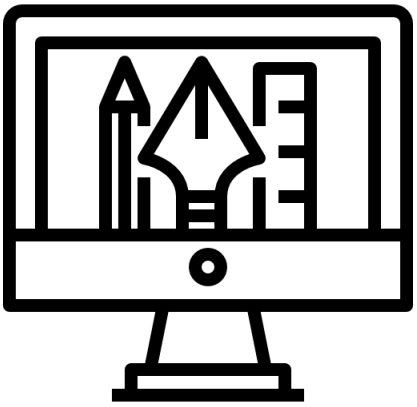
ASSIGNMENT

ON

COMPUTER GRAPHICS



Course Code : CSE3222
Course Title : Computer Graphics Lab

Submitted By	Submitted To
Akif Islam SID: 1910776135 Session: 2018-19 3rd Year Even Semester Department of CSE University of Rajshahi	Abu Raihan Shoyeb Ahmed Siddique Professor Department of CSE University of Rajshahi

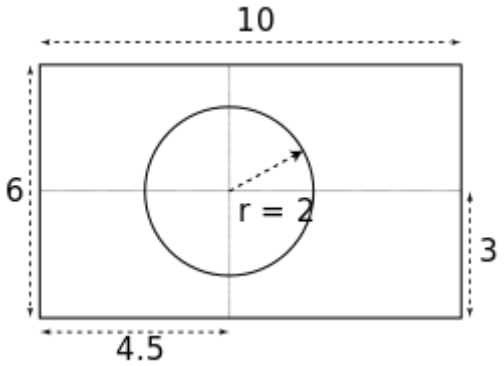
Table of Contents

Experiment 01 - Draw the National Flag of Bangladesh

```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

int main(){

   int gd = DETECT;
   int gm = DETECT;

   initgraph(&gd,&gm,"");

   // Setting Rectangle Parameter by maintaining width:height = 10:6 ratio
   int scale_factor = 30;

   int x1 = 50;
   int y1 = 50;

   int x2 = x1+10*scale_factor;
   int y2 = y1+6*scale_factor;

   // Drawing Rectangle
   setcolor(GREEN);
   setfillstyle(SOLID_FILL,GREEN);
   rectangle(x1,y1,x2,y2);
   floodfill(x1+1,y1+1,GREEN);

   //Drawing Circle
   setcolor(RED);
   setfillstyle(SOLID_FILL,RED);
   circle(x1+(x2-x1)*0.45,y1+(y2-y1)*0.5, 2*scale_factor);
   floodfill(x1+(x2-x1)*0.45+1,y1+(y2-y1)*0.5+1,RED);
```

```
    //Drawing a Handle
    setcolor(WHITE);
    setfillstyle(SOLID_FILL,WHITE);
    rectangle(x1-max(scale_factor/3,10),y1,x1,y2*2);

    getch();

}
```

Measurement used to Draw the Flag

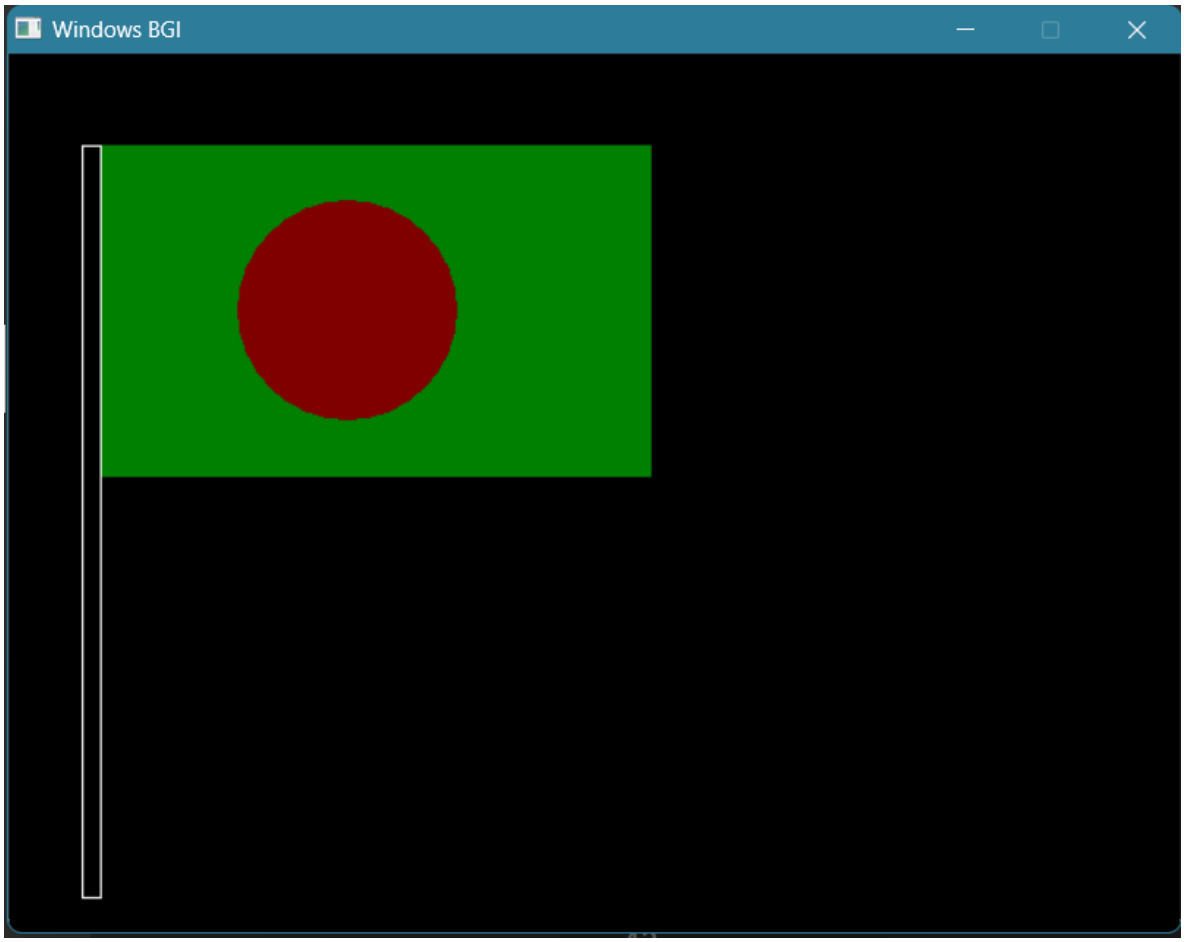


Input

For input, the user has to write the starting point of the flag. This is basically the (x1,y1) value of the upper left corner of the green triangle.

```
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-wm22iycq.x45' '--stdout=Microsoft-MIEngine-Out-lmli3pmj.hc2' '--stderr=Microsoft-MIEngine-Error-xdkhdbxm.ask' '--pid=Microsoft-MIEngine-Pid-erg0rbvo.ghk' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter the start point of the flag (x1,y1):50 50
```

