**Experiment No** : 6

**Aim** : Using Mid-Point Circle algorithm, draw any 5 animated facial emoji

**Theory** :
Since the circle is a frequently used component in pictures and graphs, a procedure for generating either full circles or circular arcs is included in many graphics packages. Hence various algorithms for drawing rasterized circles have been formulated. One such algorithm is the Mid-Point Circle algorithm. This algorithm takes advantage of the symmetry of the circle and only computes points for 1 quadrant of the circle.

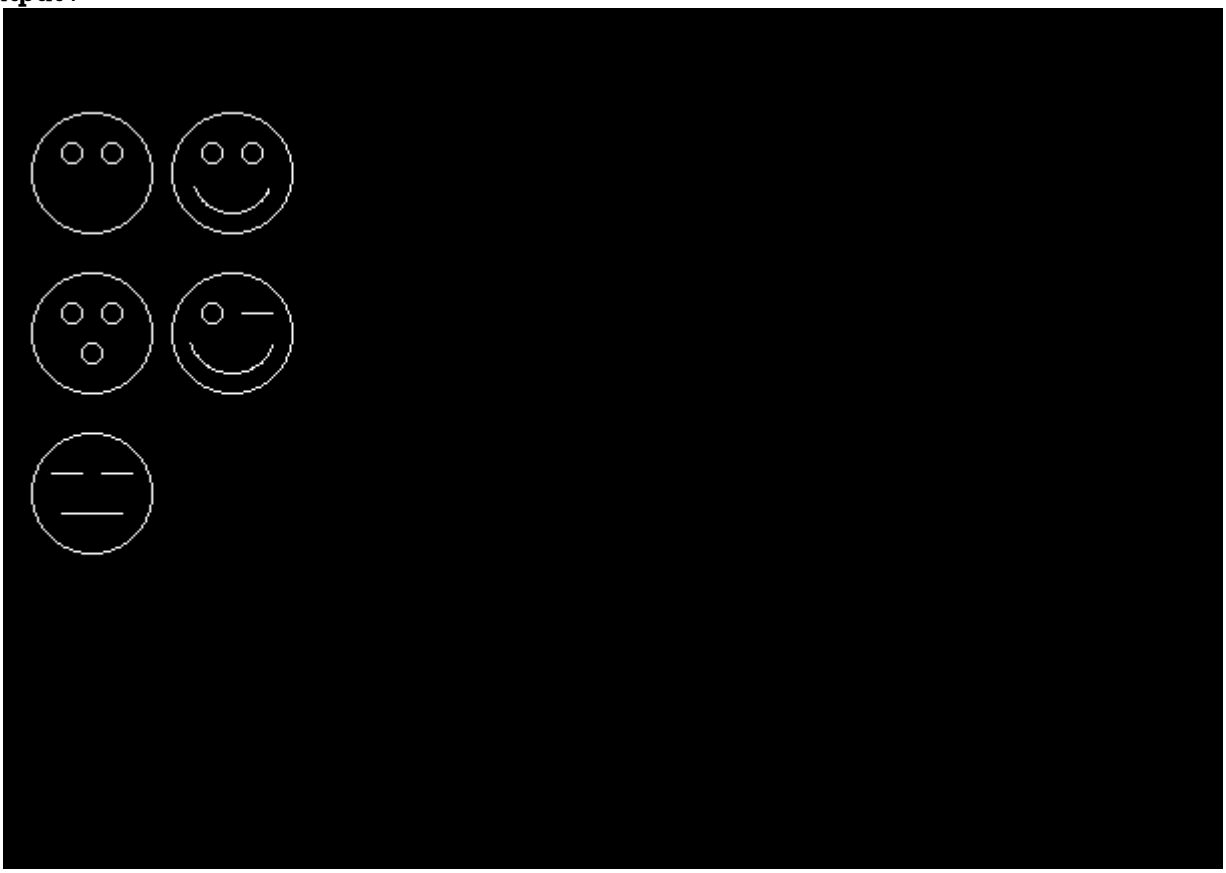
**Algorithm**
1. Input radius r and circle center (xc, yc), then set the coordinates for the first point on the circumference of a circle centered on the origin as (x0, y0) = (0, r)
2. Calculate the initial value of the decision parameter as p0 = 5/4 − r
3. At each xk position, starting at k = 0, perform the following test: If pk < 0, the next point along the circle centered on (0, 0) is (xk + 1, yk) and pk+1 = pk + 2xk+1 + 1
Otherwise, the next point along the circle is (xk + 1, yk − 1) and pk + 1 = pk + 2xk + 1 + 1 − 2yk + 1 where 2xk+1 = 2xk + 2 and 2yk+1 = 2yk − 2
4. Determine symmetry points in the other seven octants.
5. Move each calculated pixel position (x, y) onto the circular path

