**Experiment -1:**  **Scan Conversion of a point.**

**Code:**

#include<iostream>

using namespace std;

char screen[80][25]={};

void scanConvertPoint(int x,int y)

{

screen[x][y]='\*';

}

int main()

{

int x,y;

cout<<"Enter X axis point: ";

cin>>x;

cout<<"\nEnter Y axis point: ";

cin>>y;

scanConvertPoint(x,y);

for(int i=0;i<25;i++)

{

for(int j=0;j<80;j++)

{

cout<<screen[j][i];

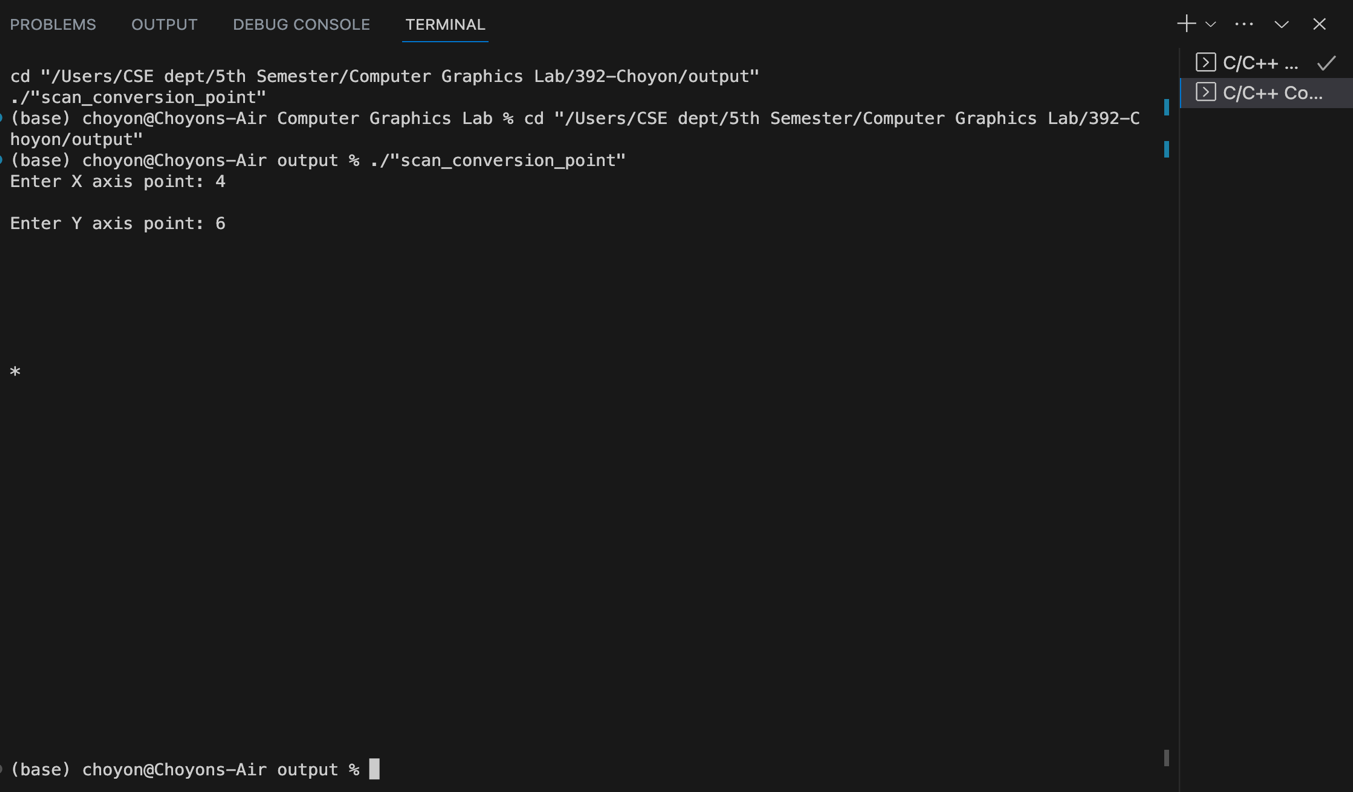
}

cout<<endl;

}

return 0;

}



**Experiment -2:**  **Scan conversion of a line using DDA algorithm.**

**Code:**

#include <iostream>

#include <bits/stdc++.h>

using namespace std;

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

{

int dx = x2 - x1;

int dy = y2 - y1;

int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

float xIncrement = static\_cast<float>(dx) / steps;

float yIncrement = static\_cast<float>(dy) / steps;

float x = x1;

float y = y1;

for (int i = 0; i <= steps; i++) {

int roundedX = static\_cast<int>(round(x));

int roundedY = static\_cast<int>(round(y));

cout << "(" << roundedX << ", " << roundedY << ")" << endl;

x += xIncrement;

y += yIncrement;

