Kingdom of Saudi Arabia
المملكة العربية السعودية
Ministry of Education
وزارة التعليم
Umm AlQura University
جامعة أم القرى
Adham University College
الكلية الجامعية بأضم
Computer Science Department
قسم الحاسب الآلي

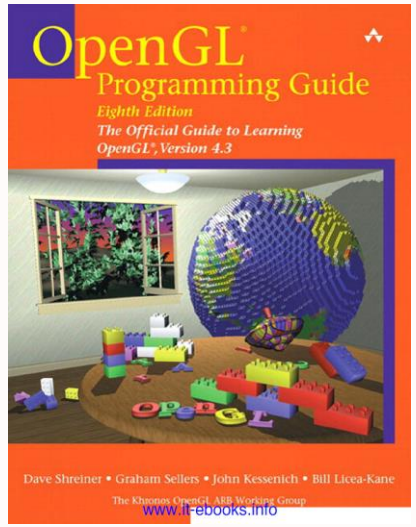
CS Department

Computer Graphics Course, 3-6803430

T.Mariah Khayat

References

- *Lab Lectures, Computer Graphics, Taif University, Faculty Of Computers And Information Technology, TA. Maha Thafar &TA. Haifa Alshehri, TA.Sohair Soliman & L.Shakila Bano.*
- *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 4.3, 8th edition, Dave Shreiner, Graham Sellers, John Kessenich, Bill Licea-Kane & The Khronos OpenGL ARB Working Group, Addison-Wesley.*



Lecture Twelve

Animation

content

1. Animation and OpenGL
2. Writing Texts Using OpenGL
3. Spider Animation Program
4. Animation and Interaction Program
5. Solar System Animation Program

Animation and Open GL

Animate:

- To give life or motion

Animation:

- A group of techniques that can make visual content appear to move.

Animation using double buffering:

① Request a double buffered color buffer

```
glutInitDisplayMode( GLUT_RGB | GLUT_DOUBLE );
```

② Clear color buffer

```
glClear( GL_COLOR_BUFFER_BIT );
```

③ Render scene

④ Request swap of front and back buffers

```
glutSwapBuffers();
```

- Repeat steps 2 - 4 for animation

Writing Texts Using OpenGL

You can write a text by the following instructions:

1. Use the function: `glutBitmapCharacter(void *font, int char)`
   - ✓ `Char =` letter, symbol or the variable that store string
2. Define `font` variable that determine the type, shape and size of the font and it can take the values:
   - ✓ `GLUT_BITMAP_TIMES_ROMAN_10 or 24`
   - ✓ `GLUT_BITMAP_HELVETICA_10 or 12 or 18`
3. Use: `glRasterPos{2,3,4}{I,d,s,f}(x,y,z,w)` to determine the position.
4. Define the function "Text" as in the program to write string

To write 3D text use:

1. `void glutStrokeCharacter(void *font,int char)`
   - ✓ `Font = GLUT_STROKE_ROMAN , GLUT_STROKE_MONO_ROMAN`
2. Using with `glTranslatef(..) , glRotatef(..)`

Spider Animation Program

```
#include<windows.h>
#include<GL/glut.h>
float x = 0;
void *font = GLUT_BITMAP_TIMES_ROMAN_24; //Type and Size of the Text.
void Text(int x, int y, char *Stringg) { //The Function for the Text.
    int i, l;
    glRasterPos2i(x, y);
    l = (int)strlen(Stringg); //The Length of the String
    for (i = 0; i < l; i++)
        glutBitmapCharacter(font, Stringg[i]);}
void Init() {
    glClearColor(0.0, 1.0, 0.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-100, 100, -100, 100, -110, 110);
    glMatrixMode(GL_MODELVIEW); }
```

Spider Animation Program

```
void Draw() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_COLOR_MATERIAL);
    //Net
    glPushMatrix();
    glColor3f(1, 1, 1);
    glTranslatef(0, 0, -100);
    glutWireSphere(95, 15, 15);
    glPopMatrix();
glPushMatrix();
glTranslatef(x, 0, 0);
    //Body of Spider
    glPushMatrix();
    glColor3f(1.0, 0.8, 0.8);
    glTranslatef(-50, 0, 0);
    glutSolidSphere(13, 100, 100);
    glPopMatrix();
    //Head of Spider
    glPushMatrix();
    glColor3f(0, 0, 0);
    glTranslatef(-37, 0,0);
    glutSolidSphere(8, 100, 100);
    glPopMatrix();
    //Eye1
    glPushMatrix();
    glColor3f(1, 0, 0);
    glTranslatef(-30, 1.5, 0);
    glutSolidSphere(2, 100, 100);
    glPopMatrix();
    //Eye2
    glPushMatrix();
    glColor3f(1, 0, 0);
    glTranslatef(-30, -1.5, 0);
    glutSolidSphere(2, 100, 100);
    glPopMatrix();
    //4 Legs
    glColor3f(0.8, 0.8, 0.7);
    glLineWidth(4);
    glBegin(GL_LINES);
    glVertex2f(-55, 10);
    glVertex2f(-55, 20);
    glVertex2f(-45, 10);
    glVertex2f(-45, 10);
    glVertex2f(-55, -10);
    glVertex2f(-55, -20);
    glVertex2f(-45, -10);
    glVertex2f(-45, -20);
    glEnd();
    if (x <= 100)
        x+= 0.2;
    else
        x = 100;
    glPopMatrix(); //End of the Mode
    glPushMatrix(); //The Other Bug
    glColor3f(0, 0, 1);
    glTranslatef(77, 4, -3);
    glutSolidSphere(5, 100, 100);
    glTranslatef(0, -5, 0);
    glutSolidSphere(7, 50, 50);
    glTranslatef(0, -7, 0);
    glutSolidSphere(9, 50, 50);
    glPopMatrix();
    glDisable(GL_LIGHTING);
    glDisable(GL_LIGHT0);
    //Trunk of Tree
    glBegin(GL_QUAD_STRIP);
    glColor3ub(147, 73, 1);
    glVertex3f(-80, 100, -105);
    glVertex3f(-80, -100, -105);
    glColor3ub(210, 106, 2);
    glVertex3f(0, 100, -105);
    glVertex3f(0, -100, -105);
    glColor3ub(88, 44, 1);
    glVertex3f(80, 100, -105);
    glVertex3f(80, -100, -105);
    glEnd();
    //Write the Text
    glColor3f(0,0,0);
    Text(-100, 0, "Spider>>");
    glColor3f(1,0.5,0.5);
    Text(-14, 80, "#NET#");
    glColor3f(0,0,1);
    Text(62, -25, "^Bug^");
    glutSwapBuffers(); }
```

