,int);

void dda(int,int,int,int);

void scale(void);

initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");

scale();

getch();

closegraph();

}

void drawaxis()

{

int i;

char s[25];

line(320,0,320,480);

outtextxy(5,230,"X-AXIS");

settextstyle(0,1,1);

outtextxy(336,6,"Y-AXIS");

line(0,240,640,240);

settextstyle(0,0,0);

for(i=240;i>-240;i-=20)

{

outtextxy(318,237-i,"-");

if(i==0)

continue;

itoa(i,s,10);

outtextxy(286,240 -i,s);

}

for(i=-300;i<300;i+=20)

{

itoa(i,s,10);

settextstyle(0,1,0);

outtextxy(i+325,245,s);

outtextxy(i+300+5,236,"-");

}

}

void dda(int x1,int y1,int x2,int y2)

{

int k,dx,dy,steps;

float x,y,incry,incrx;

dx=x2-x1;

dy=y2-y1;

if(abs(dx)>abs(dy))

steps=abs(dx);

else

steps=abs(dy);

incrx=(float)dx/steps;

incry=(float)dy/steps;

x=x1;

y=y1;

putpixel(320+x,240-y,WHITE);

for(k=0;k<steps;k++)

{

x+=incrx;

y+=incry;

putpixel(320+x,240-y,WHITE);

}

}

void polygon(int cx[],int cy[],int n)

{

int i;

for(i=0;i<n-1;i++)

dda(cx[i],cy[i],cx[i+1],cy[i+1]);

dda(cx[n-1],cy[n-1],cx[0],cy[0]);

}

void scale(void)

{

int ax[15],ay[15],n;

float sx,sy,h[3][2];

int tx[15]={0},ty[15]={0},temp,i,j;

cout<<"Enter the size of polygon => ";

cin>>n;

if(n<3)

{

cout<<"Invalid side error\n";

closegraph();

exit(1);

}

cout<<"Enter the co-ordinate of polygon \n";

for(i=0;i<n;i++)

{

cout<<"Enter for side"<<i+1<<"\n";

cin>>ax[i]>>ay[i];

}

cout<<"Enter the value for sx and sy =>";

cin>>sx>>sy;

cleardevice();

drawaxis();

polygon(ax,ay,n);

for(i=0;i<3;i++)

{

for(j=0;j<2;j++)

{

if(i==j)

h[i][j]=1;

else

h[i][j]=0;

}

}

for(i=0;i<3;i++)

{

h[i][0]=h[i][0]\*sx;

h[i][1]=h[i][1]\*sy;

}

for(i=0;i<n;i++)

{

temp=ax[i]\*h[0][0]+ay[i]\*h[1][0]+h[2][0];

ty[i]=(int)(ax[i]\*h[0][1]+ay[i]\*h[1][1]+h[2][1]);

tx[i]=(int)temp;

}

getch();

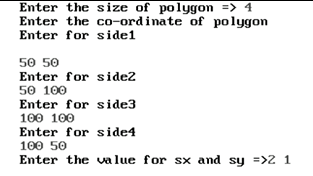
cleardevice();

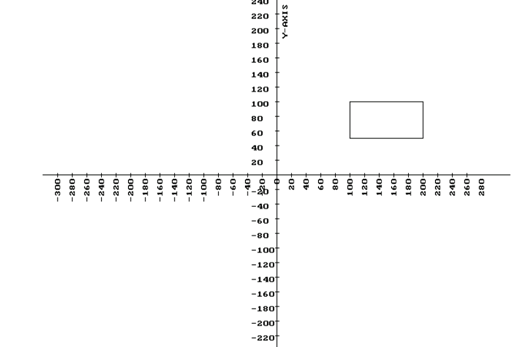
drawaxis();

polygon(tx,ty,n);

}

Output:-





**Write a program for implementation of Polymarker**

Code:-

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#include<stdio.h>

#include<stdlib.h>

void main()

{

int gd=DETECT,gm,i,j;

char s[25];

int x[10],y[10],n,midx,midy;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

midx=getmaxx()/2;

midy=getmaxy()/2;

cout<<"Enter the no. of marker :- ";

cin>>n;

for(i=0;i<n;i++)

{

cout<<"Enter the coordinate for marker ="<<i;

cin>>x[i]>>y[i];

}

clearviewport();

for(i=0;i<n;i++)

