Graphics

Lab 3

Name / Ahmed Abdallah Aboeleid

ID / 19015274

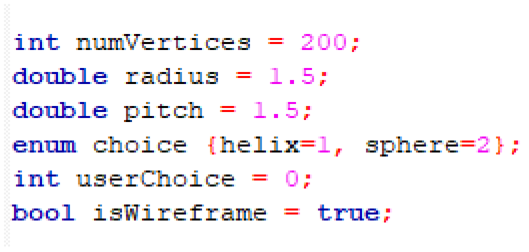
Problem Statement:

You are required to create an OpenGL project using the project template. You should implement an application that

- asks user to choose between two types of shapes (Helix and Sphere).
- At runtime, Input handling should be as follows:
  • In case of sphere:
  o When user presses Q/q, increase/decrease number of latitudinal slices.
  o When user presses P/p, increase/decrease number of longitudinal slices.
  o When user presses W/w, draw sphere in wireframe / draw filled sphere.
  • In case of helix:
  o When user presses R/r, increase/decrease radius of the helix.
  o When user presses H/h, increase/decrease pitch of helix.
  o When user presses N/n, increase/decrease number of vertices used to draw the helix.
- Number of turns of helix that should be drawn is 5 turns. Use GL_FRONT_AND_BACK option for drawing mode. Also, For each vertex drawn you, should pick random color to draw vertex with.

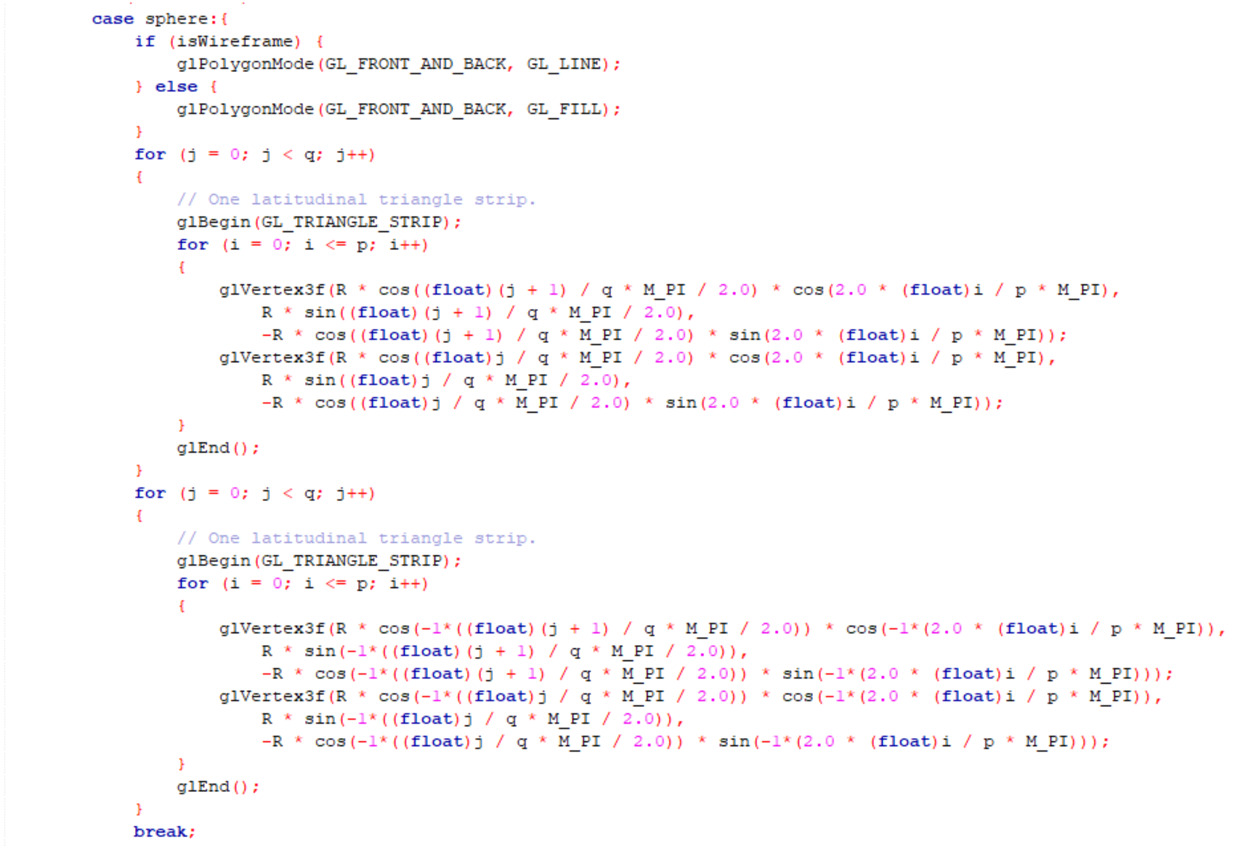
Code:

⇨ Global variable



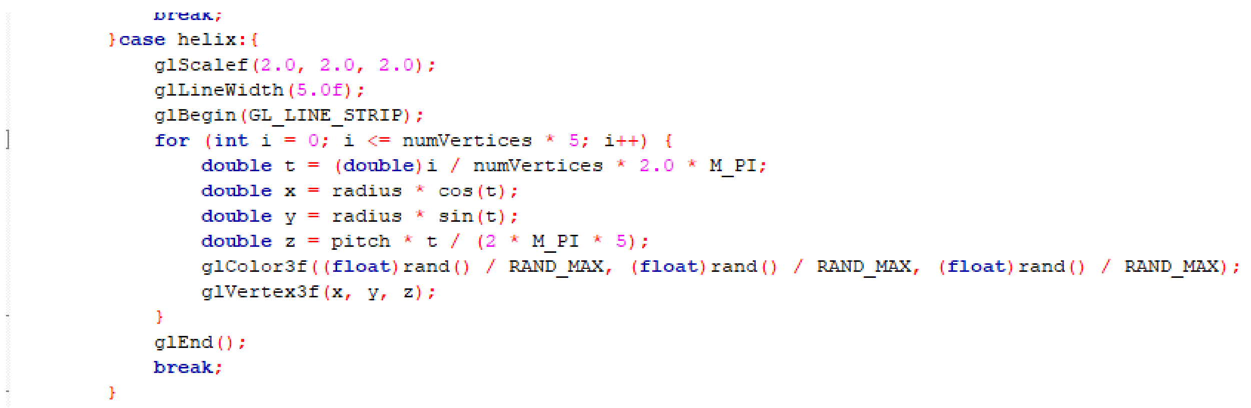
```
int numVertices = 200;
double radius = 1.5;
double pitch = 1.5;
enum choice {helix=1, sphere=2};
int userChoice = 0;
bool isWireframe = true;
```

⇨ draw sphere:



