CSE PROJECT - 2nd semester

Course Code : CSE 4271

Submitted by :

Name : Maimuna Biswas Noshin

ID : 200021347

Department : EEE

Section : C

PROJECT NAME : Solar System With Graphics in C Language

---------||C PROGRAM CODE||------------------

(As the program is done by using graphics, compiler should be turned into graphics mode)

// to run this program complier settings should be changed and turn into graphics mode//

```
// C program to draw solar system using
// computer graphics
#include <conio.h>
#include <dos.h>
#include <graphics.h>
#include<iostream>
#include <stdlib.h>
#include <GL/glut.h>
#include <math.h>
#include <stdio.h>
```

```
    // Function to manipulates the position

    // of planets on the orbit

    GLuint textureSun, textureEarth, textureMercury, textureVenus, textureMars, textureJupiter,textureSaturn, textureSaturn, textureUranus, textureNeptune, texturePluto;;

    GLUqyadric *sun, *earth, *mercury, *venus, *mars, *jupiter, *saturn, *uranus, *neptune, *pluto ;

    void planetMotion(int xrad, int yrad,

                      int midx, int midy,

                      int x[70], int y[70])
```

```
{
        int i, j = 0;


        // Positions of planets in their
        // corresponding orbits
        for (i = 360; i > 0; i = i - 6) {
                x[j] = midx - (xrad * cos((i * 3.14) / 180));
                y[j++] = midy - (yrad * sin((i * 3.14) / 180));
        }


        return;
}
```

```
// Driver Code

int main()

{


        // Initialize graphic driver

        int gdriver = DETECT, gmode, err;

        int i = 0, midx, midy;

        int xrad[9], yrad[9], x[9][70], y[9][70];

        int pos[9], planet[9];


        // Initialize graphics mode by

        // passing the three arguments

        // to initgraph()
// &gdriver is the address of gdriver
// variable, &gmode is the address of
// gmode and "C:\\Turboc3\\BGI" is the
// directory path where BGI files
// are stored
initgraph(&gdriver, &gmode, "");
err = graphresult();
```

```
if (err != grOk) {


        // Error occurred

        printf("Graphics Error: %s",

                grapherrormsg(err));

        return 0;

}


// Mid positions at x and y-axis

midx = getmaxx() - 220;

midy = getmaxy() - 150;


// Manipulating radius of all

// the nine planets

planet[0] = 8;

for (i = 1; i < 9; i++) {

        planet[i] = planet[i - 1] + 1;

}
```

```
// Offset position for the planets
// on their corresponding orbit
for (i = 0; i < 9; i++) {
        pos[i] = i * 6;
}


// Orbits for all 9 planets
xrad[0] = 70, yrad[0] = 40;
for (i = 1; i < 9; i++) {
        xrad[i] = xrad[i - 1] + 38;
        yrad[i] = yrad[i - 1] + 20;
}
```

```
// Positions of planets on their

// corresponding orbits

for (i = 0; i < 9; i++) {

        planetMotion(xrad[i], yrad[i],

                                        midx, midy, x[i],

                                        y[i]);

}
```

```
while (!kbhit()) {


        // Drawing 9 orbits

        setcolor(WHITE);

        for (i = 0; i < 9; i++) {

                setcolor(CYAN);

                ellipse(midx, midy, 0, 360,

                                xrad[i], yrad[i]);

        }


        // Sun at the mid of solar system

        outtextxy(midx, midy, " SUN");

        setcolor(YELLOW);
```

```
setfillstyle(SOLID_FILL, YELLOW);
circle(midx, midy, 30);
floodfill(midx, midy, YELLOW);

// Mercury in first orbit
setcolor(CYAN);
```

```
setfillstyle(SOLID_FILL, CYAN);
outtextxy(x[0][pos[0]],
          y[0][pos[0]],
          " MERCURY");

pieslice(x[0][pos[0]],
         y[0][pos[0]],
         0, 360, planet[0]);

// Venus in second orbit
setcolor(GREEN);
```

```
        setfillstyle(SOLID_FILL, GREEN);

                outtextxy(x[1][pos[1]],

                        y[1][pos[1]],

                        " VENUS");

                pieslice(x[1][pos[1]],

                        y[1][pos[1]],

                        0, 360, planet[1]);



                // Earth in third orbit

                setcolor(BLUE);

                setfillstyle(SOLID_FILL, BLUE);

                outtextxy(x[2][pos[2]],

                        y[2][pos[2]],

                        " EARTH");

                pieslice(x[2][pos[2]],

                        y[2][pos[2]],

                        0, 360, planet[2]);
```

```
// Mars in fourth orbit

setcolor(RED);

setfillstyle(SOLID_FILL, RED);

outtextxy(x[3][pos[3]],

                    y[3][pos[3]],

                    " MARS");

pieslice(x[3][pos[3]],

                    y[3][pos[3]],

                    0, 360, planet[3]);
```

```
// Jupiter in fifth orbit
setcolor(BROWN);
setfillstyle(SOLID_FILL, BROWN);
outtextxy(x[4][pos[4]],
          y[4][pos[4]],
          " JUPITER");
pieslice(x[4][pos[4]],
         y[4][pos[4]],
         0, 360, planet[4]);
```

```
// Saturn in sixth orbit

setcolor(LIGHTGRAY);

setfillstyle(SOLID_FILL, LIGHTGRAY);

outtextxy(x[5][pos[5]],

                y[5][pos[5]],

                " SATURN");

pieslice(x[5][pos[5]],

                y[5][pos[5]],

                0, 360, planet[5]);
```

```
// Uranus in seventh orbit

setcolor(LIGHTGREEN);

setfillstyle(SOLID_FILL, LIGHTGREEN);

        outtextxy (x [6] [pos [6]],

                y [6] [pos [6]],

                " URANUS");

        pieslice (x [6] [pos [6]],

                y [6] [pos [6]],

                0, 360, planet [6]);
```

