Experiment 4

Student Name: Gaganjot Singh **UID: 22BCS14843**
Branch: B.E. CSE III Yr **Section:** 22BCS-IOT-612-B
Semester: 6th
Subject Name: Computer Graphics with Lab **Subject Code:** 22CSH-352

1. Aim:

a) Develop a program to draw a circle using the circle generator algorithm for a given center and radius.

b) Develop a program to draw a circle using the midpoint circle algorithm for a given center and radius.

2. Objective:

To develop and implement the circle generator and midpoint circle generator algorithm to draw a circle with a given center and radius.

3. Code:

```
#include <iostream.h>

#include <graphics.h>

#include <conio.h>

#include <math.h>


void plotPoints(int xc, int yc, int x, int y)

{   putpixel(xc + x, yc + y, WHITE);

    putpixel(xc - x, yc + y, WHITE);

    putpixel(xc + x, yc - y, WHITE);

    putpixel(xc - x, yc - y, WHITE);

    putpixel(xc + y, yc + x, WHITE);

    putpixel(xc - y, yc + x, WHITE);

    putpixel(xc + y, yc - x, WHITE);

    putpixel(xc - y, yc - x, WHITE);}


void MidPoint(int xc, int yc, int rad)

{  int x = 0;
```

```
    int y = rad;
    int p = 1 - rad;

    plotPoints(xc, yc, x, y);

    while (x < y)
    {   x++;
        if (p < 0){
            p = p + 2 * x + 1;}
        else
        {   y--;
            p = p + 2 * (x - y) + 1;
        }
     plotPoints(xc, yc, x, y);}}

void CircleGenerator(int xc, int yc, int rad)
{   int x = 0;
    int y = rad;
    int d = 3 - 2 * rad;

    plotPoints(xc, yc, x, y);

    while (x <= y)
    {  if (d <= 0)
        {d = d + 4 * x + 6;}
       else
        {d = d + 4 * (x - y) + 10;
          y--;}
       x++;
        plotPoints(xc, yc, x, y);}}

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

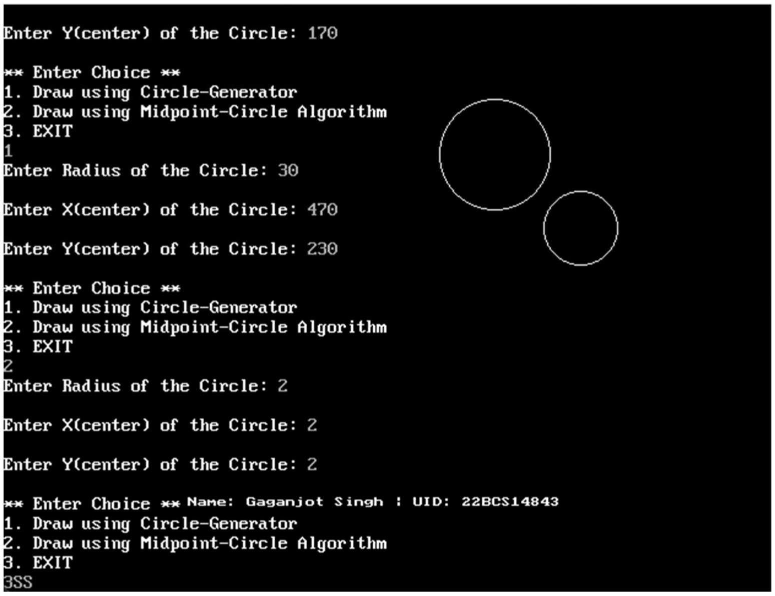
```
    initgraph(&gd, &gm, "C:\\Turboc3\\BGI");
    outtextxy(150, 450, "Name: Gaganjot Singh | UID: 22BCS14843");


    int choice, rad, xc, yc;
    do{
    cout << "Enter Radius of the Circle: ";    cin >> rad;
    cout << "\nEnter X(center) of the Circle: ";    cin >> xc;
    cout << "\nEnter Y(center) of the Circle: ";    cin >> yc;

    cout << "\n** Enter Choice **\n1. Draw using Circle-Generator\n2. Draw using
Midpoint-Circle Algorithm\n3. EXIT\n";
    cin >> choice;


      switch (choice)
        { case 1: CircleGenerator(xc, yc, rad); break;
          case 2: MidPoint(xc, yc, rad); break;
          case 3: cout<< "Exiting Program!! Bye"; break;
         default: cout << "\nEnter ONLY one of the choices"; break;}
    } while (choice != 3);
    getch();
    closegraph();
    return 0;}
```

4. Output:



```
Enter Y(center) of the Circle: 170

** Enter Choice **
1. Draw using Circle-Generator
2. Draw using Midpoint-Circle Algorithm
3. EXIT
1
Enter Radius of the Circle: 30

Enter X(center) of the Circle: 470

Enter Y(center) of the Circle: 230

** Enter Choice **
1. Draw using Circle-Generator
2. Draw using Midpoint-Circle Algorithm
3. EXIT
2
Enter Radius of the Circle: 2

Enter X(center) of the Circle: 2

Enter Y(center) of the Circle: 2

** Enter Choice ** Name: Gaganjot Singh : UID: 22BCS14843
1. Draw using Circle-Generator
2. Draw using Midpoint-Circle Algorithm
3. EXIT
3SS
```

5. Learning Outcome:

i. Gain hands-on experience with two different methods: the Circle Generator Algorithm (direct computation) and the Midpoint Circle Algorithm.

ii. Learnt how to break down a complex task (like circle drawing) into smaller functions, making the code easier to manage and understand plotPoints(), MidPoint(), CircleGenerator()

iii. Applied `switch` statements for control flow based on user choices.

iv. Applied loops (e.g., `do-while`) for repeated operations until a certain condition is met.