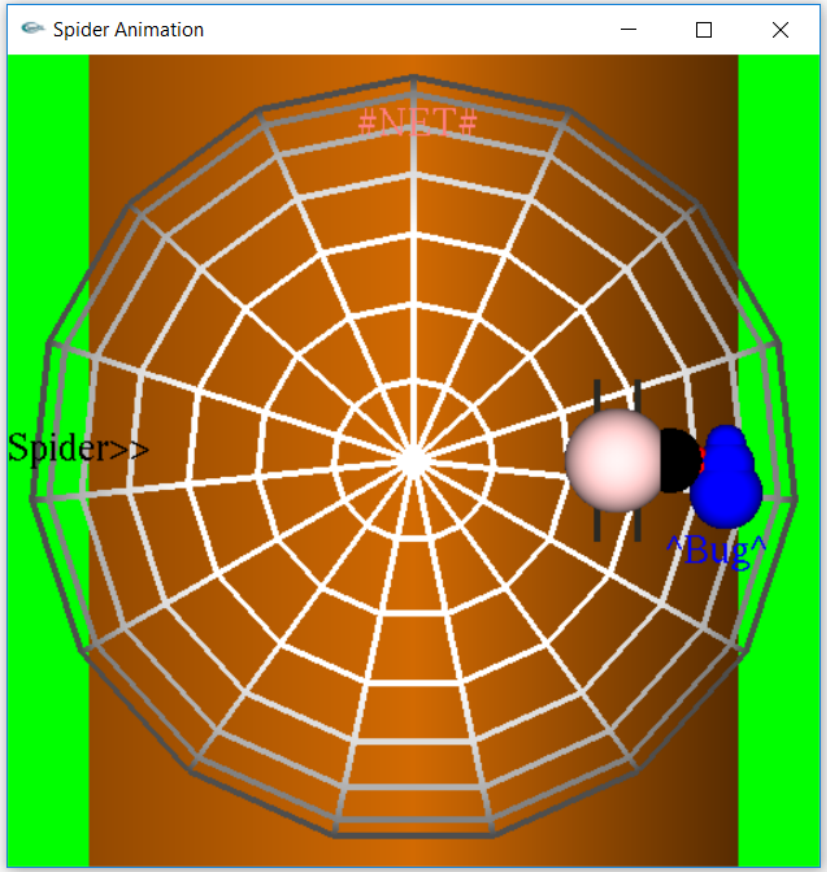
Spider Animation Program

```
int main() {
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(0,0);
    glutCreateWindow("Spider Animation");
    glutDisplayFunc(Draw);
    glutIdleFunc(Draw);
    Init();
    glutMainLoop();
    return 0; }
```

Spider Animation Program

main.c | SpiderAnimation.cpp | SpiderAnimation.layout

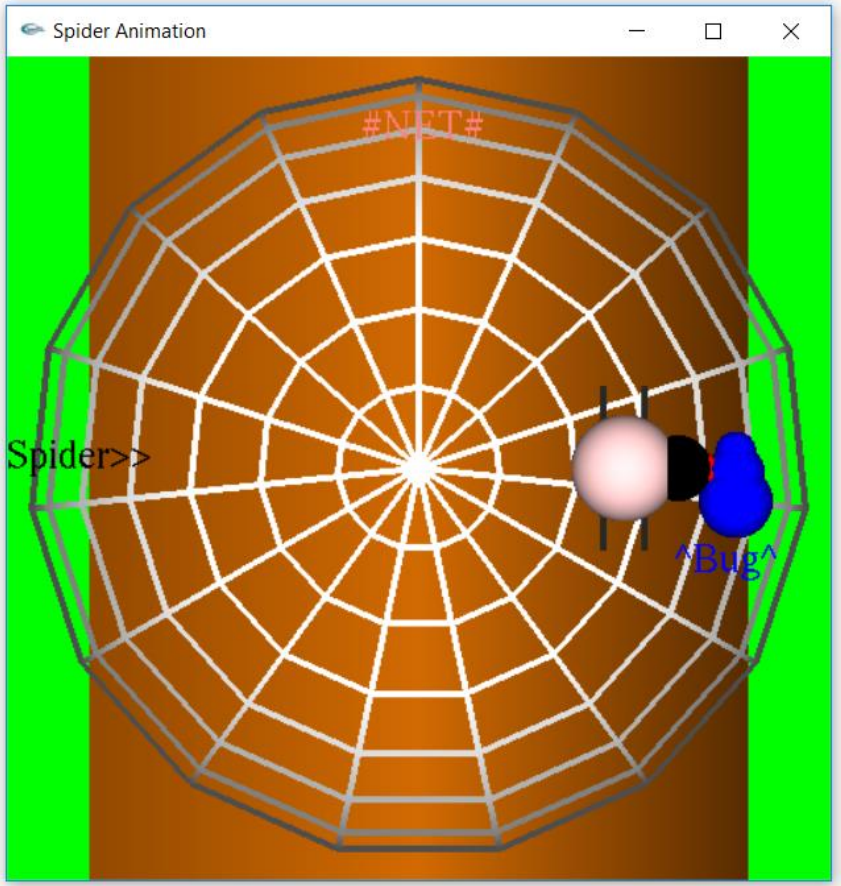
```
#include<windows.h>
#include<GL/glut.h>
float x = 0;
void *font = GLUT_BITMAP_TIMES_ROMAN_24; //Type and Size of the Text.
void Text(int x, int y, char *Stringg) { //The Function for the Text.
    int i, l;
    glRasterPos2i(x, y);
    l = (int)strlen(Stringg); //The Length of the String
    for (i = 0; i < l; i++)
        glutBitmapCharacter(font, Stringg[i]);}
void Init() {
    glClearColor(0.0, 1.0, 0.0, 0.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(-100, 100, -100, 100, -110, 110);
    glMatrixMode(GL_MODELVIEW); }
void Draw() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glEnable(GL_DEPTH_TEST);
    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);
    glEnable(GL_COLOR_MATERIAL);
    //Net
    glPushMatrix();
    glColor3f(1, 1, 1);
    glTranslatef(0, 0, -100);
    glutWireSphere(95, 15, 15);
    glPopMatrix();
glPushMatrix();
glTranslatef(x, 0, 0);
    //Body of Spider
    glPushMatrix();
    glColor3f(1.0, 0.8, 0.8);
    glTranslatef(-50, 0, 0);
    glutSolidSphere(13, 100, 100);
    glPopMatrix();
```