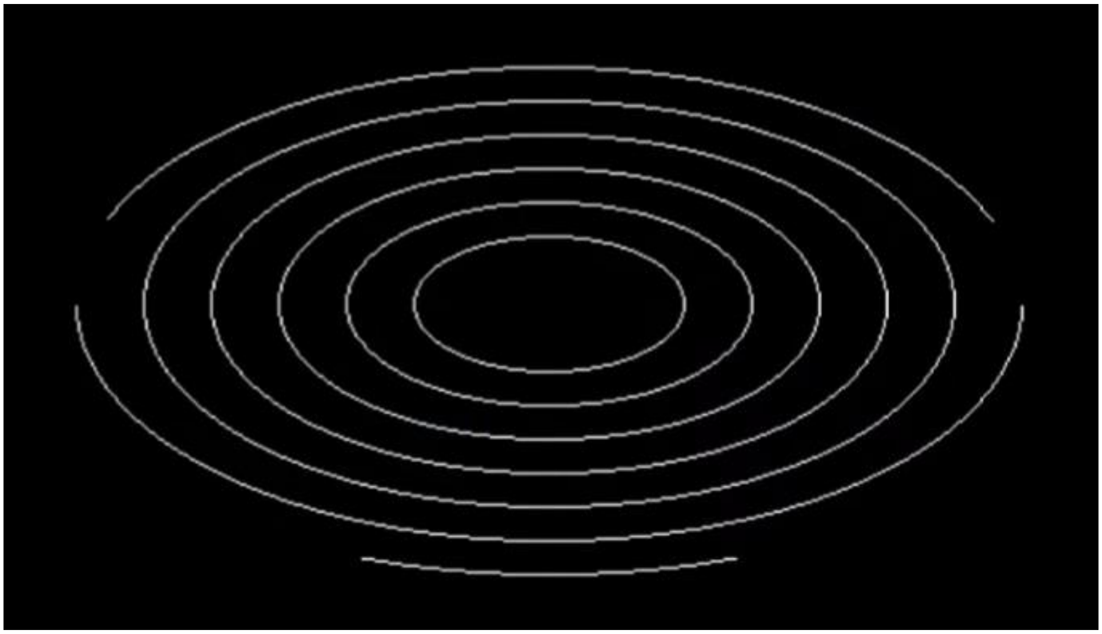
```
// Neptune in eighth orbit

setcolor (LIGHTBLUE);

setfillstyle (SOLID_FILL, LIGHTBLUE);

outtextxy (x [7] [pos [7]],

                    y [7] [pos [7]],

                    " NEPTUNE");

pieslice (x [7] [pos [7]],

                    y [7] [pos [7]],

                    0, 360, planet [7]);


// Pluto in ninth orbit

setcolor (LIGHTRED);

setfillstyle (SOLID_FILL, LIGHTRED);
```