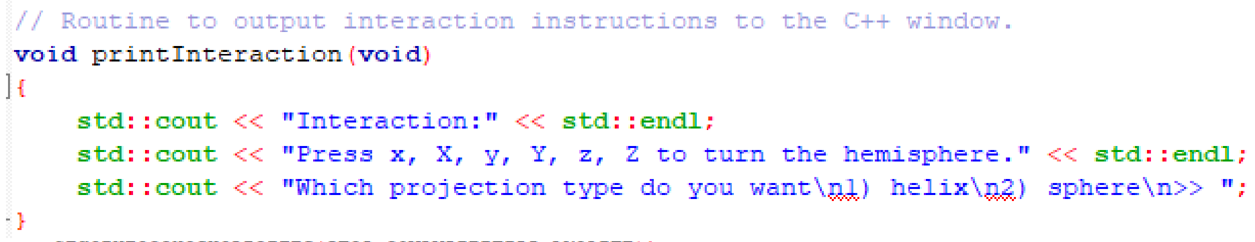
```
        case sphere:{
            if (isWireframe) {
                glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);
            } else {
                glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
            }
            for (j = 0; j < q; j++)
            {
                // One latitudinal triangle strip.
                glBegin(GL_TRIANGLE_STRIP);
                for (i = 0; i <= p; i++)
                {
                    glVertex3f(R * cos((float)(j + 1) / q * M_PI / 2.0) * cos(2.0 * (float)i / p * M_PI),
                        R * sin((float)(j + 1) / q * M_PI / 2.0),
                        -R * cos((float)(j + 1) / q * M_PI / 2.0) * sin(2.0 * (float)i / p * M_PI));
                    glVertex3f(R * cos((float)j / q * M_PI / 2.0) * cos(2.0 * (float)i / p * M_PI),
                        R * sin((float)j / q * M_PI / 2.0),
                        -R * cos((float)j / q * M_PI / 2.0) * sin(2.0 * (float)i / p * M_PI));
                }
                glEnd();
            }
            for (j = 0; j < q; j++)
            {
                // One latitudinal triangle strip.
                glBegin(GL_TRIANGLE_STRIP);
                for (i = 0; i <= p; i++)
                {
                    glVertex3f(R * cos(-1*((float)(j + 1) / q * M_PI / 2.0)) * cos(-1*(2.0 * (float)i / p * M_PI)),
                        R * sin(-1*((float)(j + 1) / q * M_PI / 2.0)),
                        -R * cos(-1*((float)(j + 1) / q * M_PI / 2.0)) * sin(-1*(2.0 * (float)i / p * M_PI)));
                    glVertex3f(R * cos(-1*((float)j / q * M_PI / 2.0)) * cos(-1*(2.0 * (float)i / p * M_PI)),
                        R * sin(-1*((float)j / q * M_PI / 2.0)),
                        -R * cos(-1*((float)j / q * M_PI / 2.0)) * sin(-1*(2.0 * (float)i / p * M_PI)));
                }
                glEnd();
            }
            break;
```

⇨ draw helix:

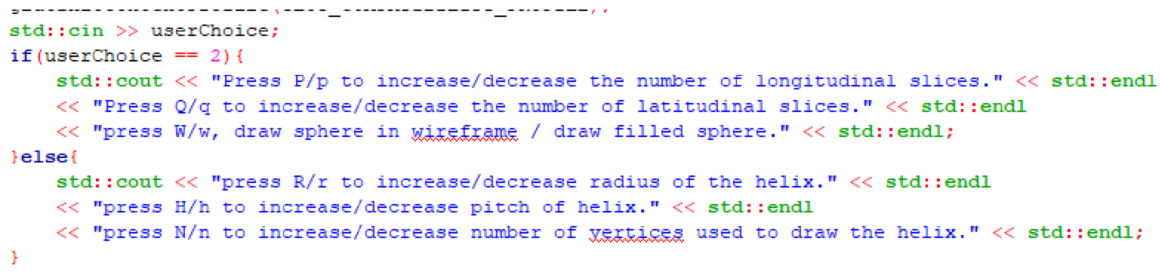


```
            break;
        }case helix:{
            glScalef(2.0, 2.0, 2.0);
            glLineWidth(5.0f);
            glBegin(GL_LINE_STRIP);
            for (int i = 0; i <= numVertices * 5; i++) {
                double t = (double)i / numVertices * 2.0 * M_PI;
                double x = radius * cos(t);
                double y = radius * sin(t);
                double z = pitch * t / (2 * M_PI * 5);
                glColor3f((float)rand() / RAND_MAX, (float)rand() / RAND_MAX, (float)rand() / RAND_MAX);
                glVertex3f(x, y, z);
            }
            glEnd();
            break;
        }
```

⇨ print statement:



```
// Routine to output interaction instructions to the C++ window.
void printInteraction(void)
{
    std::cout << "Interaction:" << std::endl;
    std::cout << "Press x, X, y, Y, z, Z to turn the hemisphere." << std::endl;
    std::cout << "Which projection type do you want\n1) helix\n2) sphere\n>> ";
}
```