Spider Animation Program

main.c | SpiderAnimation.cpp | SpiderAnimation.layout

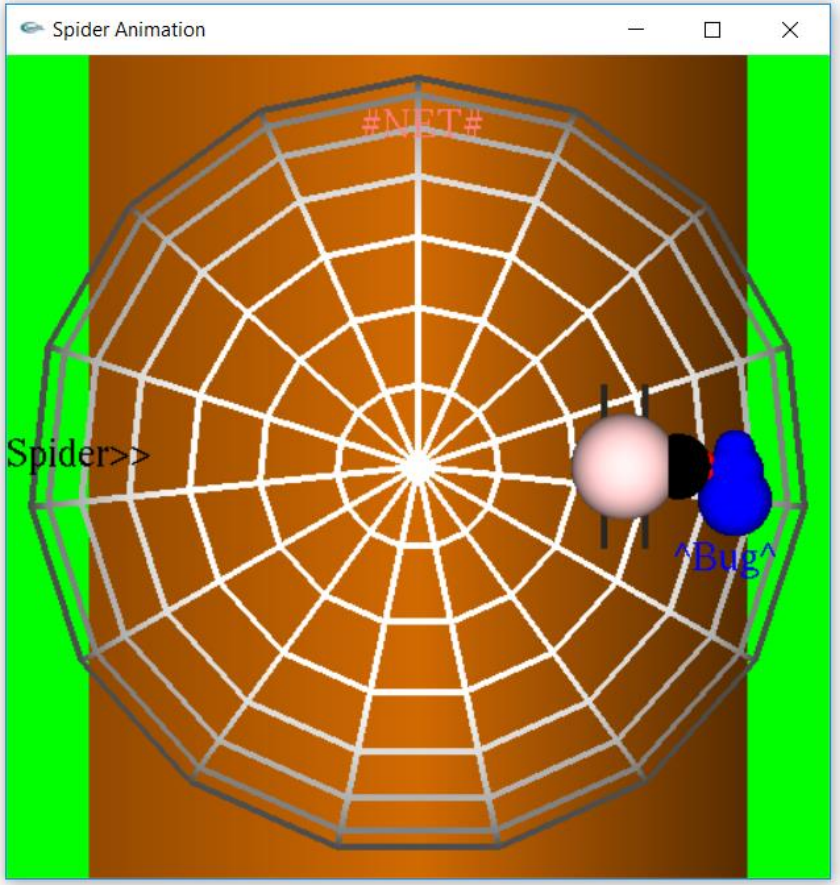
```
29    glPushMatrix();
30    glTranslatef(x, 0, 0);
31        //Body of Spider
32        glPushMatrix();
33        glColor3f(1.0, 0.8, 0.8);
34        glTranslatef(-50, 0, 0);
35        glutSolidSphere(13, 100, 100);
36        glPopMatrix();
37        //Head of Spider
38        glPushMatrix();
39        glColor3f(0, 0, 0);
40        glTranslatef(-37, 0,0);
41        glutSolidSphere(8, 100, 100);
42        glPopMatrix();
43        //Eye1
44        glPushMatrix();
45        glColor3f(1, 0, 0);
46        glTranslatef(-30, 1.5, 0);
47        glutSolidSphere(2, 100, 100);
48        glPopMatrix();
49        //Eye2
50        glPushMatrix();
51        glColor3f(1, 0, 0);
52        glTranslatef(-30, -1.5, 0);
53        glutSolidSphere(2, 100, 100);
54        glPopMatrix();
55        //4 Legs
56        glColor3f(0.8, 0.8, 0.7);
57        glLineWidth(4);
58        glBegin(GL_LINES);
59        glVertex2f(-55, 10);
60        glVertex2f(-55, 20);
61        glVertex2f(-45, 10);
62        glVertex2f(-45, 20);
63        glVertex2f(-55, -10);
64        glVertex2f(-55, -20);
```



Spider Animation Program

main.c | SpiderAnimation.cpp | SpiderAnimation.layout

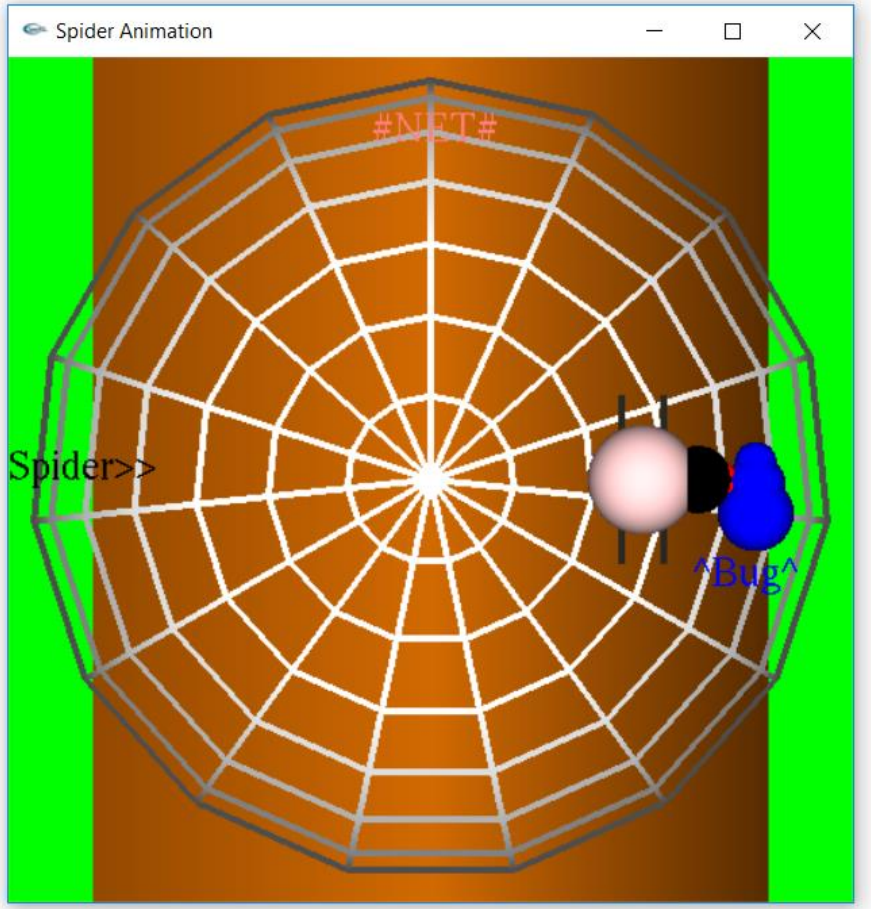
```
55      //4 Legs
56      glColor3f(0.8, 0.8, 0.7);
57      glLineWidth(4);
58      glBegin(GL_LINES);
59      glVertex2f(-55, 10);
60      glVertex2f(-55, 20);
61      glVertex2f(-45, 10);
62      glVertex2f(-45, 20);
63      glVertex2f(-55, -10);
64      glVertex2f(-55, -20);
65      glVertex2f(-45, -10);
66      glVertex2f(-45, -20);
67      glEnd();
68      if (x <= 100)
69          x+= 0.2;
70      else
71          x = 100;
72      glPopMatrix(); //End of the Mode
73      glPushMatrix(); //The Other Bug
74      glColor3f(0, 0, 1);
75      glTranslatef(77, 4, -3);
76      glutSolidSphere(5, 100, 100);
77      glTranslatef(0, -5, 0);
78      glutSolidSphere(7, 50, 50);
79      glTranslatef(0, -7, 0);
80      glutSolidSphere(9, 50, 50);
81      glPopMatrix();
82      glDisable(GL_LIGHTING);
83      glDisable(GL_LIGHT0);
84      //Trunk of Tree
85      glBegin(GL_QUAD_STRIP);
86      glColor3ub(147, 73, 1);
87      glVertex3f(-80, 100, -105);
88      glVertex3f(-80, -100, -105);
89      glColor3ub(210, 106, 2);
90      glVertex3f(0, 100, -105);
```