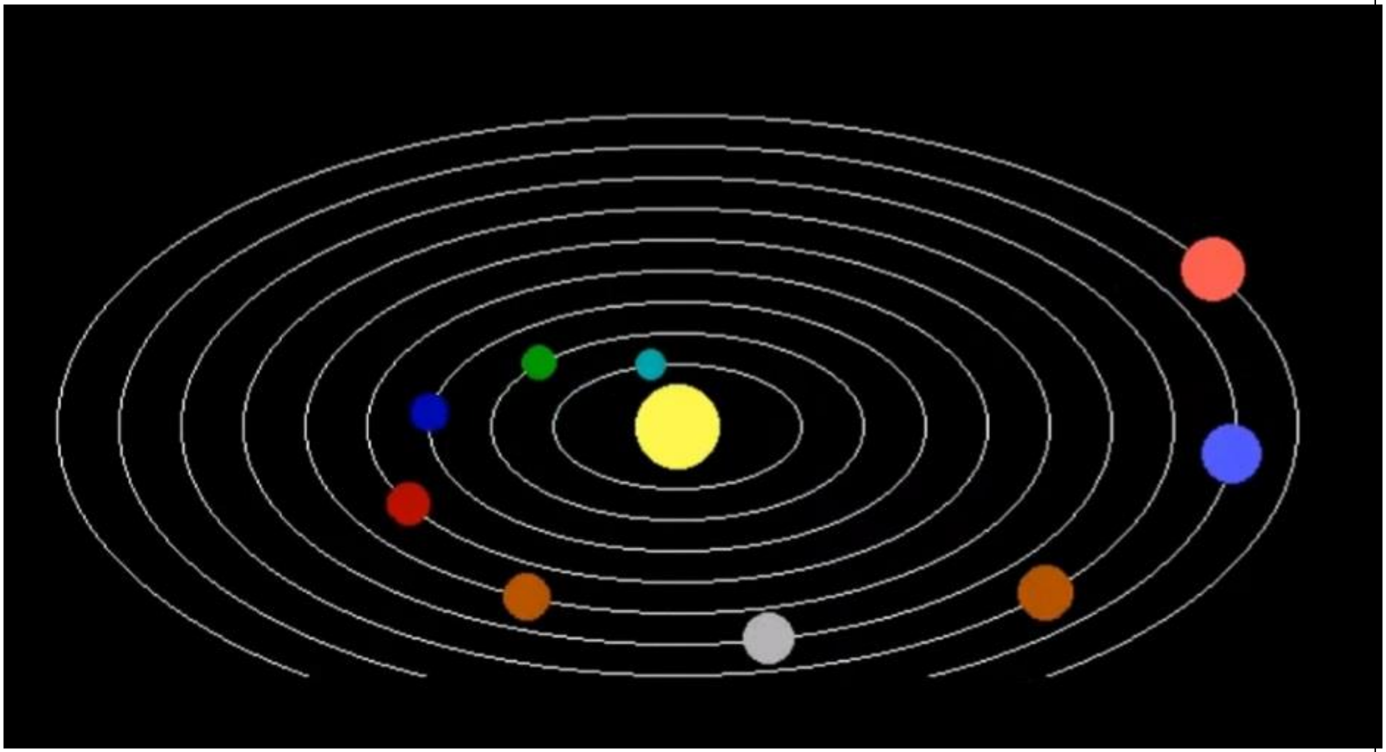
```
    outtextxy (x [8] [pos [8]],
                        y [8] [pos [8]],
                        " PLUTO");
            pieslice (x [8] [pos [8]],
                        y [8] [pos [8]],
                        0, 360, planet [8]);


            // Checking for one complete
            // rotation
            for (i = 0; i < 9; i++) {
                  if (pos[i] <= 0) {
                        pos[i] = 59;
                  }
                  else {
                        pos[i] = pos[i] - 1;
                  }
            }
```