Output



Experiment 02 - Simulate Two Dimensional Geometric Translation, Rotation & Scaling

Translation

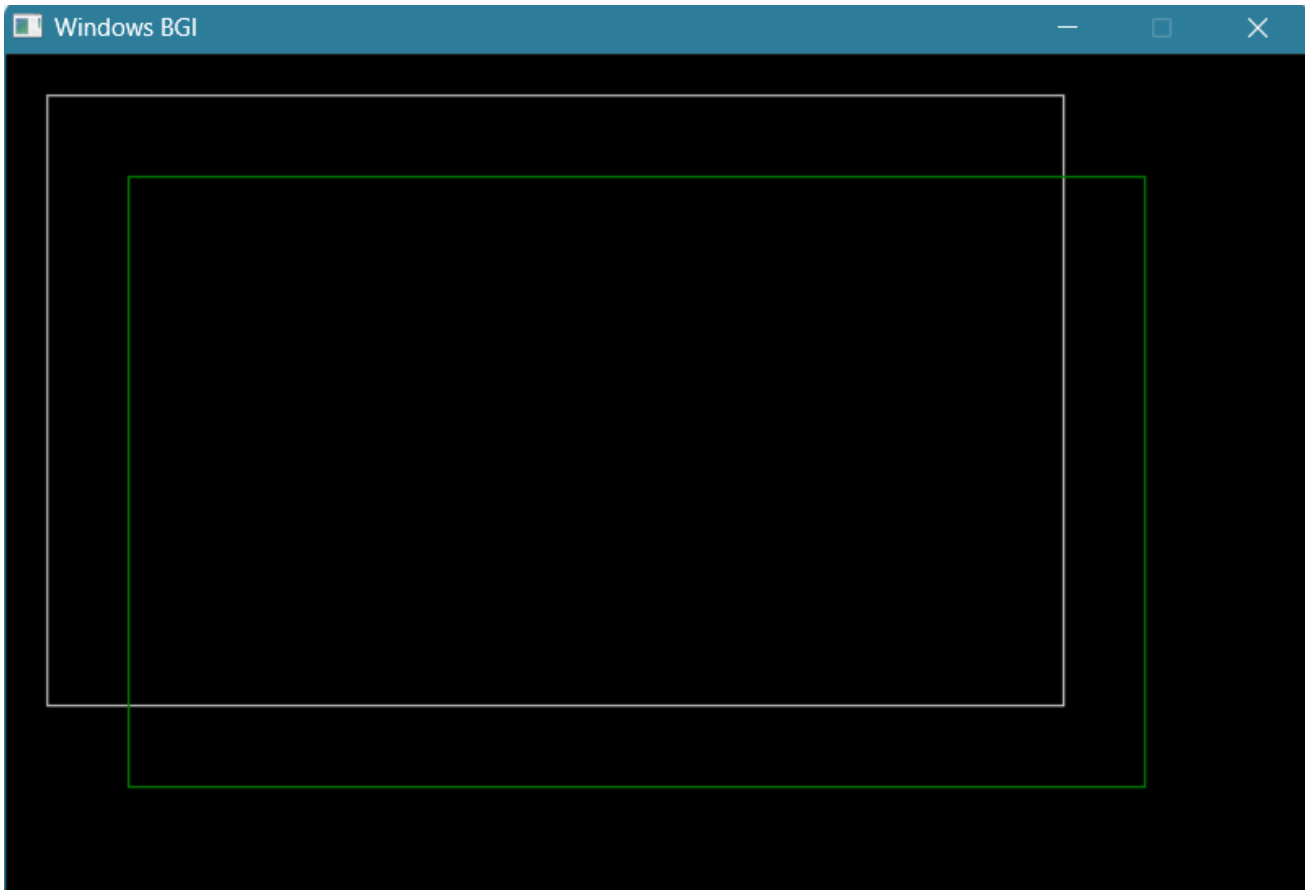
```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

void draw_polygon(vector<pair<int,int>> poly,int color){

   setcolor(color);

   for (int i = 0; i < poly.size(); i++)
      line(poly[i].first,poly[i].second,poly[(i+1)%poly.size()].first,
      poly[(i+1)%poly.size()].second);

}

int main(){

   int gd = DETECT;
   int gm = DETECT;

   initgraph(&gd,&gm,"");

   // Taking No of Sides of a Polygon
   cout<<"Enter No of  Side : ";
   int no_of_side;
   cin>>no_of_side;

   // Taking coordinates of the sides
   vector<pair<int,int>> coordinates;
```

```
for (int i = 0; i < no_of_side; i++)
   {
      int x,y;
      cout<<"Enter (X"<<i+1<<",Y"<<i+1<<"):";
      cin>>x>>y;
      coordinates.push_back({x,y});

   }

   //Taking Translation Factor
   cout<<"Enter Translation Factor (Tx,Ty) :";
   int tx,ty;
   cin>>tx>>ty;

   draw_polygon(coordinates,7);

   //Translate all points
   for (int i = 0; i < coordinates.size(); i++)
   {
      coordinates[i].first+=tx;
      coordinates[i].second+=ty;
   }

   draw_polygon(coordinates,2);
   getch();

}
```

Input

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-n15v2th0.ijt' '--stdout=Microsoft-MIEngine-Out-ksh45z0m.eau' '--stderr=Microsoft-MIEngine-Error-llu0ut1l.5nb' '--pid=Microsoft-MIEngine-Pid-3al2ffbx.pbw' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter No of  Side : 4
Enter (X1,Y1):20 20
Enter (X2,Y2):20 320
Enter (X3,Y3):520 320
Enter (X4,Y4):520 20
Enter Translation Factor (Tx,Ty) :40 40
```

Output



Rotation