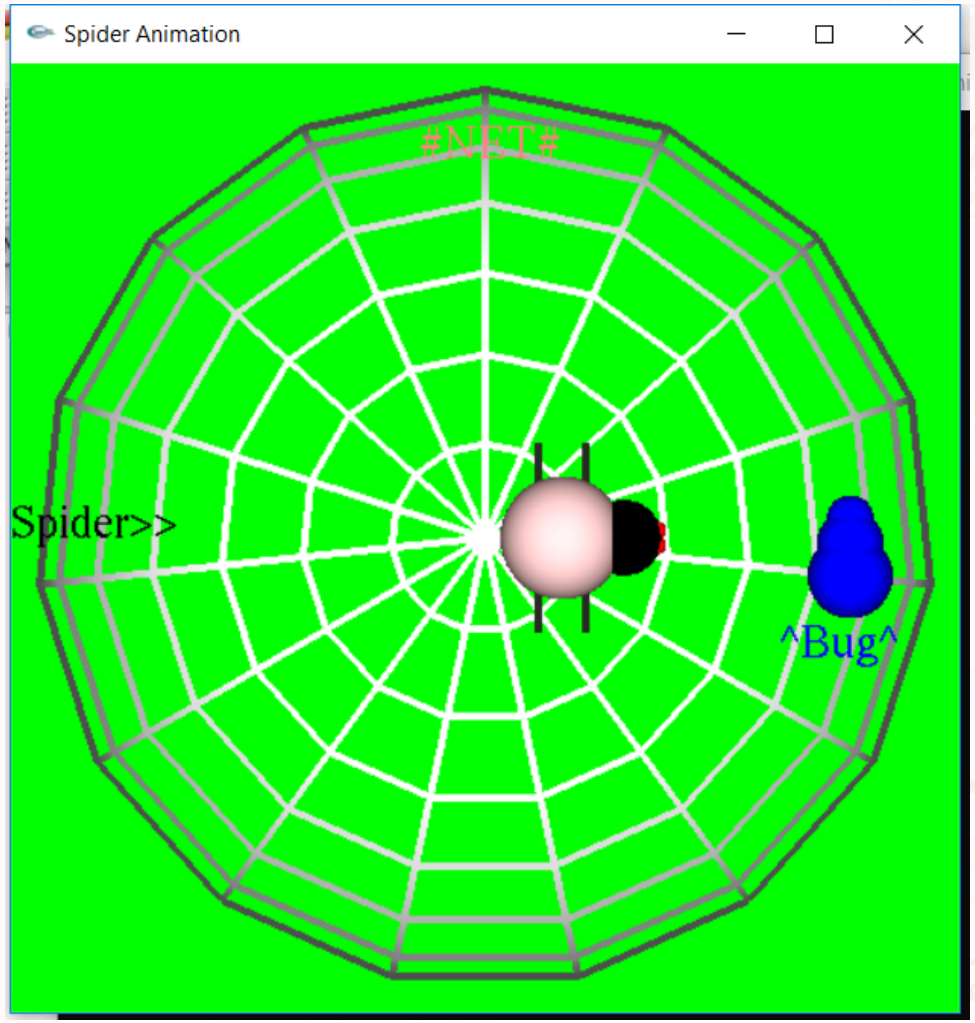
Spider Animation Program

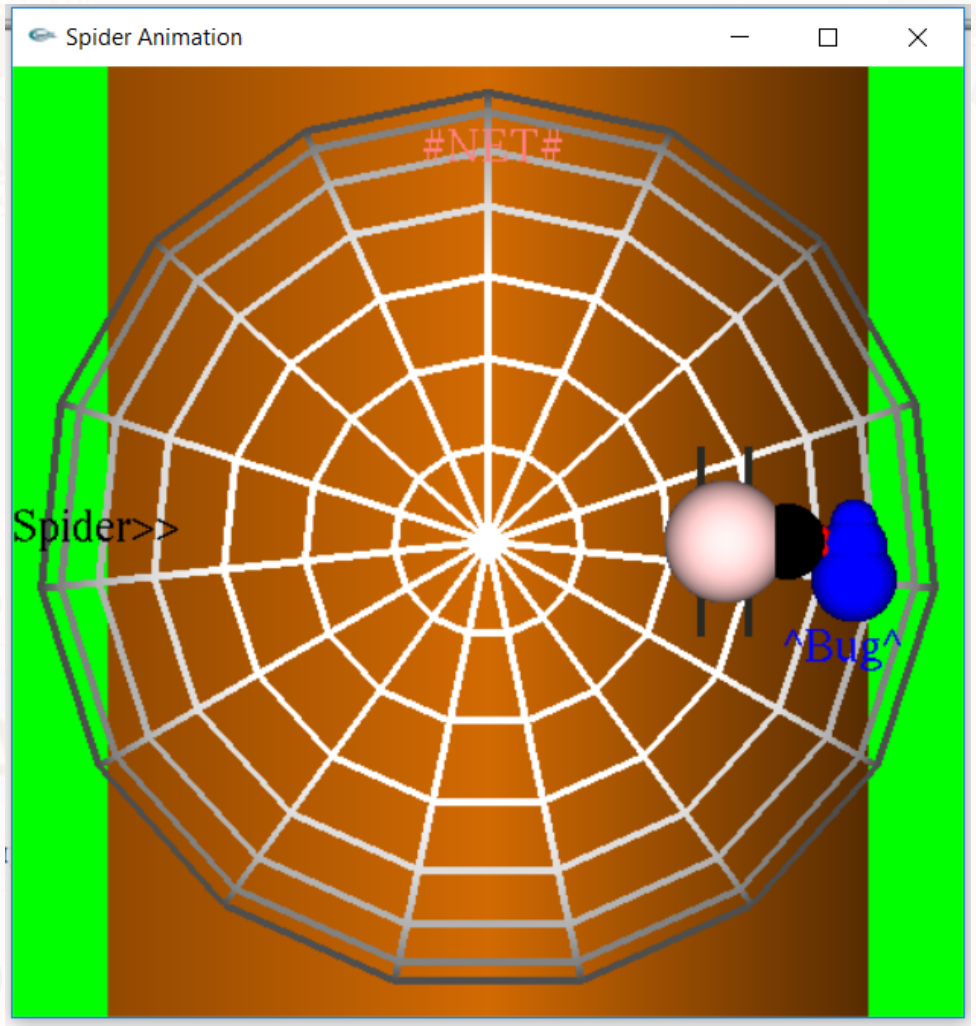
main.c | SpiderAnimation.cpp | SpiderAnimation.layout

```
80      glutSolidSphere(9, 50, 50);
81      glPopMatrix();
82      glDisable(GL_LIGHTING);
83      glDisable(GL_LIGHT0);
84      //Trunk of Tree
85      glBegin(GL_QUAD_STRIP);
86      glColor3ub(147, 73, 1);
87      glVertex3f(-80, 100, -105);
88      glVertex3f(-80, -100, -105);
89      glColor3ub(210, 106, 2);
90      glVertex3f(0, 100, -105);
91      glVertex3f(0, -100, -105);
92      glColor3ub(88, 44, 1);
93      glVertex3f(80, 100, -105);
94      glVertex3f(80, -100, -105);
95      glEnd();
96      //Write the Text
97      glColor3f(0,0,0);
98      Text(-100, 0, "Spider>>");
99      glColor3f(1,0.5,0.5);
100     Text(-14, 80, "#NET#");
101     glColor3f(0,0,1);
102     Text(62, -25, "^Bug^");
103     glutSwapBuffers(); }
104 int main() {
105     glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
106     glutInitWindowSize(500, 500);
107     glutInitWindowPosition(0,0);
108     glutCreateWindow("Spider Animation");
109     glutDisplayFunc(Draw);
110     glutIdleFunc(Draw);
111     Init();
112     glutMainLoop();
113     return 0; }
```



Spider Animation Program





Animation and Interaction Program

```
#include<windows.h>
#include<GL/glut.h>
int RunMode = 1; //Used as a boolean (1 or 0) for "ON" and "OFF"
float CurrentAngle = 0;
float AnimateStep =  1;
void Keyboard (unsigned char key, int x, int y) {
    switch (key) {
        case 27: exit(1); //Esc Key
        case 'a':
        case 'A':