{

outtextxy(320+x[i],240-y[i],"#");

}

line(midx,0,midx,midy+midy);

line(0,midy,midx+midx,midy);

for(j=240;j>-240;j-=20)

{

outtextxy(316,236-j,"-");

if(j==0)

continue;

itoa(j,s,10);

outtextxy(287,240-j,s);

}

for(j=-280;j<300;j+=20)

{

itoa(j,s,10);

settextstyle(0,1,0);

outtextxy(j+325,245,s);

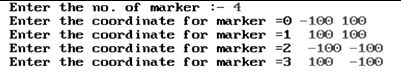
outtextxy(j+315+5,236,"-");

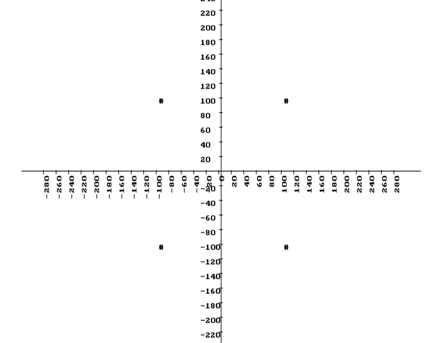
}

getch();

}

Output:-





**Write a program for implementation of Polyline**

**Code:-**

#include<iostream.h>

#include<graphics.h>

#include<stdlib.h>

#include<math.h>

#include<dos.h>

#include<conio.h>

void main()

{

int gd=DETECT,gm,i;

float n,x[10],y[10],z[10],midx,midy,x1[10],y1[10],x2[10],y2[10],xinc[10],yinc[10],length[10];

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

midx=getmaxx()/2;

midy=getmaxy()/2;

cout<<"How many line you have draw? ";

cin>>n;

for(i=1;i<=n;i++)

{

cout<<"Enter the value of x1,y1,x2,y2 for line"<<i<<":";

cin>>x1[i]>>y1[i]>>x2[i]>>y2[i];

}

clearviewport();

int j;

char s[25];

settextstyle(0,0,0);

line(0,midy,midx+midx,midy);

outtextxy(570,228,"X-AXIS");

settextstyle(0,0,0);

line(midx,0,midx,midy+midy);

outtextxy(332,6,"Y-AXIS");

settextstyle(0,0,0);

for(j=240;j>-240;j-=20)

{

outtextxy(318,237-1,"-");

if(j==0)

continue;

itoa(j,s,10);

outtextxy(286,240-j,s);

}

for(j=-300;j<300;j+=20)

{

outtextxy(j+300+5,236,"-");

if(j==0)

continue;

itoa(j,s,10);

settextstyle(0,1,0);

outtextxy(j+325,245,s);

}

for(i=1;i<=n;i++)

{

if(abs(x2[i]-x1[i])>=abs(y2[i]-y1[i]))

{

length[i]=abs(x2[i]-x1[i]);

}

else

{

length[i]=abs(y2[i]-y1[i]);

}

xinc[i]=(x2[i]-x1[i])/length[i];

yinc[i]=(y2[i]-y1[i])/length[i];

x[i]=x1[i]+0.5;

y[i]=x1[i]+0.5;

int k=1;

while(k<=length[i])

{

putpixel(x[i]+midx,z[i]+midy,GREEN);

x[i]+=xinc[i];

y[i]+=yinc[i];

z[i]=-y[i];

k++;

}

}

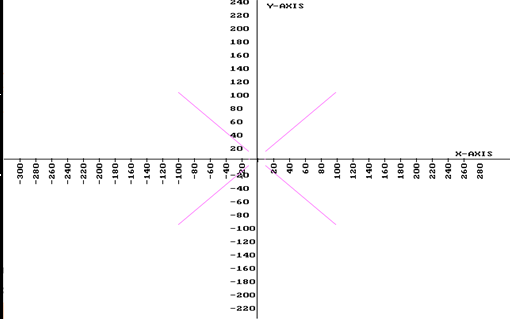
getch();

closegraph();

}

Output:-





**Write a program for implementation of Eight Connected Filling Algorithm.**

Code:-

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#include<dos.h>

#include<stdlib.h>

int n,p[10][2];

void drawpixel(void)

{

int i;

for(i=0;i<n;i++)

line(p[i][0],p[i][1],p[(i+1)%n][0],p[(i+1)%n][1]);

return;

}

void floodfilleight(int x,int y)