```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

double to_radian(int degree){
   return (degree*3.1416)/180;
}

void draw_polygon(vector<pair<int,int>> poly,int color){

   setcolor(color);

   for (int i = 0; i < poly.size(); i++)
      line(poly[i].first,poly[i].second,poly[(i+1)%poly.size()].first,
       poly[(i+1)%poly.size()].second);

}

int main(){

   int gd = DETECT;
   int gm = DETECT;

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

   // Taking No of Sides of a Polygon
   cout<<"Enter No of  Side : ";
   int no_of_side;
   cin>>no_of_side;

   // Taking co-ordinates of the sides
   vector<pair<int,int>> coordinates;

   for (int i = 0; i < no_of_side; i++)
```

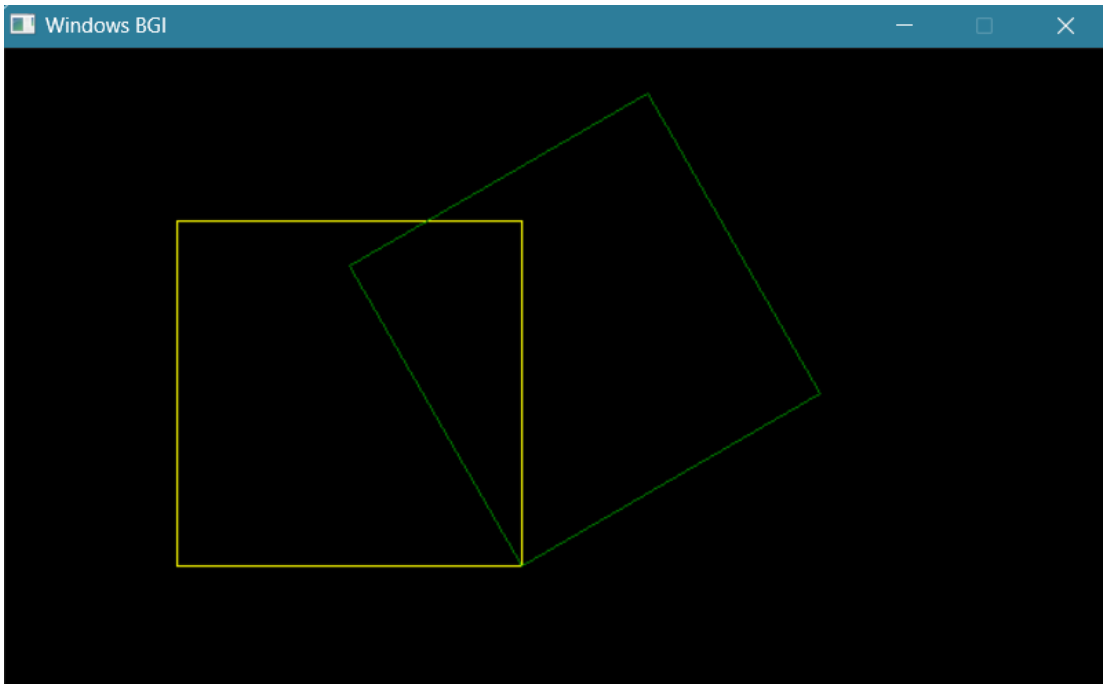
```
{
    int x,y;
    cout<<"Enter (X"<<i+1<<",Y"<<i+1<<"):";
    cin>>x>>y;
    coordinates.push_back({x,y});

}


cout<<"Rotation Angle in Degree : ";
int angle;
cin>>angle;

//Taking Scaling Factor
cout<<"Enter Pivot Point for Rotation (rx,ry) :";
int rx,ry;
cin>>rx>>ry;


draw_polygon(coordinates,14);

//Translate to Origin, Then Rotate, Then Back to Pivot Point
for (int i = 0; i < coordinates.size(); i++)
{
    double x = coordinates[i].first;
    double y = coordinates[i].second;

    //Move to Origin
    int x_shift=x-rx;
    int y_shift=y-ry;

    //Rotate
    x = x_shift*cos(to_radian(angle))-y_shift*sin(to_radian(angle));
    y = x_shift*sin(to_radian(angle))+y_shift*cos(to_radian(angle));

    //Back to Pivot Point
    x+=rx;
```

```
        y+=ry;

        coordinates[i].first = x;
        coordinates[i].second = y;
    }

    draw_polygon(coordinates,2);
    getch();

}
```

Input

```
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-jk02dp3l.y5x' '--stdout=Microsoft-MIEngine-Out-qujsqjcy.wry' '--stderr=Microsoft-MIEngine-Error-tstl0b44.qnl' '--pid=Microsoft-MIEngine-Pid-xql3nxbs.dsa' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter No of  Side : 4
Enter (X1,Y1):100 100
Enter (X2,Y2):100 300
Enter (X3,Y3):300 300
Enter (X4,Y4):300 100
Rotation Angle in Degree : 60
Enter Pivot Point for Rotation (rx,ry) :300 300
```

Output



Scaling

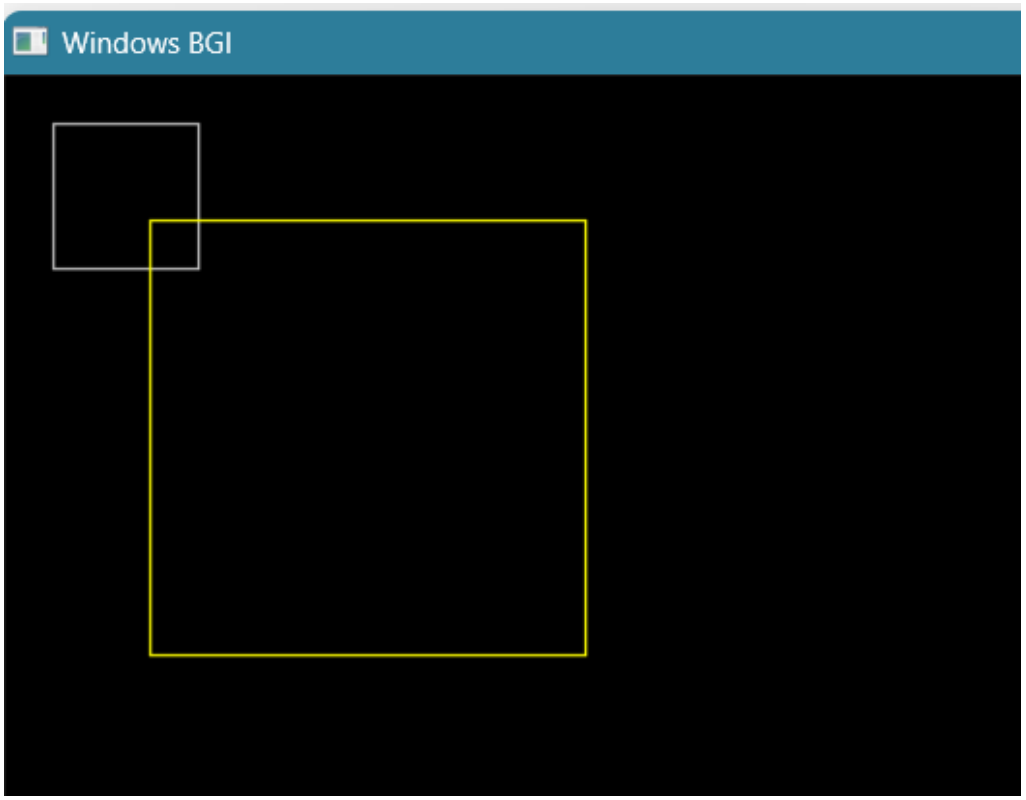
```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