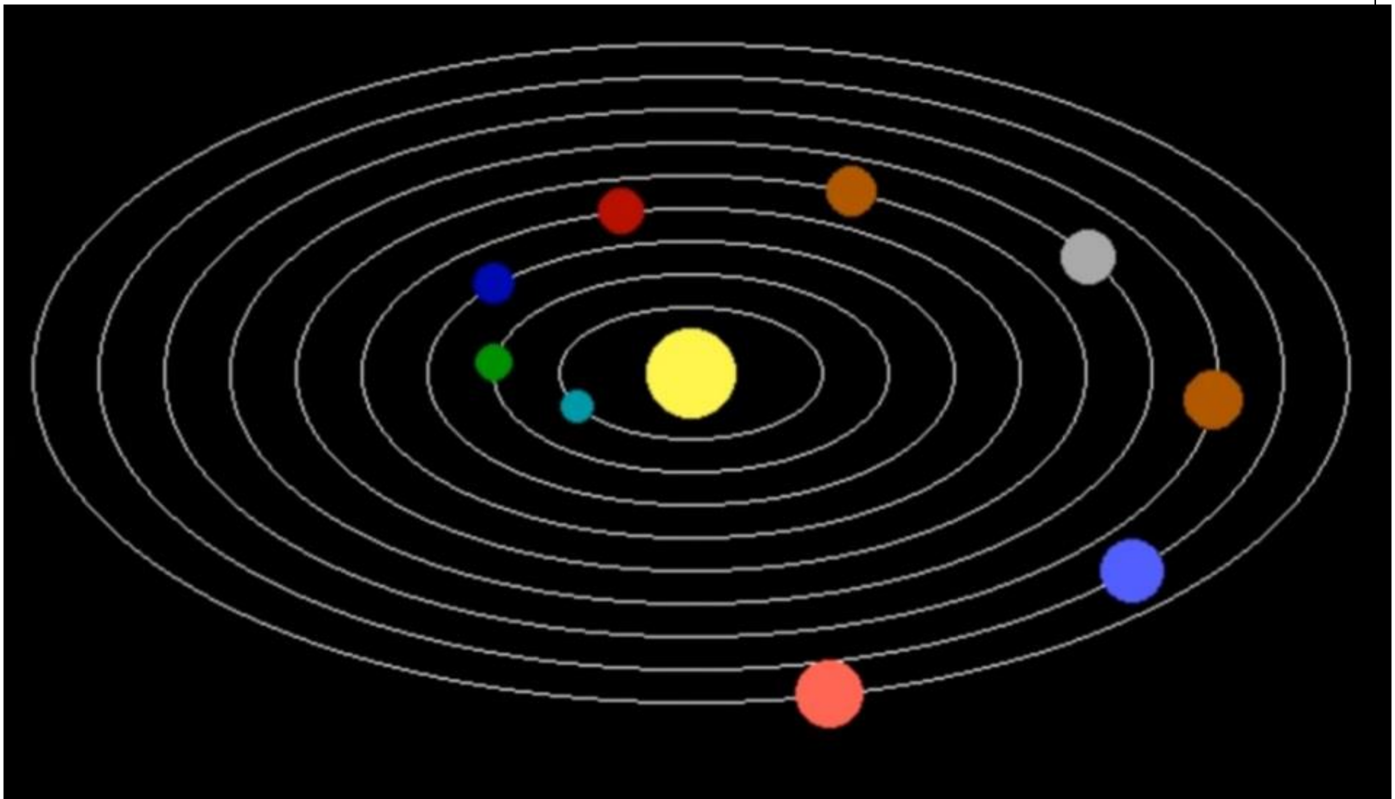
```
        // Sleep for 100 milliseconds
        delay (100);

        // Clears graphic screen
        cleardevice ();
    }

    // Deallocate memory allocated
    // for graphic screen
    closegraph();

    return 0;
}
```