}

}

int main()

{

int x1 = 0;

int y1 = 0;

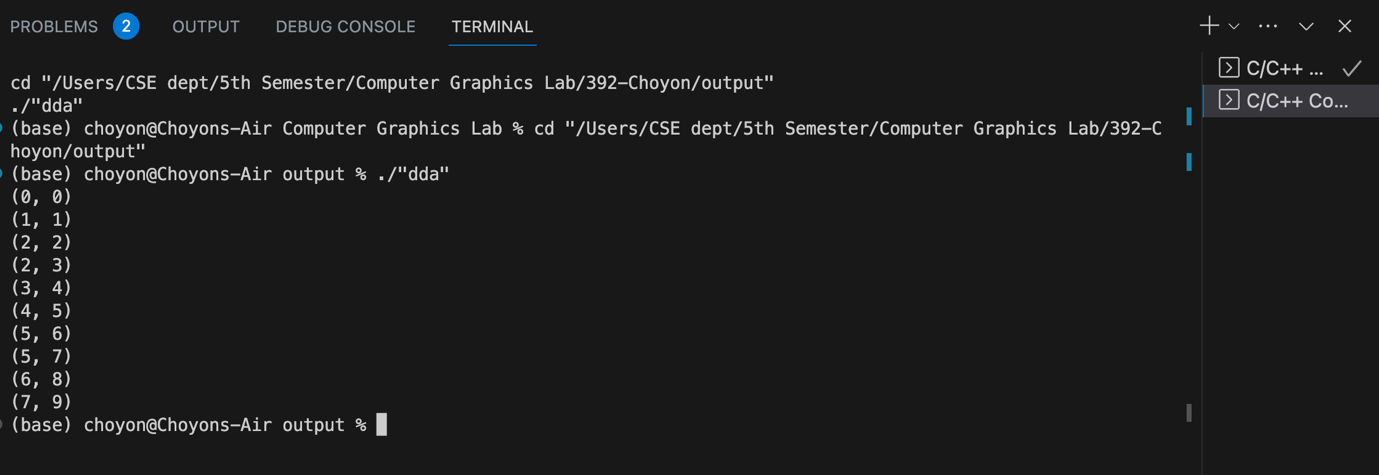
int x2 = 7;

int y2 = 9;

scanConvertLine(x1, y1, x2, y2);

return 0;

}



**Experiment -3:**  **Scan Conversion of a line using Bressenham’s algorithm.**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

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

{

int dx = abs(x2 - x1);

int dy = abs(y2 - y1);

int sx = (x1 < x2) ? 1 : -1;

int sy = (y1 < y2) ? 1 : -1;

int err = dx - dy;

const int width = 80;

const int height = 24;

char grid[height][width];

for (int i = 0; i < height; ++i)

{

for (int j = 0; j < width; ++j)

{

grid[i][j] = ' ';

}

}

int x = x1;

int y = y1;

while (x != x2 || y != y2)

{

if (x >= 0 && x < width && y >= 0 && y < height)

{

grid[y][x] = '\*';

}

int e2 = 2 \* err;

if (e2 > -dy)

{

err -= dy;

x += sx;

}

if (e2 < dx)

{

err += dx;

y += sy;

}

}

for (int i = 0; i < height; ++i)

{

for (int j = 0; j < width; ++j)

{

cout << grid[i][j];

}

cout << endl;

}

}

int main()

{

cout<<"Scan converting a line using Bresenham algorithm";

int x1 = 10;

int y1 = 5;