void draw_polygon(vector<pair<int,int>> poly,int color){

   setcolor(color);

   for (int i = 0; i < poly.size(); i++)
      line(poly[i].first,poly[i].second,poly[(i+1)%poly.size()].first,
      poly[(i+1)%poly.size()].second);

}

int main(){

   int gd = DETECT;
```

```
    int gm = DETECT;

    initgraph(&gd,&gm,"");

    // Taking No of Sides of a Polygon
    cout<<"Enter No of  Side : ";
    int no_of_side;
    cin>>no_of_side;

    // Taking co-ordinates of the sides
    vector<pair<int,int>> coordinates;

    for (int i = 0; i < no_of_side; i++)
    {
        int x,y;
        cout<<"Enter (X"<<i+1<<",Y"<<i+1<<"):";
        cin>>x>>y;
        coordinates.push_back({x,y});

    }

    //Taking Scaling Factor
    cout<<"Enter Scaling Factor (Sx,Sy) :";
    int sx,sy;
    cin>>sx>>sy;

    draw_polygon(coordinates,7);
    //Translate all points
    for (int i = 0; i < coordinates.size(); i++)
    {
        coordinates[i].first*=sx;
        coordinates[i].second*=sy;
    }

    draw_polygon(coordinates,14);
    getch();
}
```

Input

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-nmben5jn.l1r' '--stdout=Microsoft-MIEngine-Out-1qjrhtit.0wh' '--stderr=Microsoft-MIEngine-Error-wtpxcglj.4v0' '--pid=Microsoft-MIEngine-Pid-fwvd2uge.4w4' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter No of  Side : 4
Enter (X1,Y1):20 20
Enter (X2,Y2):20 80
Enter (X3,Y3):80 80
Enter (X4,Y4):80 20
Enter Scaling Factor (Sx,Sy) :3 3
```

Output



Experiment 03- Simulate Hidden Surface Elimination

```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

void Triangle()
{
   int x[] = {10, 50, 100};
   int y[] = {100, 20, 100};

   setcolor(YELLOW);

   for (int i = 0; i < 3; i++)
   {
      line(x[i], y[i], x[(i + 1) % 3], y[(i + 1) % 3]);
   }
   setfillstyle(SOLID_FILL, YELLOW);
   floodfill(50, 25, YELLOW);
}

void Circle()
{
   setcolor(MAGENTA);
   circle(100, 100, 45);
   setfillstyle(SOLID_FILL, MAGENTA);
   floodfill(101, 101, MAGENTA);
}

void Rectangle()
{
   setcolor(GREEN);
   rectangle(100-20, 100-20, 180, 180);
   setfillstyle(SOLID_FILL, GREEN);
   floodfill(101, 101, GREEN);
```

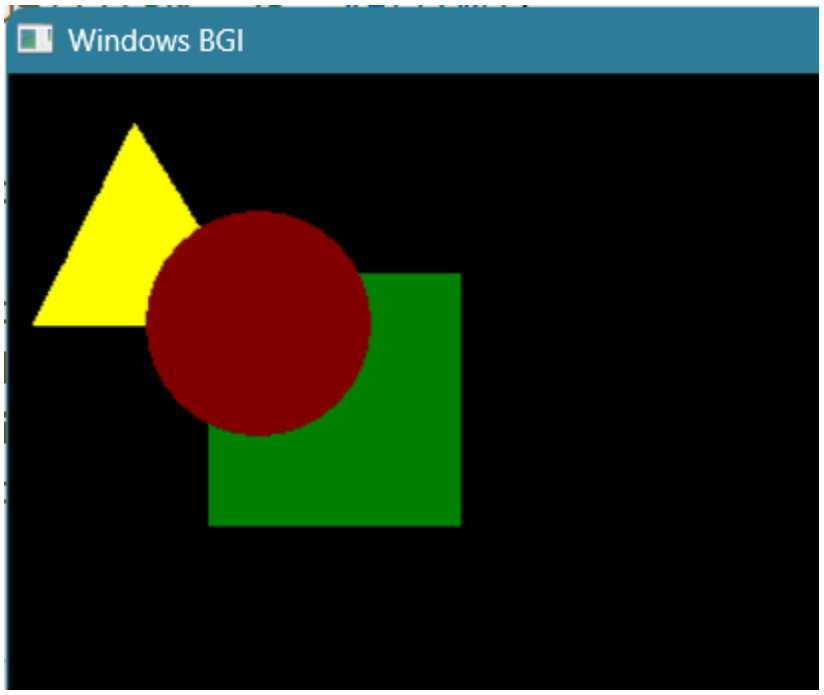
```
}

int main()
{
   string sequence;
   cin >> sequence;
   int gd = DETECT;
   int gm = DETECT;
   initgraph(&gd, &gm, "");

   for (int i = 0; i < sequence.size(); i++)
   {
      if(sequence[i]=='C')
         Circle();
      else if(sequence[i]=='R')
         Rectangle();
      else
         Triangle();
   }


   getch();
}
```

