Practical No:A1

Write C++ program to draw a concave polygon and fill it with desired color using scan fill algorithm. Apply the concept of inheritance

**Code :**

#include <conio.h>

#include <iostream>

#include <graphics.h>

#include <stdlib.h>

using namespace std;

class point

{

 public:

 int x,y;

};

class poly

{

 private:

 point p[20];

 int inter[20],x,y;

 int v,xmin,ymin,xmax,ymax;

 public:

 int c;

 void read();

 void calcs();

 void display();

 void ints(float);

 void sort(int);

};

void poly::read()

{

 int i;

 cout<<"\n Scan Fill Algorithm ";

 cout<<"\n Enter Number Of Vertices Of Polygon: ";

 cin>>v;

 if(v>2)

 {

 for(i=0;i<v; i++) //ACCEPT THE VERTICES

 {

 cout<<"\nEnter co-ordinate no. "<<i+1<<" : ";

 cout<<"\n\tx"<<(i+1)<<"=";

 cin>>p[i].x;

 cout<<"\n\ty"<<(i+1)<<"=";

 cin>>p[i].y;

 }

 p[i].x=p[0].x;

 p[i].y=p[0].y;

 xmin=xmax=p[0].x;

 ymin=ymax=p[0].y;

 }

 else

 cout<<"\n Enter valid no. of vertices.";

}

void poly::calcs()

{

 for(int i=0;i<v;i++)

 {

 if(xmin>p[i].x)

 xmin=p[i].x;

 if(xmax<p[i].x)

 xmax=p[i].x;

 if(ymin>p[i].y)

 ymin=p[i].y;

 if(ymax<p[i].y)

 ymax=p[i].y;

 }

}

void poly::display()

{

 int ch1;

 char ch='y';

 float s,s2;

 do

 {

 cout<<"\n\nMENU:";

 cout<<"\n\n\t1 . Scan line Fill ";

 cout<<"\n\n\t2 . Exit ";

 cout<<"\n\nEnter your choice:";

 cin>>ch1;

 switch(ch1)

 {

 case 1:

 s=ymin+0.01;

 delay(100);

 cleardevice();

 while(s<=ymax)

 {

 ints(s);

 sort(s);

 s++;

 }

 break;

 case 2:

 exit(0);

 }

 cout<<"Do you want to continue?: ";

 cin>>ch;

 }while(ch=='y' || ch=='Y');

}

void poly::ints(float z)

{

 int x1,x2,y1,y2,temp;

 c=0;

 for(int i=0;i<v;i++)

 {

 x1=p[i].x;

 y1=p[i].y;

 x2=p[i+1].x;

 y2=p[i+1].y;

 if(y2<y1)

 {

 temp=x1;

 x1=x2;

 x2=temp;

 temp=y1;

 y1=y2;

 y2=temp;

 }

 if(z<=y2&&z>=y1)

 {

 if((y1-y2)==0)

 x=x1;

 else

 {

 x=((x2-x1)\*(z-y1))/(y2-y1);

 x=x+x1;

 }

 if(x<=xmax && x>=xmin)

 inter[c++]=x;

 }

 }

}

void poly::sort(int z) // sorting

{

 int temp,j,i;

 for(i=0;i<v;i++)

 {

 line(p[i].x,p[i].y,p[i+1].x,p[i+1].y);

 }

 delay(100);

 for(i=0; i<c;i+=2)

 {

 delay(100);

 line(inter[i],z,inter[i+1],z);

 }

}

int main() //main

{

 int cl;

 initwindow(500,600);

 cleardevice();

 poly x;

 x.read();

 x.calcs();

 cleardevice();

 cout<<"\n\tEnter The Color You Want :(In Range 0 To 15 )->"; //selecting color

 cin>>cl;

 setcolor(cl);

 x.display();

 closegraph(); //closing graph

 getch();

 return 0;

}

**Input :**

Number of Vertices : 4

Cordinates 1st :

x1= 200

y1= 200

Cordinates 2st :

x2= 200

y2= 400

Cordinates 3st :

x3= 400

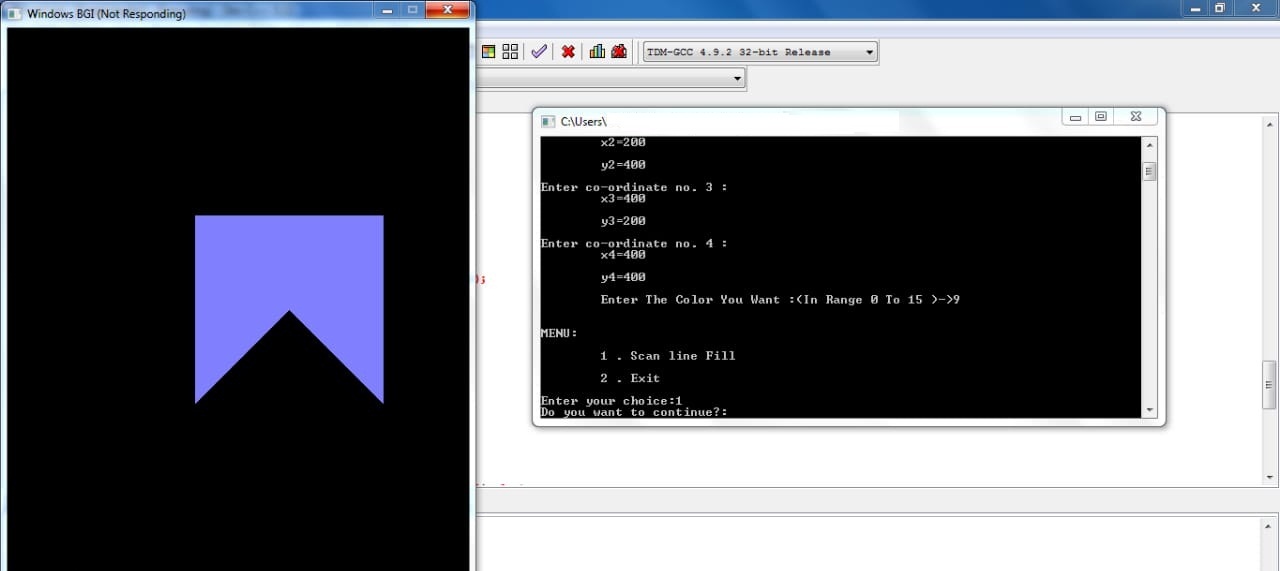
y3= 200

Cordinates 4st :

x4= 400

y4= 400

**Output :**

[](https://1.bp.blogspot.com/-LpN0i5yjq4c/X7OYVPM1AwI/AAAAAAAAAN0/DQ2Y9Yeh5QAx4JY_OkBO_3p5fYUnHesCwCLcBGAsYHQ/s1280/concave.jpeg)