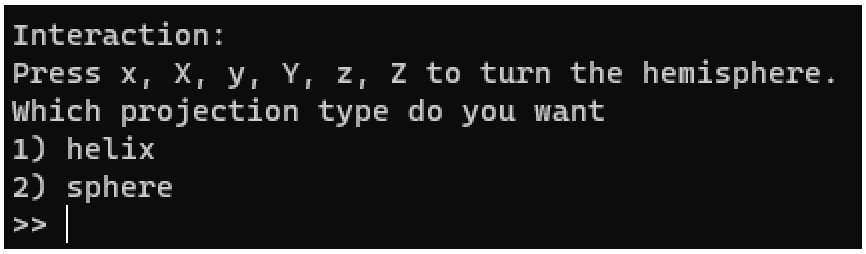
```
    std::cin >> userChoice;
    if(userChoice == 2){
        std::cout << "Press P/p to increase/decrease the number of longitudinal slices." << std::endl
        << "Press Q/q to increase/decrease the number of latitudinal slices." << std::endl
        << "press W/w, draw sphere in wireframe / draw filled sphere." << std::endl;
    }else{
        std::cout << "press R/r to increase/decrease radius of the helix." << std::endl
        << "press H/h to increase/decrease pitch of helix." << std::endl
        << "press N/n to increase/decrease number of vertices used to draw the helix." << std::endl;
    }
```

⇨ **handle input user and pressing:**

```
case 'r':
    radius += 0.1;
    glutPostRedisplay();
    break;
case 'R':
    radius -= 0.1;
    glutPostRedisplay();
    break;
case 'N':
    numVertices += 10;
    glutPostRedisplay();
    break;
case 'n':
    numVertices -= 10;
    glutPostRedisplay();
    break;
case 'H':
    pitch += 1.0;
    glutPostRedisplay();
    break;
case 'h':
    pitch -= 1.0;
    glutPostRedisplay();
    break;
case 'W':
    isWireframe = true;
    glutPostRedisplay();
    break;
case 'w':
    isWireframe = false;
    glutPostRedisplay();
    break;
case 'P':
    p += 1;
    glutPostRedisplay();
    break;
case 'p':
    if (p > 3) p -= 1;
    glutPostRedisplay();
    break;
```

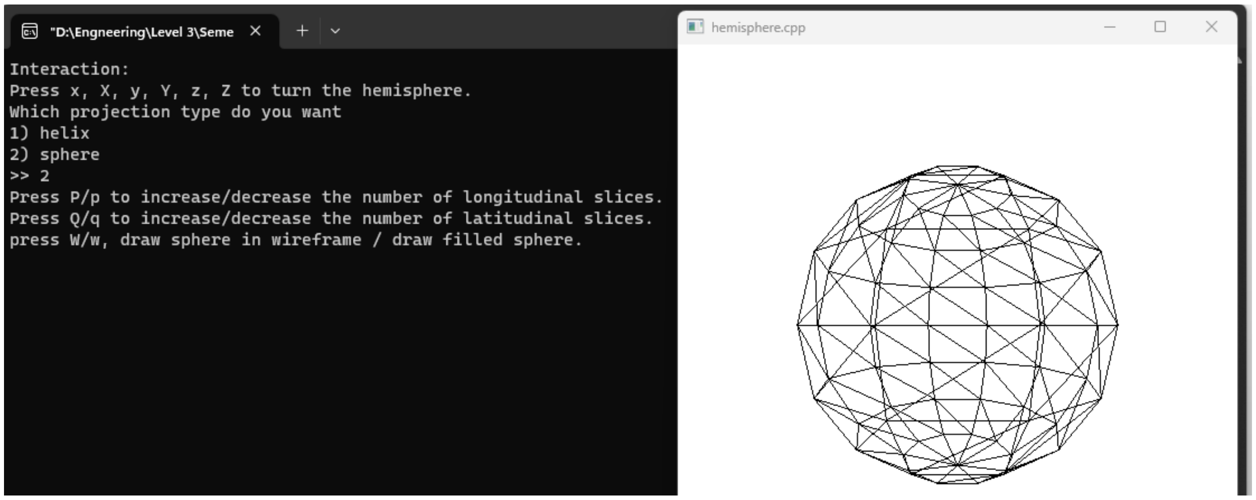
Screenshoots:

⇨ start screen



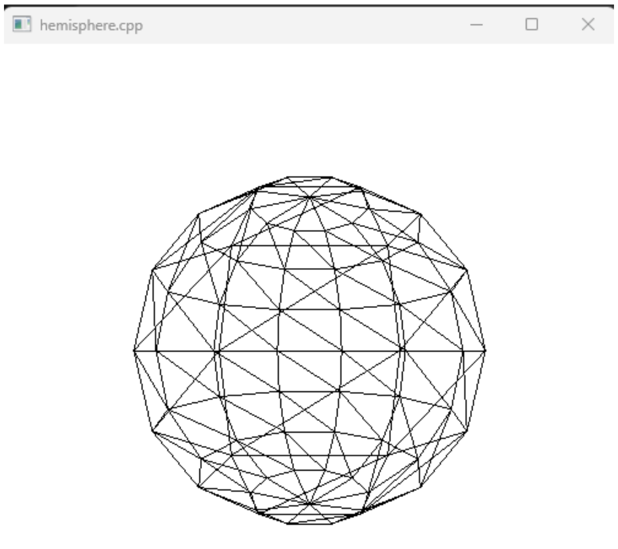
```
Interaction:
Press x, X, y, Y, z, Z to turn the hemisphere.
Which projection type do you want
1) helix
2) sphere
>>
```

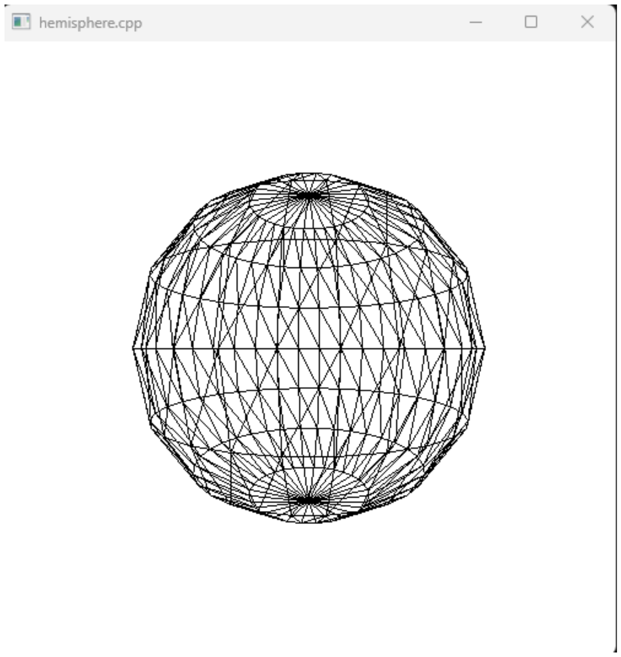
⇨ draw sphere

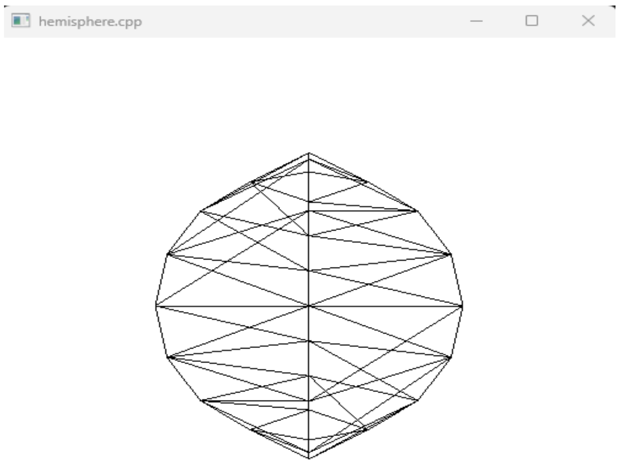


```
Interaction:
Press x, X, y, Y, z, Z to turn the hemisphere.
Which projection type do you want
1) helix
2) sphere
>> 2
Press P/p to increase/decrease the number of longitudinal slices.
Press Q/q to increase/decrease the number of latitudinal slices.
press W/w, draw sphere in wireframe / draw filled sphere.
```