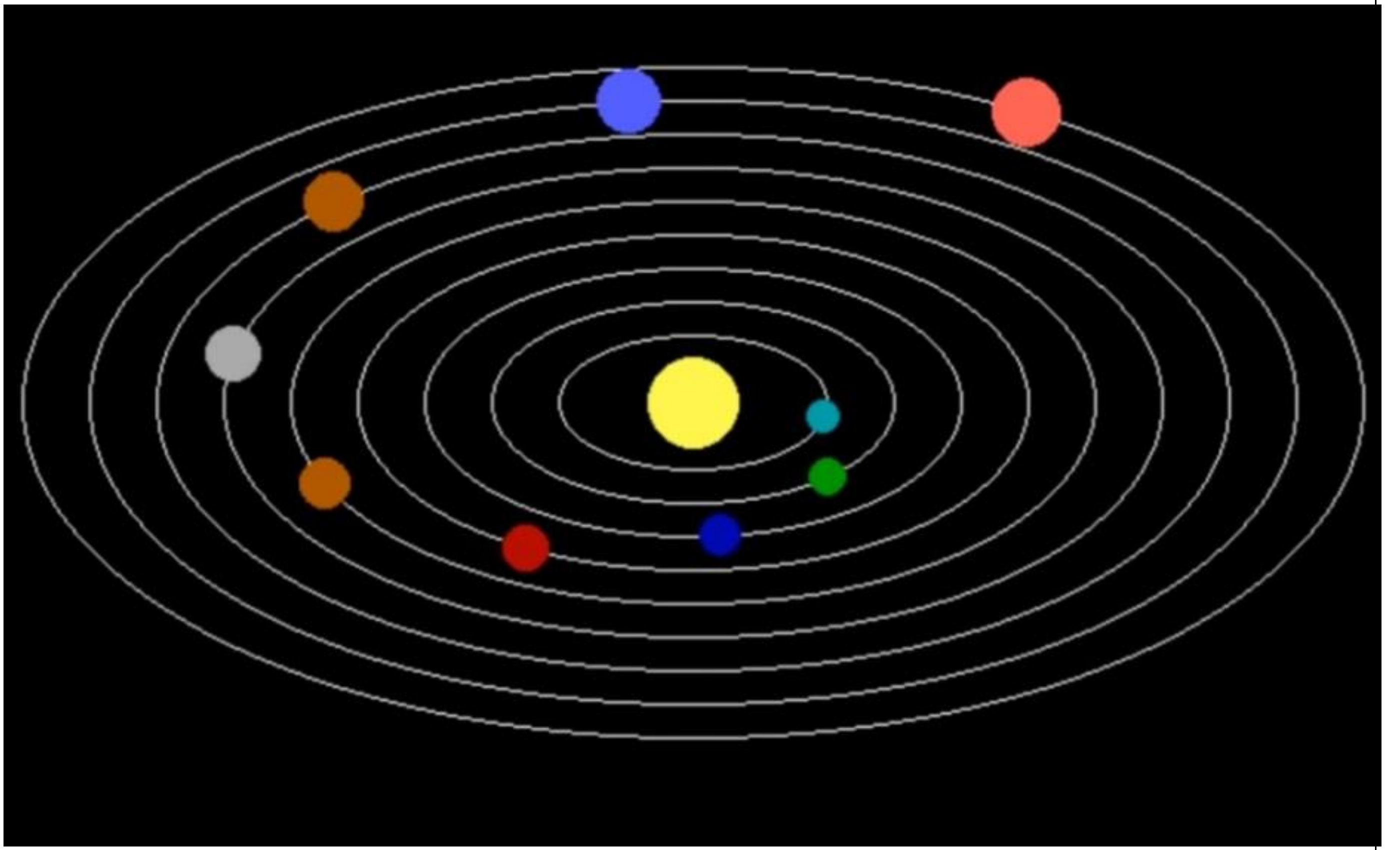
OUTPUT:

After running this program using TDM GCC 32 compiler, a dynamic solar system model has obtained.









DEMO LINK:

https://youtu.be/lVa12mVV9rM

FEATURES:

1. Making elliptical arc:

The elliptical arcs are the locus of the planets. To define the elliptical pathway of every planet four integers have been used.

To define the center of the all elliptical arcs midx and midy have been used. This is the co-ordinate of the sun. Then the distance of every planet from the sun has been defined by xrad and yrad.

```
while (!kbhit()) {

        // Drawing 9 orbits
        setcolor(WHITE);
        for (i = 0; i < 9; i++) {
                setcolor(CYAN);
                ellipse(midx, midy, 0, 360,
                                xrad[i], yrad[i]);
        }
```

This part has been used for showing the elliptical path of the planets.

Function initgraph() is used to initialize graphics.

2. Specifying the planets:

To give all the planets their indigenous features some functions have been used. Function setcolour() has used to color the planets.
Function setfillstyle() has used to define how the planets will be filled.
Function pieslice() has been used to color a portion of the planets.

3. Motion of the planets:

```
for (i = 0; i < 9; i++) {
```

```
if (pos[i] <= 0) {
        pos[i] = 59;
}
else {
        pos[i] = pos[i] - 1;
}
```

This part has been used to move the planets to give the solar system a dynamic motion. i<9 to check that all the nine planets are moving. Pos[i] has been used for one complete rotation.