        RunMode = 1; break;
        case 's':
        case 'S':
        RunMode = 0; break; }
    glutPostRedisplay(); }
void SpecialKeys (int key, int x, int y) {
    if (key == GLUT_KEY_UP)
        AnimateStep += 0.1;
    if (key == GLUT_KEY_DOWN)
        AnimateStep -= 0.1;
        glutPostRedisplay(); }

void DrawScene() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    if (RunMode == 1) {
        CurrentAngle += AnimateStep;
    }
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glColor3f(1, 0, 0);
    glTranslatef(1.5, 1.5, 0.0);
    glRotatef(CurrentAngle, 0, 0, 1);
    glTranslatef(-1.5, -1.5, 0.0);
```

Animation and Interaction Program

```
 //Draw 3 Triangles:
    glBegin(GL_TRIANGLES);
    glColor3f(1, 0, 0);
    glVertex3f(0.3, 1, 0.5);
    glVertex3f(2.7, 0.8, 0);
    glVertex3f(2.7, 1.15, 0);

    glColor3f(0, 1, 0);
    glVertex3f(2.5, 0.7, 0.5);
    glVertex3f(1.4, 2.8, 0);
    glVertex3f(1.2, 2.7, 0);

    glColor3f(0, 0, 1);
    glVertex3f(1.6, 2.7, 0.5);
    glVertex3f(0.3, 0.7, 0);
    glVertex3f(0.5, 0.6, 0);
    glEnd();
    glutSwapBuffers();

    if (RunMode == 1)
        glutPostRedisplay();
}

void resizeWindow(int w, int h) {
    glViewport(0, 0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0, 3, 0, 3, -1, 1);
}

int main() {
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowPosition(10, 60);
    glutInitWindowSize(600, 600);
    glutCreateWindow("Simple Animation!");
    glutKeyboardFunc(Keyboard);
    glutSpecialFunc(SpecialKeys);
    glutReshapeFunc(resizeWindow);
    glutDisplayFunc(DrawScene);
    glutMainLoop();
    return 0;
}
```