Input

The user has to type a string with 3 characters. Each of the letters defines a shape (Ex. R for Rectangle, C for Circle, T for Triangle)

```
PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-rl4hutjn.tzz' '--stdout=Microsoft-MIEngine-Out-vf35cwlj.nlr' '--stderr=Microsoft-MIEngine-Error-kz1ccdd1.x1t' '--pid=Microsoft-MIEngine-Pid-khtzhqn5.enf' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
RTC
```

Output



Experiment 04 - Draw a line with Bresenham Line Drawing Algorithm

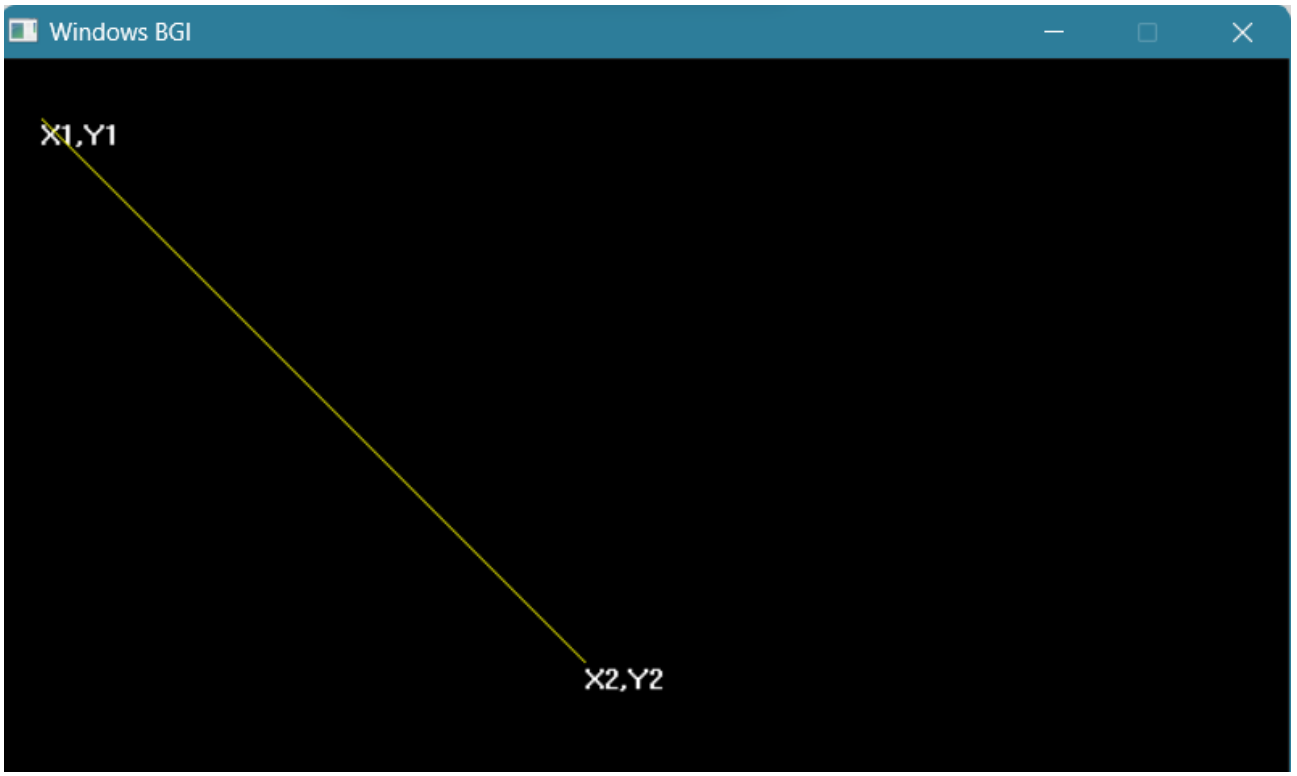
```
#include<bits/stdc++.h>
#include<graphics.h>

using namespace std;

int main(){
   int gd = DETECT;
   int gm = DETECT;
   initgraph(&gd,&gm," ");

   cout<<"Enter (x1,y1):";
   int x1,y1;
   cin>>x1>>y1;

   cout<<"Enter (x2,y2):";
   int x2,y2;
   cin>>x2>>y2;

   int dx = x2-x1;
   int dy = y2-y1;

   int cur_x = x1;
   int cur_y = y1;
   outtextxy(cur_x,cur_y,"X1,Y1");
   putpixel(cur_x,cur_y,14);
   int P = 2*dy-dx;

   while(cur_x<x2 || cur_y<y2){
      if(P<0){
         cur_x++;
         putpixel(cur_x,cur_y,14);
         P+=2*dy;
      }
```

```
        else{
            cur_x++;
            cur_y++;
            putpixel(cur_x,cur_y,14);
            P+=2*dy-2*dx;
        }


    }
    outtextxy(cur_x,cur_y,"X2,Y2");
    getch();

}
```

Input & Output #1

A line from (20,30) to (20,300)