{

if(getpixel(x,y)!=WHITE)

{

putpixel(x,y,WHITE);

floodfilleight(x,y-1);

floodfilleight(x,y+1);

floodfilleight(x-1,y);

floodfilleight(x+1,y);

floodfilleight(x+1,y-1);

floodfilleight(x+1,y+1);;

floodfilleight(x-1,y-1);

floodfilleight(x-1,y+1);

}

return;

}

void drawaxis(void)

{

char r[25];

int i;

line(320,0,320,479);

outtextxy(590,230,"X-AXIS");

settextstyle(0,1,1);

outtextxy(330,5,"Y-AXIS");

line(0,240,639,240);

settextstyle(0,0,0);

for(i=240;i>-240;i-=20)

{

line(318,240-i,322,240-i);

itoa(i,r,10);

outtextxy(285,240-i,r);

}

for(i=-300;i<300;i+=20)

{

line(i+300,238,i+300,242);

itoa(i,r,10);

settextstyle(0,1,0);

outtextxy(i+320,243,r);

}

}

void main()

{

void drawaxis(void);

void drawpixel(void);

void floodfilleight(int,int);

int i,x,y,gd=DETECT,gm;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

cout<<"Enter the no. of point : ";

cin>>n;

cout<<"Enter the coordinate of "<<n<<"point";

for(i=0;i<n;i++)

cin>>p[i][0]>>p[i][1];

cout<<"\nEnter the interior point: ";

cin>>x>>y;

clrscr();

drawaxis();

for(i=0;i<n;i++)

{

p[i][0]=320+p[i][0];

p[i][1]=240-p[i][1];

}

x=320+x;

y=240-y;

drawpixel();

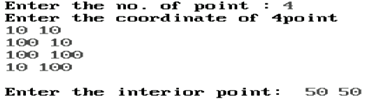
getch();

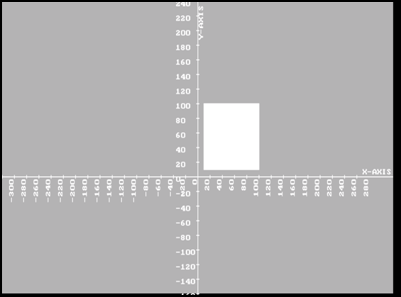
floodfilleight(x,y);

getch();

}

Output:-





**Write a program for implementation of Four Connected Algorithm.**

Code:

#include<iostream.h>

#include<graphics.h>

#include<dos.h>

#include<stdlib.h>

#include<conio.h>

int n,p[10][2];

void drawpixel(void)

{

int i;

for(i=0;i<n;i++)

line(p[i][0],p[i][1],p[(i+1)%n][0],p[(i+1)%n][1]);

return;

}

void floodfillfour(int x,int y)

{

if(getpixel(x,y)!=WHITE)

{

putpixel(x,y,WHITE);

floodfillfour(x+1,y);

floodfillfour(x-1,y);

floodfillfour(x,y+1);

floodfillfour(x,y-1);

}

return;

}

void drawaxis(void)

{

char r[25];

int i;

line(320,0,320,479);

outtextxy(590,230,"X-AXIS");

settextstyle(0,1,1);

outtextxy(330,5,"Y-AXIS");

line(0,240,639,240);

settextstyle(0,0,0);

for(i=240;i>-240;i-=20)

{

line(318,240-i,322,240-i);

itoa(i,r,10);

outtextxy(285,240-i,r);

}

for(i=-300;i<300;i+=20)

{

line(i+300,238,i+300,242);

itoa(i,r,10);

settextstyle(0,1,0);

outtextxy(i+320,243,r);

}

}

void main()

{

void drawaxis(void);

void drawpixel(void);

void floodfillfour(int,int);

int i,x,y,gd=DETECT,gm;

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

cout<<"\n\n\t Enter the no. of point :";

cin>>n;

cout<<"\t Enter the coordinate of "<<n<<"point :";

for(i=0;i<n;i++)

cin>>p[i][0]>>p[i][1];

cout<<"\n\n\t Enter the inter of point:";

cin>>x>>y;

clrscr();

drawaxis();

for(i=0;i<n;i++)

{

p[i][0]=320+p[i][0];

p[i][1]=240-p[i][1];

}

x=320+x;

y=240-y;

drawpixel();

getch();

floodfillfour(x,y);

getch();

}

Output:-