Animation and Interaction Program

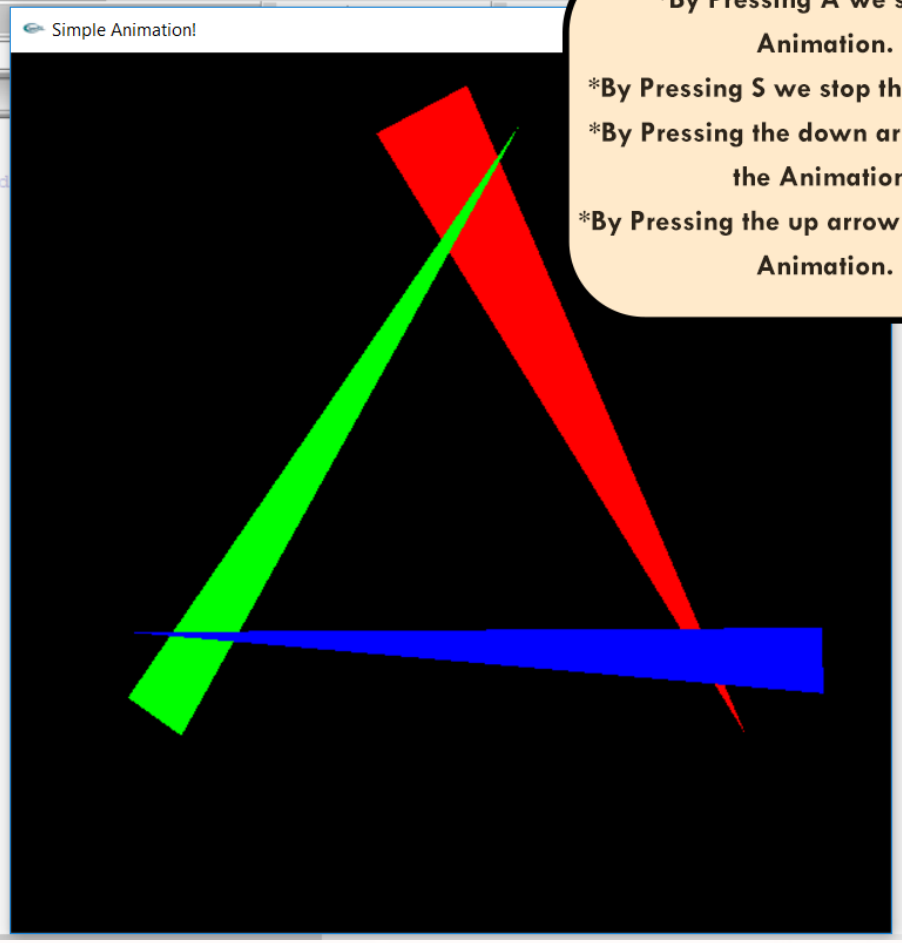
ionAnimation] - Code::Blocks 17.12

uild Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

Debug

AnimationandInteraction.cpp

```
#include<windows.h>
#include<GL/glut.h>
int RunMode = 1; //Used as a boolean (1 or 0) for "ON" and
float CurrentAngle = 0;
float AnimateStep =  1;
void Keyboard (unsigned char key, int x, int y) {
    switch (key) {
        case 27: exit(1); //Esc Key
        case 'a':
        case 'A':
        RunMode = 1; break;
        case 's':
        case 'S':
        RunMode = 0; break; }
    glutPostRedisplay(); }
void SpecialKeys (int key, int x, int y) {
    if (key == GLUT_KEY_UP)
        AnimateStep += 0.1;
    if (key == GLUT_KEY_DOWN)
        AnimateStep -= 0.1;
        glutPostRedisplay(); }
void DrawScene() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    if (RunMode == 1) {
        CurrentAngle += AnimateStep; }
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glColor3f(1, 0, 0);
    glTranslatef(1.5, 1.5, 0.0);
    glRotatef(CurrentAngle, 0, 0, 1);
    glTranslatef(-1.5, -1.5, 0.0);
    //Draw 3 Triangles:
    glBegin(GL_TRIANGLES);
```



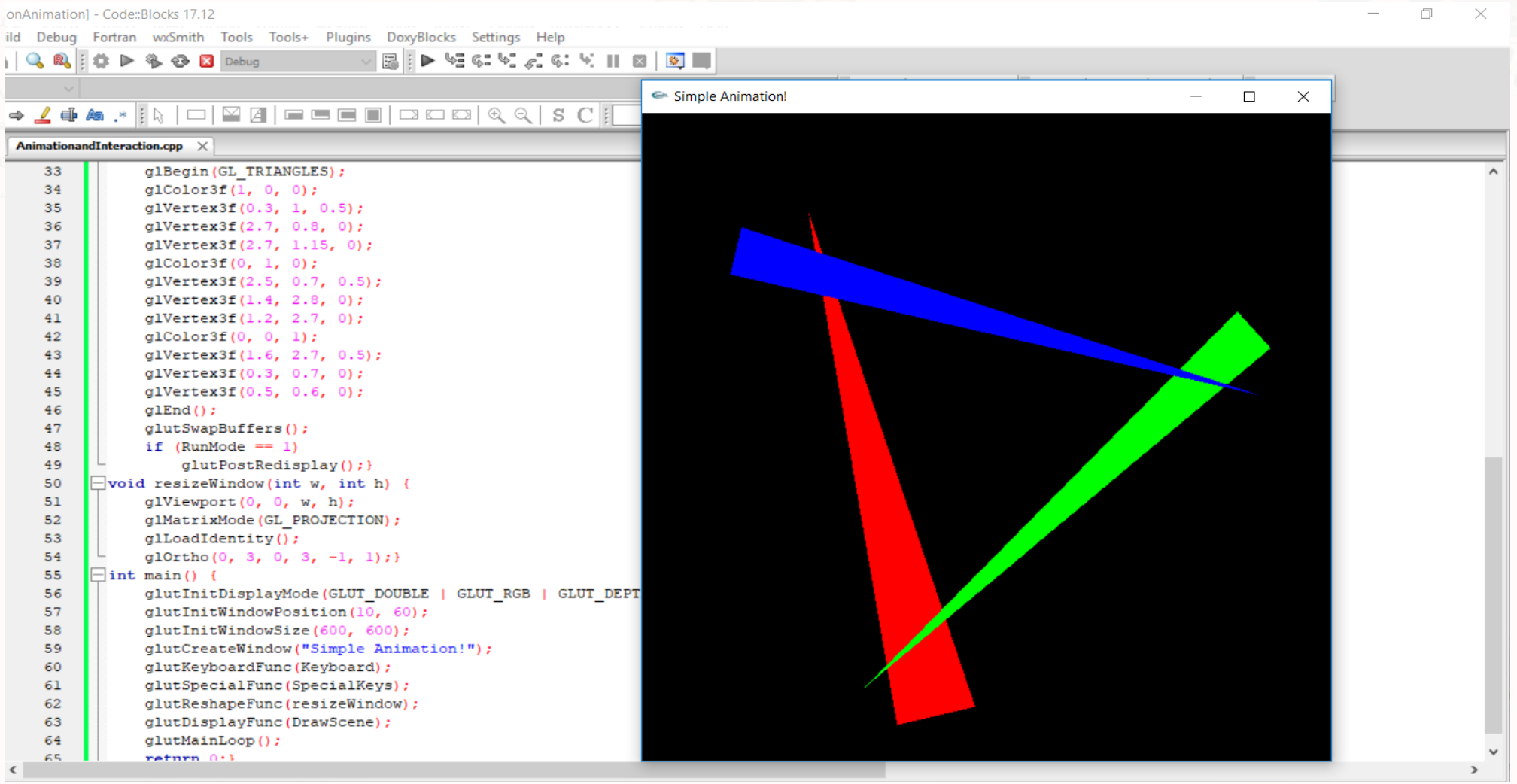
*By Pressing A we start the Animation.

*By Pressing S we stop the Animation.