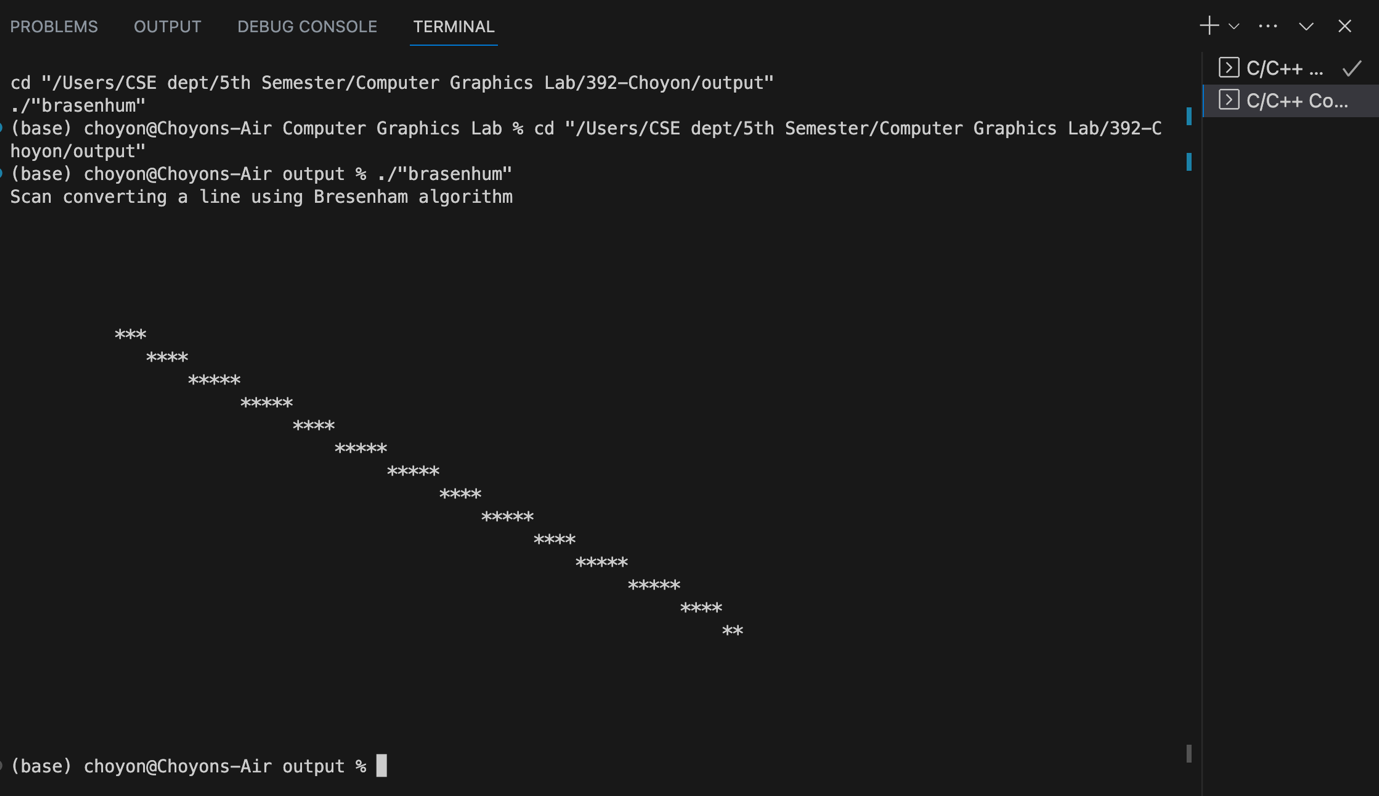
int x2 = 70;

int y2 = 18;

scanConvertLineBresenham(x1, y1, x2, y2);

return 0;

}



**Experiment -4:** **Scan Conversion of a circle using Bressenham’s algorithm.**

**Code:**

#include <iostream>

#include <cmath>

using namespace std;

void scanConvertCircleBresenham(int centerX, int centerY, int radius)

{

int x = 0;

int y = radius;

int d = 3 - 2 \* radius;

const int width = 80;

const int height = 24;

char grid[height][width];

for (int i = 0; i < height; ++i)

{

for (int j = 0; j < width; ++j)

{

grid[i][j] = ' ';

}

}

while (x <= y)

{

if (centerX + x >= 0 && centerX + x < width && centerY + y >= 0 && centerY + y < height)

grid[centerY + y][centerX + x] = '\*';

if (centerX - x >= 0 && centerX - x < width && centerY + y >= 0 && centerY + y < height)

grid[centerY + y][centerX - x] = '\*';

if (centerX + x >= 0 && centerX + x < width && centerY - y >= 0 && centerY - y < height)

grid[centerY - y][centerX + x] = '\*';

if (centerX - x >= 0 && centerX - x < width && centerY - y >= 0 && centerY - y < height)

grid[centerY - y][centerX - x] = '\*';

if (centerY + x >= 0 && centerY + x < height && centerX + y >= 0 && centerX + y < width)

grid[centerY + x][centerX + y] = '\*';

if (centerY - x >= 0 && centerY - x < height && centerX + y >= 0 && centerX + y < width)

grid[centerY - x][centerX + y] = '\*';

if (centerY + x >= 0 && centerY + x < height && centerX - y >= 0 && centerX - y < width)

grid[centerY + x][centerX - y] = '\*';

if (centerY - x >= 0 && centerY - x < height && centerX - y >= 0 && centerX - y < width)

grid[centerY - x][centerX - y] = '\*';

if (d < 0)

{

d = d + 4 \* x + 6;

}

else

{

d = d + 4 \* (x - y) + 10;

y--;

}

x++;

}

for (int i = 0; i < height; ++i)

{

for (int j = 0; j < width; ++j)

{

cout << grid[i][j];

}

cout <<endl;

}

}

int main()

{

int centerX = 40;

int centerY = 12;

int radius = 10;

scanConvertCircleBresenham(centerX, centerY, radius);

return 0;

}