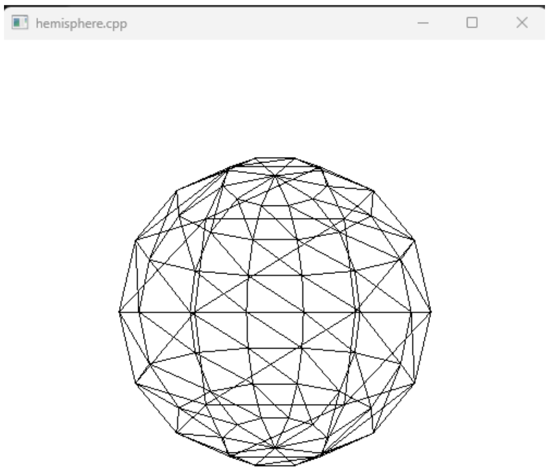
⇨ Press P/p to increase/decrease the number of longitudinal slices.

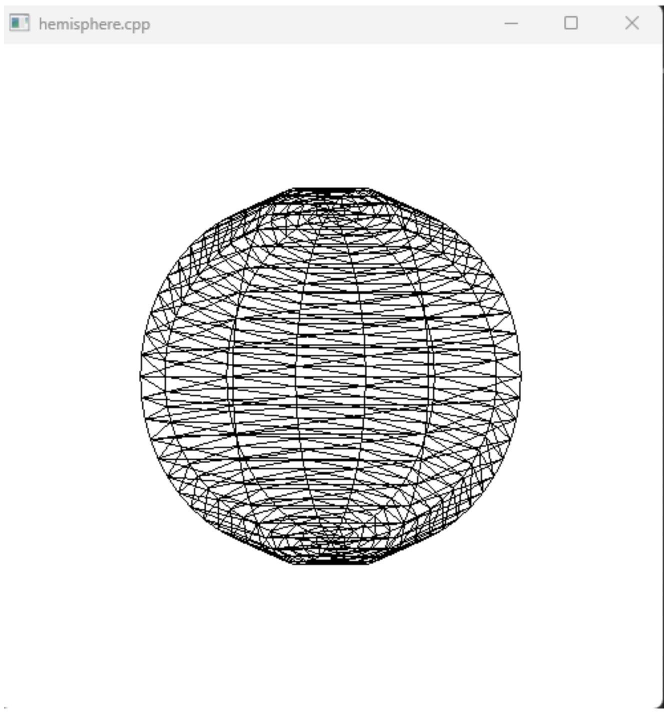


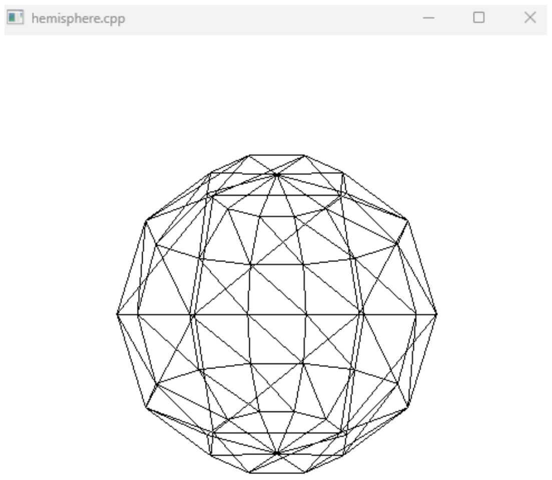




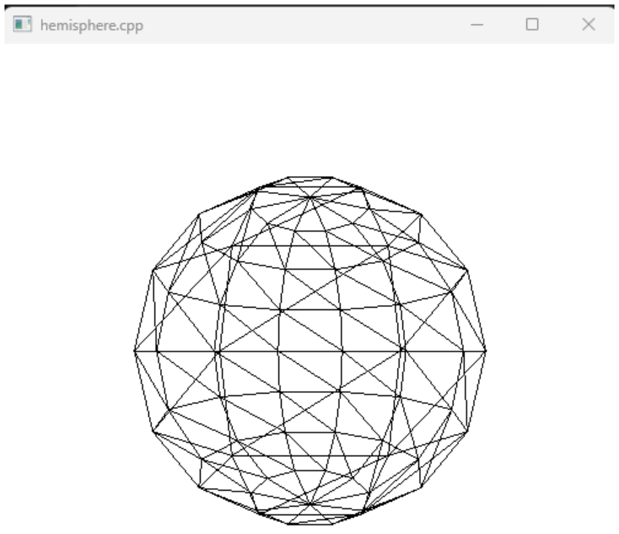
⇨ Press Q/q to increase/decrease the number of latitudinal slices