*By Pressing the down arrow we slow the Animation.

*By Pressing the up arrow we speed the Animation.

Animation and Interaction Program



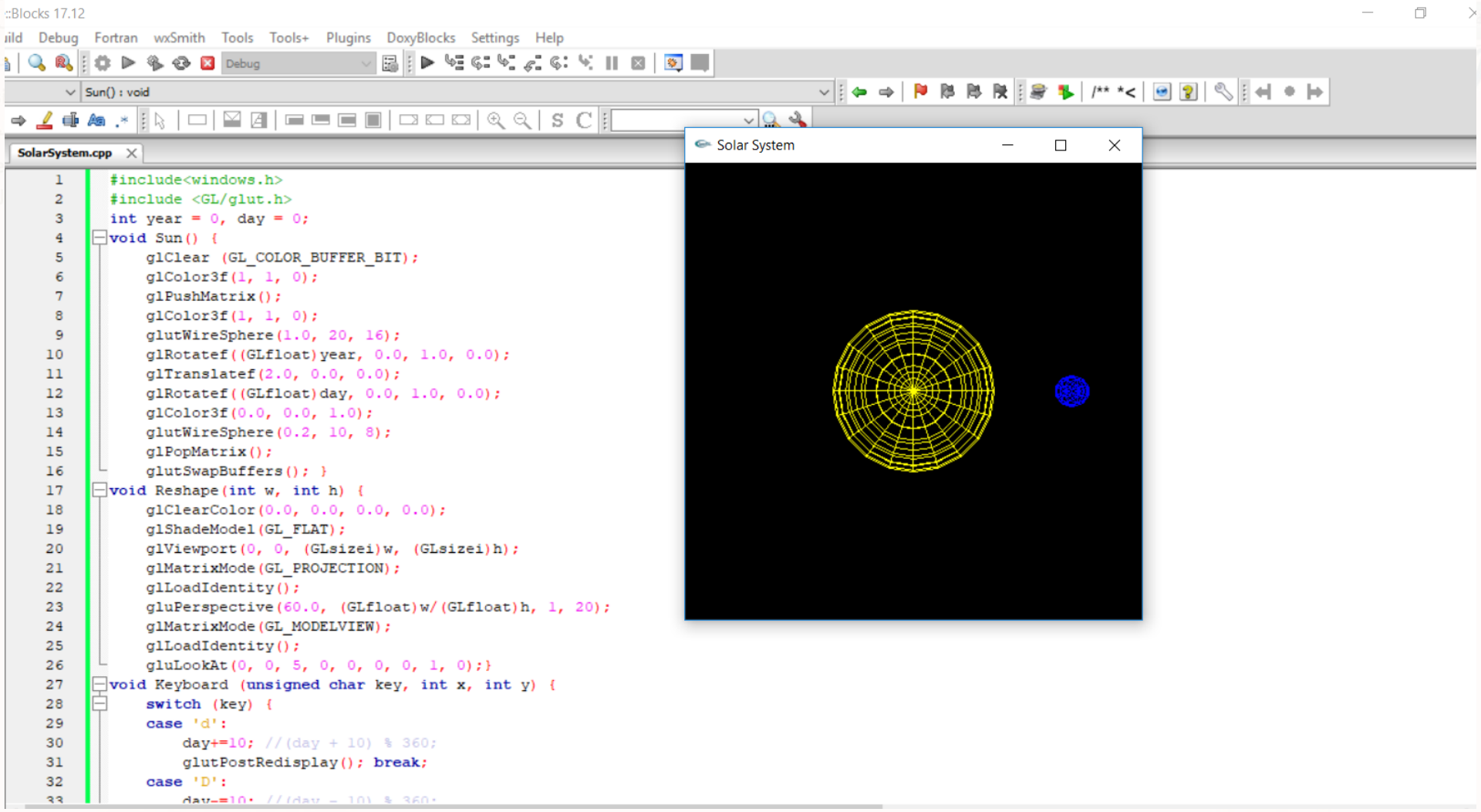
Solar System Animation Program

```
#include<windows.h>
#include <GL/glut.h>
int year = 0, day = 0;
void Sun() {
    glClear (GL_COLOR_BUFFER_BIT);
    glColor3f(1, 1, 0);
    glPushMatrix();
    glColor3f(1, 1, 0);
    glutWireSphere(1.0, 20, 16);
    glRotatef((GLfloat)year, 0.0, 1.0, 0.0);
    glTranslatef(2.0, 0.0, 0.0);
    glRotatef((GLfloat)day, 0.0, 1.0, 0.0);
    glColor3f(0.0, 0.0, 1.0);
    glutWireSphere(0.2, 10, 8);
    glPopMatrix();
    glutSwapBuffers(); }
void Reshape(int w, int h) {
    glClearColor(0.0, 0.0, 0.0, 0.0);
    glShadeModel(GL_FLAT);
    glViewport(0, 0, (GLsizei)w, (GLsizei)h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(60.0, (GLfloat)w/(GLfloat)h, 1, 20);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    gluLookAt(0, 0, 5, 0, 0, 0, 0, 1, 0);}
```

Solar System Animation Program

```
void Keyboard (unsigned char key, int x, int y) {
    switch (key) {
    case 'd':
        day+=10; //(day + 10) % 360;
        glutPostRedisplay(); break;
    case 'D':
        day-=10; //(day - 10) % 360;
        glutPostRedisplay(); break;
    case 'y':
        year+=5; //(year + 5) % 360;
        glutPostRedisplay(); break;
    case 'Y':
        year-=5; //(year - 5) % 360;
        glutPostRedisplay(); break;
    case 27:
        exit(0); break;
    default: break; }}
int main(int argc, char** argv) {
  glutInit(&argc, argv);
  glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
  glutInitWindowSize(400, 400);
  glutInitWindowPosition(100, 100);
  glutCreateWindow("Solar System");
  glutDisplayFunc(Sun);
  glutReshapeFunc(Reshape);
  glutKeyboardFunc(Keyboard);
  glutMainLoop();}
```