**Code & Output** :
```
#include<graphics.h>
#include<iostream>
using namespace std;
void mid_point_circle(int xc, int yc, int r);
void find_for_all_octants_shift(int x, int y, int xc, int yc);
void emoji_1(int i);
void emoji_2(int i);
void emoji_3(int i);
void emoji_4(int i);
void emoji_5(int i);

int main()
{
  int gd=DETECT,gm;
  initgraph(&gd,&gm,NULL);
  for(int i=0;i<=100;i++){
   cleardevice();
   emoji_1(i);
   emoji_2(i);
   emoji_3(i);
   emoji_4(i);
   emoji_5(i);
   delay(100);
  }
  closegraph();
  return 0;
}

void mid_point_circle(int xc, int yc, int r)
{
   int x=0, y=r;
   float pk = (5/4) - r;

   while(true)
```

```
    {
        find_for_all_octants_shift(x,y,xc,yc);
        cout<<endl;
        if(x >= y)
            break;
        if(pk < 0)
        {
            x = x + 1;
            y = y;
            pk = pk + (2*x) + 1;
        }
        else
        {
            x = x + 1;
            y = y - 1;
            pk = pk + (2*x) + 1 - (2*y);
        }
    }
}

void find_for_all_octants_shift(int x, int y, int xc, int yc)
{
    int x1,x2,x3,x4,x5,x6,x7,x8;
    int y1,y2,y3,y4,y5,y6,y7,y8;

    x1 = x + xc; y1 = y + yc;
    putpixel(x1,y1,WHITE);

    x2 = x + xc; y2 = (-1*y) + yc;
    putpixel(x2,y2,WHITE);

    x3 = y + xc; y3 = (-1*x) + yc;
    putpixel(x3,y3,WHITE);

    x4 = (-1*y) + xc; y4 = (-1*x) + yc;
    putpixel(x4,y4,WHITE);

    x5 = (-1*x) + xc; y5 = (-1*y) + yc;
    putpixel(x5,y5,WHITE);

    x6 = (-1*x) + xc; y6 = y + yc;
    putpixel(x6,y6,WHITE);

    x7 = (-1*y) + xc; y7 = x + yc;
    putpixel(x7,y7,WHITE);

    x8 = y + xc; y8 = x + yc;
    putpixel(x8,y8,WHITE);

}
void emoji_1(int i) // : )
{
    mid_point_circle(50+i,100,30);
    mid_point_circle(40+i,90,5);
    mid_point_circle(60+i,90,5);
    setcolor(BLACK);
    arc(50+i,100,180,360,20);
}

void emoji_2(int i) // :|
```

```
{
   mid_point_circle(50+i,180,30);
   mid_point_circle(40+i,170,5);
   mid_point_circle(60+i,170,5);
   mid_point_circle(50+i,190,5);
   setcolor(WHITE);

}

void emoji_3(int i) // -_-
{
   mid_point_circle(50+i,260,30);
   setcolor(WHITE);
   line(30+i,250,45+i,250);
   line(55+i,250,70+i,250);
   line(35+i,270,65+i,270);
}

void emoji_4(int i) // :')
{
   mid_point_circle(120+i,100,30);
   mid_point_circle(110+i,90,5);
   mid_point_circle(130+i,90,5);
   setcolor(WHITE);
   arc(120+i,100,20,160,20);
}

void emoji_5(int i) // B)
{
   mid_point_circle(120+i,180,30);
   mid_point_circle(110+i,170,5);
   setcolor(WHITE);
   line(125+i,170,140+i,170);
   arc(120+i,178,20,160,22);
}
```

**Output** :



**Conclusion** : Program to draw 5 emojis using mid point circle algorithm was successfully written and executed

**Deepraj Bhosale**
**181105016**