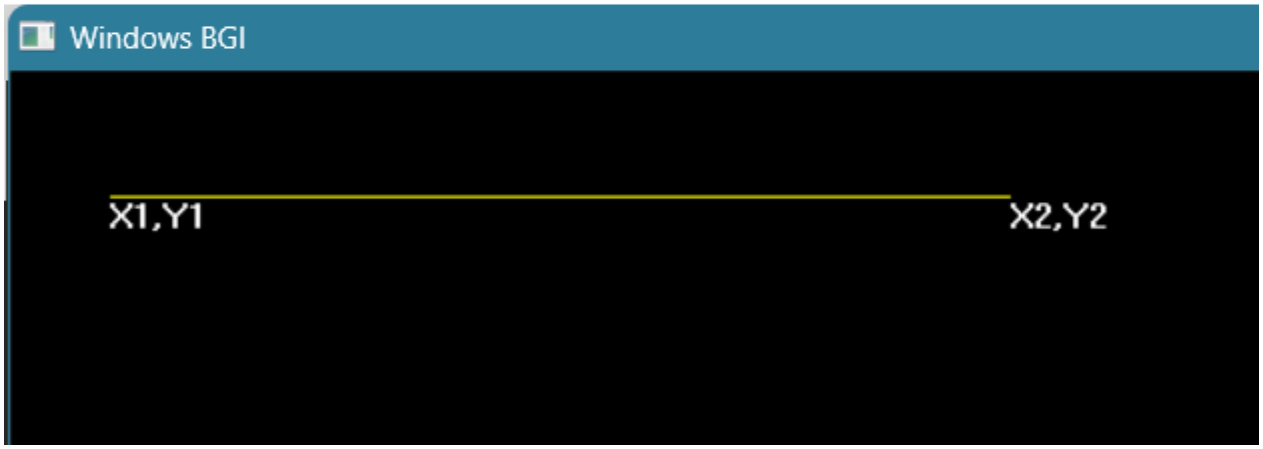
```
PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-0llgsvyt.jpf' '--stdout=Microsoft-MIEngine-Out-schfbxy2.rh0' '--stderr=Microsoft-MIEngine-Error-mohnym5i.ach' '--pid=Microsoft-MIEngine-Pid-r2z2qqku.5iy' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter (x1,y1):20 30
Enter (x2,y2):20 300
```



Input & Output #2

A line from (40,50) to (400,50)

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-uempqx14.k1y' '--stdout=Microsoft-MIEngine-Out-11oknsf5.ixa' '--stderr=Microsoft-MIEngine-Error-gsbe4pon.xg3' '--pid=Microsoft-MIEngine-Pid-wuuikwwo.jjd' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
Enter (x1,y1):40 50
Enter (x2,y2):400 50
```



Experiment 05 - Draw a Circle with Midpoint Circle Drawing Algorithm

```
#include <bits/stdc++.h>
#include <graphics.h>

using namespace std;

int main()
{

   int gd, gm;
   gd = DETECT, gm = DETECT;
   initgraph(&gd, &gm, "");

   // Input Parameter
   int r = 150;
   int x = 200;
   int y = 200;

   cin>>x>>y>>r;

   // Process
   int P = 1 - r;

   int cur_x = 0, cur_y = r;

   vector<pair<int, int>> points;

   while (cur_x < cur_y)
   {
      points.push_back({cur_x, cur_y});
      points.push_back({cur_y, cur_x});

      cur_x++;
```

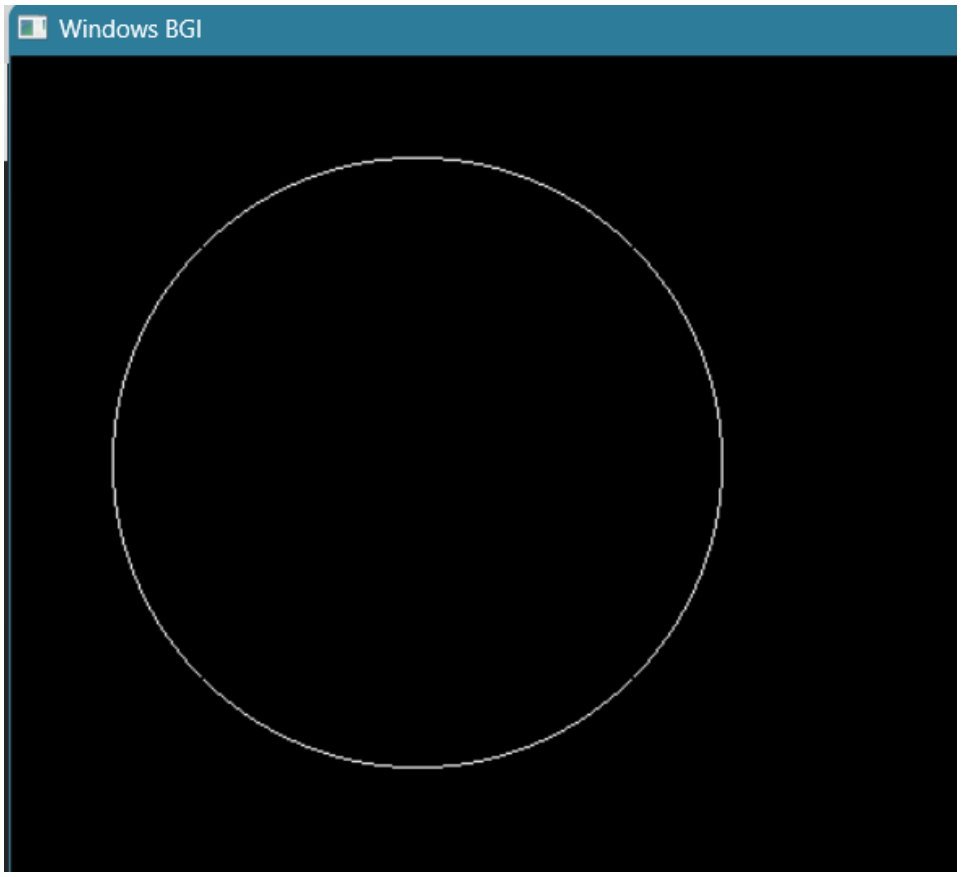
```
        if (P < 0)
        {
            P += 2 * cur_x + 1;
        }
        else
        {
            cur_y--;
            P += 2 * (cur_x - cur_y) + 1;
        }
    }

    for (int i = 0; i < points.size(); i++)
    {
        putpixel(x + points[i].first, y + points[i].second, WHITE);
        putpixel(x - points[i].first, y + points[i].second, WHITE);
        putpixel(x - points[i].first, y - points[i].second, WHITE);
        putpixel(x + points[i].first, y - points[i].second, WHITE);


    }

    getch();
}
```

Input

The user has to input three parameters, the center of the circle (x), the center of the circle (y), the radius of the circle.