Solar System Animation Program



Solar System Animation Program

::Blocks 17.12

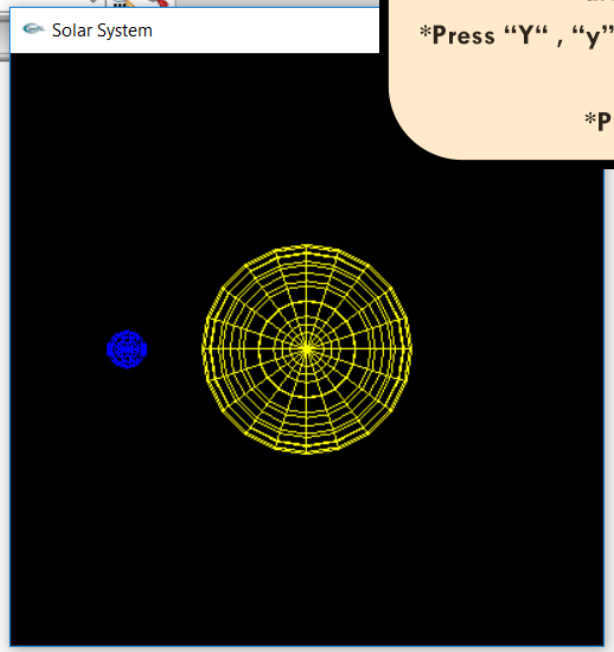
ild Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help

Debug

Sun() : void

SolarSystem.cpp

```
23      gluPerspective(60.0, (GLfloat)w/(GLfloat)h, 1, 20);
24      glMatrixMode(GL_MODELVIEW);
25      glLoadIdentity();
26      gluLookAt(0, 0, 5, 0, 0, 0, 0, 1, 0);}
27  void Keyboard (unsigned char key, int x, int y) {
28      switch (key) {
29      case 'd':
30          day+=10; //(day + 10) % 360;
31          glutPostRedisplay(); break;
32      case 'D':
33          day-=10; //(day - 10) % 360;
34          glutPostRedisplay(); break;
35      case 'y':
36          year+=5; //(year + 5) % 360;
37          glutPostRedisplay(); break;
38      case 'Y':
39          year-=5; //(year - 5) % 360;
40          glutPostRedisplay(); break;
41      case 27:
42          exit(0); break;
43      default: break; }}
44  int main(int argc, char** argv) {
45    glutInit(&argc, argv);
46    glutInitDisplayMode (GLUT_DOUBLE | GLUT_RGB);
47    glutInitWindowSize(400, 400);
48    glutInitWindowPosition(100, 100);
49    glutCreateWindow("Solar System");
50    glutDisplayFunc(Sun);
51    glutReshapeFunc(Reshape);
52    glutKeyboardFunc(Keyboard);
53    glutMainLoop();}
54
```

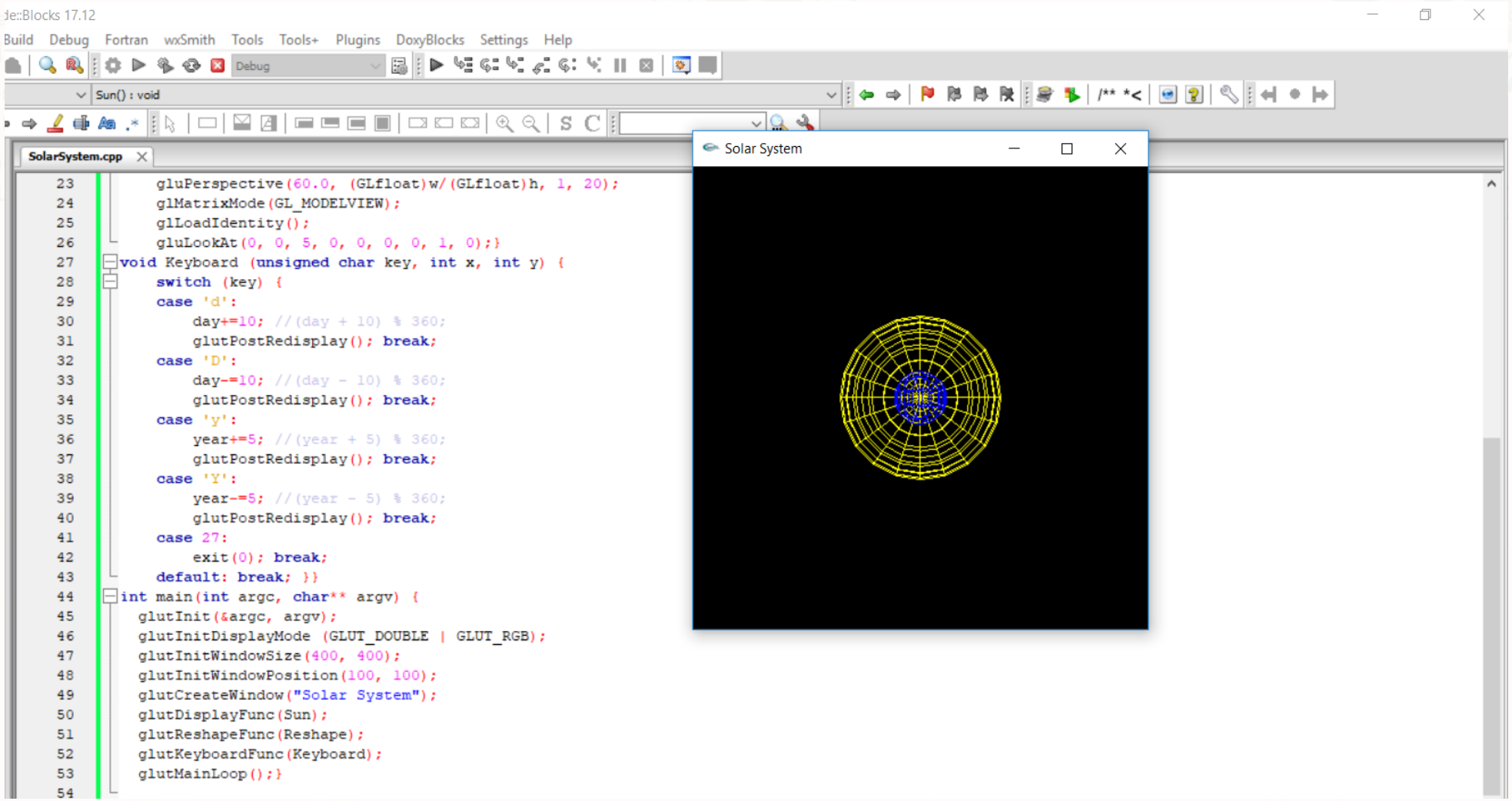


*Press “D“ , ”d” to rotate the earth around itself (Day)

*Press “Y“ , “y” to rotate the earth around the sun (Year)

*Press ‘Esc’ to exit.

Solar System Animation Program



وصلى الله وبارك على نبينا محمد

The End Summary of Lecture Twelve

T.Mariah Khayat
الأستاذة/ مارية خياط
Adham University College
الكلية الجامعية بأضم

mskhayat@uqu.edu.sa