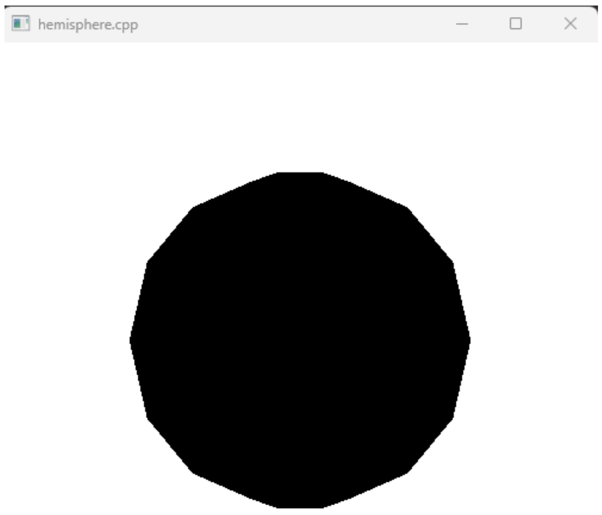




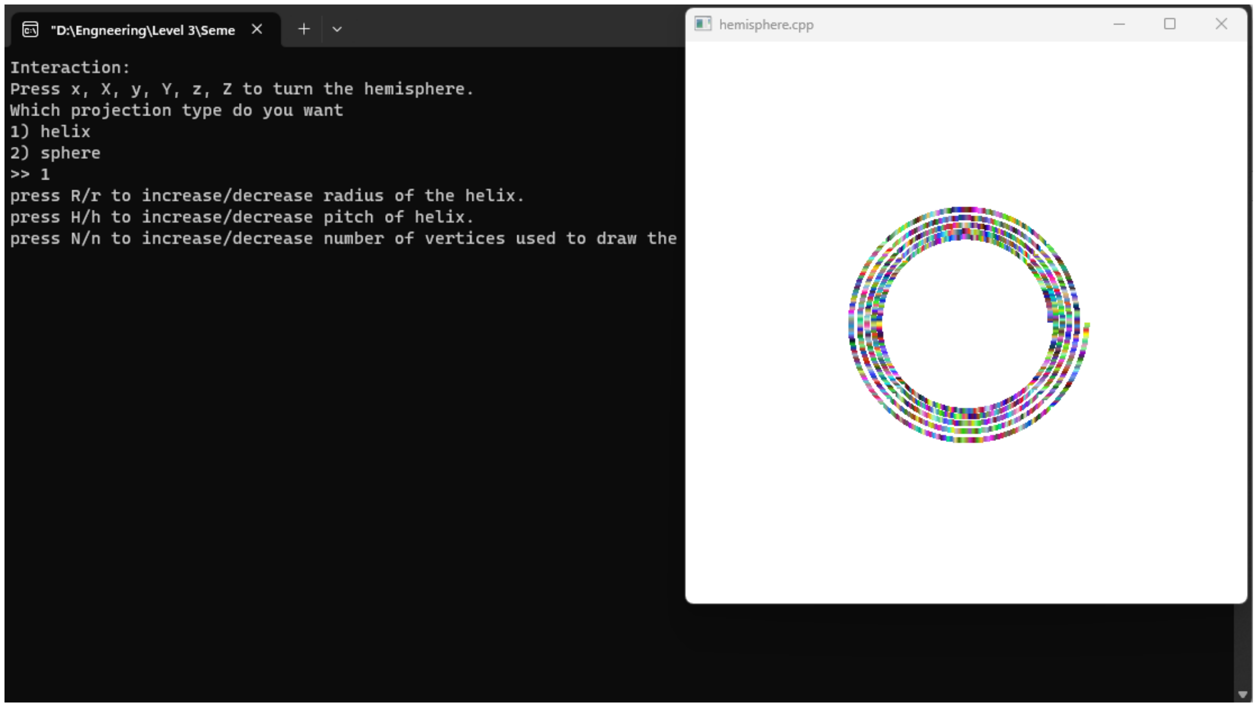


⇨ press W/w, draw sphere in wireframe / draw filled sphere.



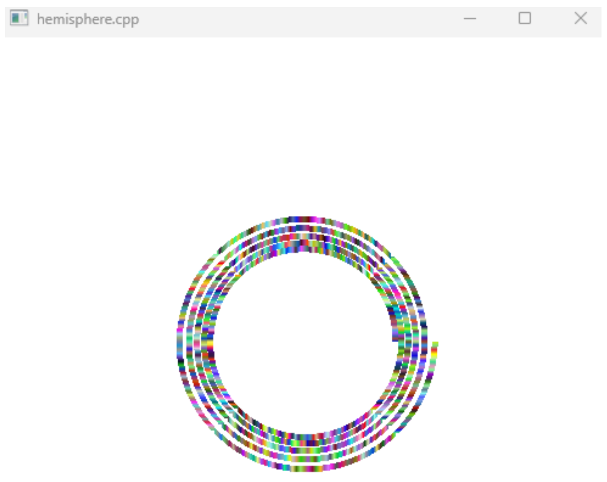


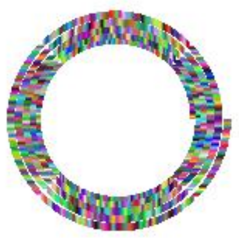
⇨ draw helix

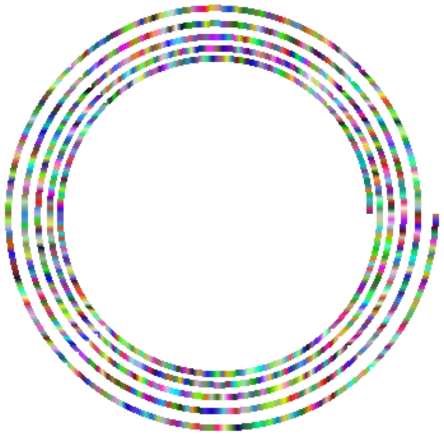


```
Interaction:
Press x, X, y, Y, z, Z to turn the hemisphere.
Which projection type do you want
1) helix
2) sphere
>> 1
press R/r to increase/decrease radius of the helix.
press H/h to increase/decrease pitch of helix.
press N/n to increase/decrease number of vertices used to draw the
```

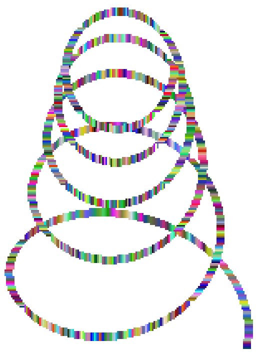
⇨ press R/r to increase/decrease radius of the helix.

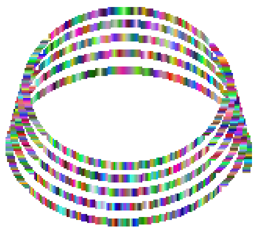






⇨ press H/h to increase/decrease pitch of helix.





⇨ press N/n to increase/decrease number of vertices used to draw the helix.