```
PS C:\Users\Akif\Desktop\Graphics-Lab>  & 'c:\Users\Akif\.vscode\extensions\ms-vscode.cpptools-1.16.3-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-1brjnnj2.jxr' '--stdout=Microsoft-MIEngine-Out-uxkkd3x3.dzv' '--stderr=Microsoft-MIEngine-Error-pogywpnl.q10' '--pid=Microsoft-MIEngine-Pid-1vo3novx.zwk' '--dbgExe=C:\TDM-GCC-32\gdb32\bin\gdb32.exe' '--interpreter=mi'
200 200 150
```

Output



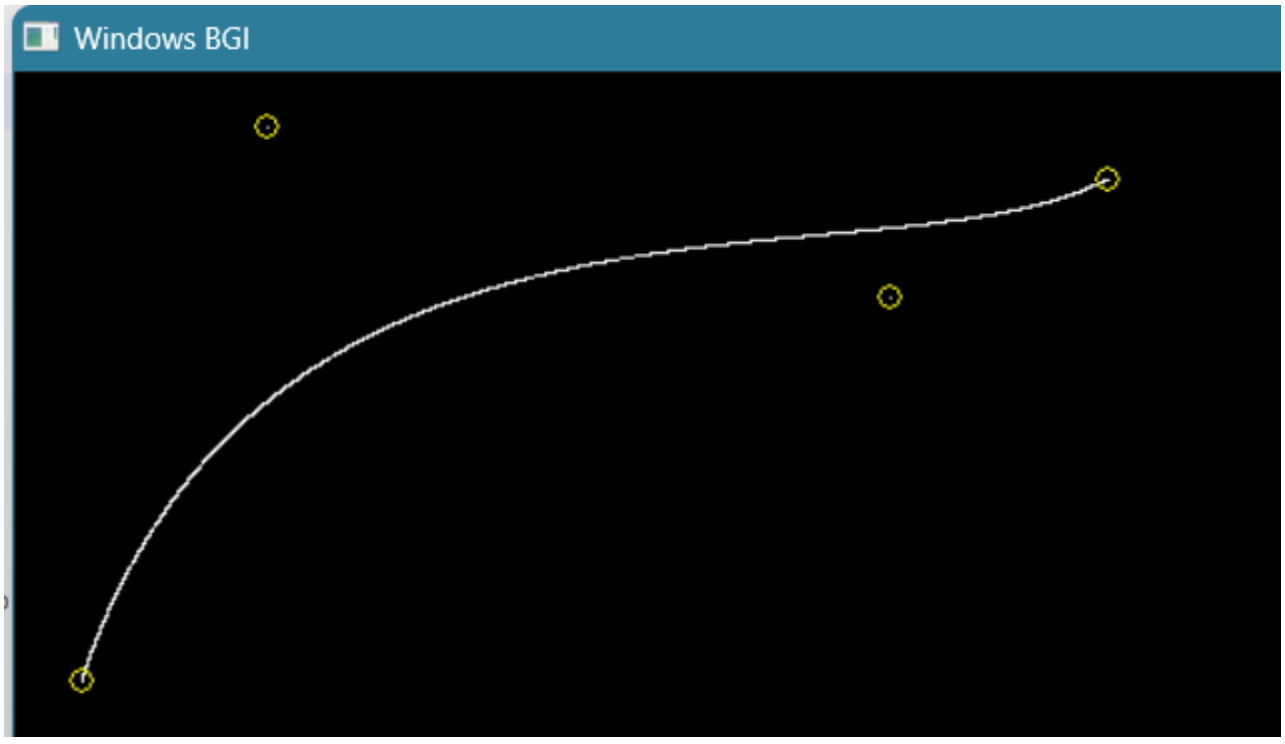
Experiment 06 - Draw a Curve with Bezier Curve Algorithm

```
#include <bits/stdc++.h>
#include <graphics.h>

using namespace std;

int factorial(int n)
{
   if(n<2)
      return 1;
   return n*factorial(n-1);
}
```

```
double nCr(int n, int r)
{
   return (double)(factorial(n)/(factorial(r)*factorial(n-r)));
}

double bezierFunction(int k, int n, double u)
{
   return nCr(n, k) * pow(u, k) * pow((1 - u), (n - k));
}

void bezierCurve(vector<pair<int, int>> points)
{
   setcolor(YELLOW);
   int n = points.size() - 1;
   double eps = 0.0001;

   for (double u = 0; u <= 1; u += eps)
   {
      double x = 0, y = 0;

      for (int k = 0; k <= n; k++)
      {
         double bez = bezierFunction(k, n, u);
         x += points[k].first * bez;
         y += points[k].second * bez;
      }

      putpixel(x, y, WHITE);
   }
   for (auto x: points)
   {
      putpixel(x.first, x.second, WHITE);
      circle(x.first, x.second, 5);
   }
}
```

```
int main()
{
   int gd = DETECT, gm = DETECT;
   initgraph(&gd, &gm, "");

   vector<pair<int, int>> points = {{27, 243}, {101, 22}, {350, 90}, {437, 43}};
   bezierCurve(points);

   getch();
   closegraph();
   return 0;
}
```

Output



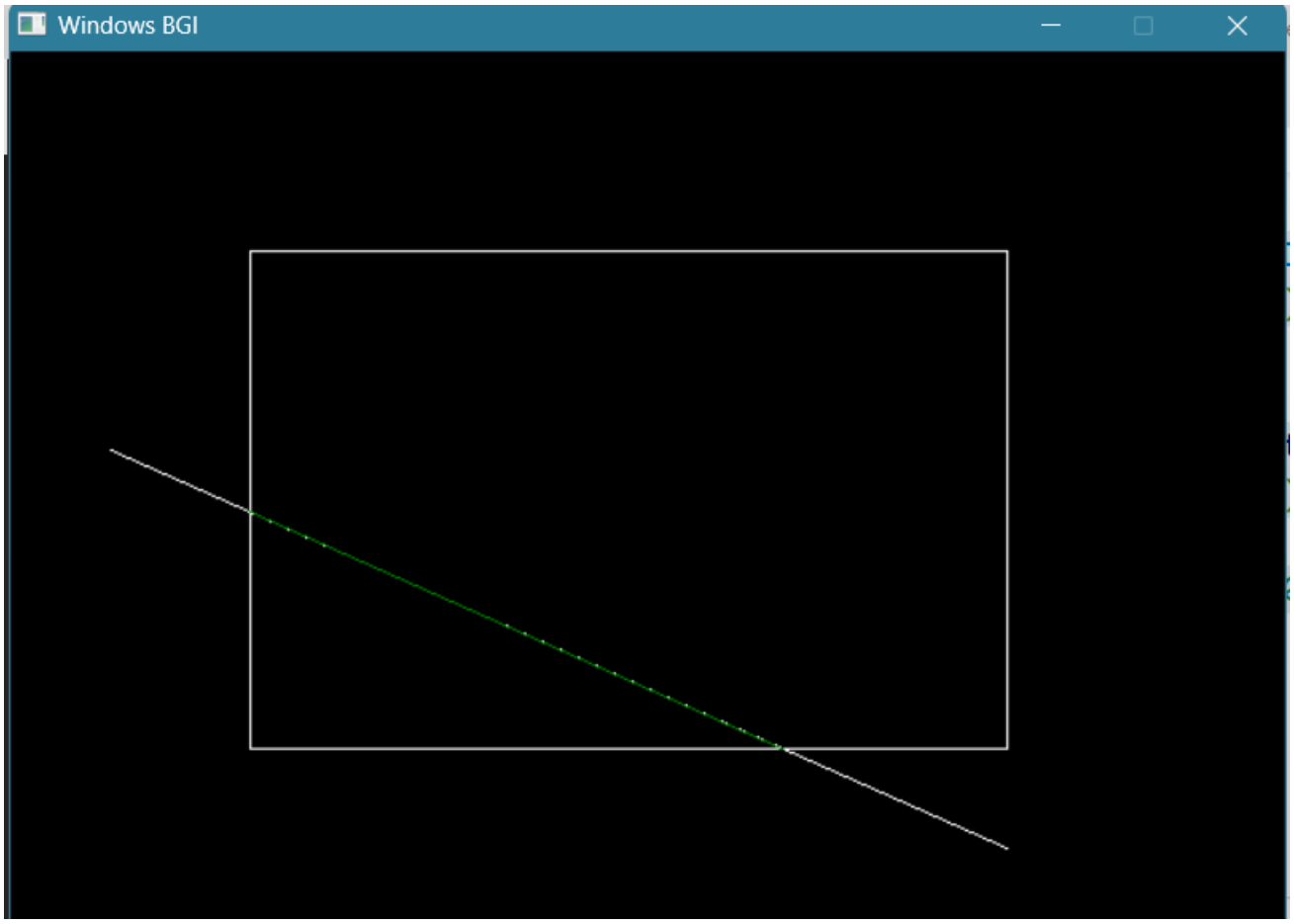
Experiment 07 - Simulate Cohen Sutherland Line Clipping Algorithm

```
#include <graphics.h>
#include <bits/stdc++.h>

using namespace std;

double x_left = 120, x_right = 500, y_bottom = 100, y_top = 350; //... Clipping window
int Left = 1, Right = 2, Bottom = 4, Top = 8; //... Region code

int regionCode(int x, int y)
{
   int code = 0;
   if (x > x_right) code |= Right;
   else if (x < x_left) code |= Left;
   if (y > y_top) code |= Top;
   else if (y < y_bottom) code |= Bottom;
   return code;
}

void cohenSutherland(double x1, double y1, double x2, double y2)
{
   int code1 = regionCode(x1, y1);
   int code2 = regionCode(x2, y2);
   while (true)
   {
      double x, y;
      if (!(code1 | code2)) //... Line is completely inside
      {
         line(x1, y1, x2, y2);
         return;
      }
      else if (code1 & code2) break; //... Line is completely outside
      else //... Line is partially inside
      {
```

```
        int code = code1 ? code1 : code2;
        if (code & Top)
        {
            y = y_top;
            x = x1 + (x2 - x1) / (y2 - y1) * (y - y1);
        }
        else if (code & Bottom)
        {
            y = y_bottom;
            x = x1 + (x2 - x1) / (y2 - y1) * (y - y1);
        }
        else if (code & Left)
        {
            x = x_left;
            y = y1 + (y2 - y1) / (x2 - x1) * (x - x1);
        }
        else if (code & Right)
        {
            x = x_right;
            y = y1 + (y2 - y1) / (x2 - x1) * (x - x1);
        }
        if (code == code1)
        {
            x1 = x;
            y1 = y;
            code1 = regionCode(x1, y1);
        }
        else
        {
            x2 = x;
            y2 = y;
            code2 = regionCode(x2, y2);
        }
    }
  }
}
```

```
int main()
{
   int gd = DETECT, gm = DETECT;
   initgraph(&gd, &gm, "");

   setcolor(WHITE);
   rectangle(x_left, y_bottom, x_right, y_top);
   line(50, 200, 500, 400);
   setcolor(GREEN);
   cohenSutherland(50, 200, 500, 400);

   getch();
   closegraph();
   return 0;
}
```

Output



Experiment 08 - Draw the Snowflake Pattern with Fractal Geometry

```
#include <graphics.h>
#include <bits/stdc++.h>

using namespace std;

void snowfalke(int x1, int y1, int x5, int y5, int it)
{
   if (it)
   {
      vector<pair<int, int>> x(5);
      int dx = (x5 - x1) / 3, dy = (y5 - y1) / 3;
      x[0] = {x1, y1};
      x[1] = {x1 + dx, y1 + dy};
      x[2] = {(x1 + x5) / 2 + sqrt(3) * (y1 - y5) / 6, (y1 + y5) / 2 + sqrt(3) * (x5 - x1) / 6};
      x[3] = {x1 + 2 * dx, y1 + 2 * dy};
      x[4] = {x5, y5};
      for (int i = 0; i < 4; i++)
      {
         snowfalke(x[i].first, x[i].second, x[i + 1].first, x[i + 1].second, it - 1);
      }
   }
   else line (x1, y1, x5, y5);
   delay(5);
}

int main()
{
   int gd = DETECT, gm = DETECT;
   initgraph(&gd, &gm, "");

   int iteration = 4;
   vector<pair<int, int>> x = {{250, 15}, {50, 350}, {450, 350}};
   for (int i = 0; i < 3; i++)
```

```
    {
        snowfalke(x[i].first, x[i].second, x[(i + 1) % 3].first, x[(i + 1) % 3].second, iteration);
    }

    getch();
    closegraph();
    return 0;
}
```

Output