

```

1 #include <windows.h>
2 #include<string.h>
3 #include<stdarg.h>
4 #include<stdio.h>
5 #include <glut.h>
6 static double x=0.0;
7
8 void stroke_output(GLfloat x, GLfloat y, char *format,...)
9 {
10     va_list args;
11     char buffer[200], *p;
12     va_start(args, format);
13     vsprintf(buffer, format, args);
14     va_end(args);
15     glPushMatrix();
16     glTranslatef(-2.5, y, 0);
17     glScaled(0.003, 0.005, 0.005);
18     for (p = buffer; *p; p++)
19         glutStrokeCharacter(GLUT_STROKE_ROMAN, *p);
20     glPopMatrix();
21 }
22
23 //changing backgroun color
24 void d4()
25 {
26
27     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
28     glClearColor(0.0,0.0,0.0,1.0);
29     glLoadIdentity();
30     glTranslatef(0.0f,0.0f,-13.0f);
31     stroke_output(-2.0, 1.7, "Wel Come");
32     stroke_output(-2.0, 0.9, "To");
33     stroke_output(-2.0, 0.0, "Project Created");
34     stroke_output(-2.0, -0.9, "By");
35     stroke_output(-2.0, -1.8, "");
36     glFlush();
37     glutSwapBuffers();
38 }
39
40
41 void d5()
42 {
43     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
44     glClearColor(1.0,1.0,1.0,1.0);
45     glLoadIdentity();
46     glTranslatef(0.0f,0.0f,-13.0f);
47     stroke_output(-2.0, 1.7, "Wel Come");
48     stroke_output(-2.0, 0.9, "To");
49     stroke_output(-2.0, 0.0, "Project Created");
50     stroke_output(-2.0, -0.9, "By");
51     stroke_output(-2.0, -1.8, "");
52     glFlush();
53     glutSwapBuffers();
54 }
55
56 void d6()
57 {
58     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
59     glClearColor(0.7,0.3,.2,1.0);
60     glLoadIdentity();
61     glTranslatef(0.0f,0.0f,-13.0f);
62     stroke_output(-2.0, 1.7, "Wel Come");
63     stroke_output(-2.0, 0.9, "To");
64     stroke_output(-2.0, 0.0, "Project Created");
65     stroke_output(-2.0, -0.9, "By");
66     stroke_output(-2.0, -1.8, "");
67     glFlush();
68     glutSwapBuffers();
69 }
70
71 void d7()
72 {

```

```

73     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
74     glClearColor(0.5,0.7,0.3,1.0);
75     glLoadIdentity();
76     glTranslatef(0.0f,0.0f,-13.0f);
77     stroke_output(-2.0, 1.7, "Wel Come");
78     stroke_output(-2.0, 0.9, "To");
79     stroke_output(-2.0, 0.0, "Project Created");
80     stroke_output(-2.0, -0.9, "By");
81     stroke_output(-2.0, -1.8, "");
82     glFlush();
83     glutSwapBuffers();
84 }
85
86
87
88 void flying(double ang)
89 {
90     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
91     glLoadIdentity();
92     glTranslatef(0.0f,-.5f,-13.0f);
93     glRotatef(115,1.0f,0.0f,0.0f);
94
95     //fan
96     glPushMatrix();
97     glRotatef(ang,0.0f,0.0f,1.0f);
98     glScaled(2.9,0.2,0.1);
99     glTranslatef(0.0,0.0,16.0);
100    glRotatef(ang,0.0f,0.0f,1.0f);
101    glutSolidSphere(0.7,20,60);
102    glPopMatrix();
103
104    //2nd blade
105    glPushMatrix();
106    glRotatef(ang,0.0f,0.0f,1.0f);
107    glScaled(0.2,2.9,0.1);
108    glTranslatef(0.0,0.0,16.0);
109    glRotatef(ang,0.0f,0.0f,1.0f);
110    glutSolidSphere(0.7,20,60);
111    glPopMatrix();
112
113    glPushMatrix();
114    glutWireCone(4,3,80,120);
115    glPopMatrix();
116
117    glPushMatrix();
118    glRotated(ang,0.0,1.0,0.0);
119    glTranslatef(0.05,-3.0,0.0);
120    glRotated(ang,0.0,1.0,0.0);
121    glutSolidSphere(0.3,20,60);
122    glPopMatrix();
123    glFlush();
124    glutSwapBuffers();
125 }
126
127
128 void fly()
129 {
130     x += 5.30;
131     flying(x);
132 }
133
134
135 void nofly()
136 {
137     flying(0);
138 }
139
140
141 void doInit()
142 {
143
144     /* Background and foreground color */

```

```

145     glClearColor(0.0,0.0,0.0,0.0);
146     glColor3f(.0,1.0,1.0);
147     glViewport(0,0,640,480);
148
149     /* Select the projection matrix and reset it then
150     setup our view perspective */
151     glMatrixMode(GL_PROJECTION);
152     glLoadIdentity();
153     gluPerspective(30.0f,(GLfloat)640/(GLfloat)480,0.1f,200.0f);
154     /* Select the modelview matrix, which we alter with rotatef() */
155     glMatrixMode(GL_MODELVIEW);
156     glLoadIdentity();
157     glClearDepth(2.0f);
158     glEnable(GL_DEPTH_TEST);
159     glDepthFunc(GL_LEQUAL);
160 }
161
162 void doDisplay()
163 {
164     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
165     glLoadIdentity();
166     glTranslatef(0.0f,0.0f,-13.0f);
167     stroke_output(-2.0, 1.7, "Wel Come");
168     stroke_output(-2.0, 0.9, "To");
169     stroke_output(-2.0, 0.0, "Project Created");
170     stroke_output(-2.0, -0.9, "By");
171     stroke_output(-2.0, -1.8, "");
172
173     GLfloat mat_ambient[]={0.0f,1.0f,2.0f,1.0f};
174     GLfloat mat_diffuse[]={0.0f,1.5f,.5f,1.0f};
175     GLfloat mat_specular[]={5.0f,1.0f,1.0f,1.0f};
176     GLfloat mat_shininess[]={50.0f};
177     glMaterialfv(GL_FRONT,GL_AMBIENT,mat_ambient);
178     glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
179     glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
180     glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);
181
182     /*GLfloat lightIntensity[]={3.7f,0.7f,0.7f,1.0f};    Orange
183     GLfloat light_position[]={2.0f,5.0f,3.0f,1.0f};*/
184
185     /*light source properties*/
186     GLfloat lightIntensity[]={1.7f,1.7f,1.7f,1.0f};
187     GLfloat light_position[]={2.0f,0.0f,0.0f,0.0f};
188     glLightfv(GL_LIGHT0,GL_POSITION,light_position);
189     GLfloat light_position2[]={0.0f,0.0f,8.0f,0.0f};
190     glLightfv(GL_LIGHT0,GL_POSITION,light_position2);
191     GLfloat light_position3[]={0.0f,5.0f,2.0f,0.5f};
192     glLightfv(GL_LIGHT0,GL_POSITION,light_position3);
193     glLightfv(GL_LIGHT0,GL_DIFFUSE,lightIntensity);
194     glFlush();
195     glutSwapBuffers();
196
197 }
198
199 void menu(int id)
200 {
201     switch(id)
202     {
203
204     case 1:glutIdleFunc(fly);
205         break;
206     case 2:exit(0);
207         break;
208
209     }
210     glFlush();
211     glutSwapBuffers();
212     glutPostRedisplay();
213 }
214
215 void mykey(unsigned char key,int x,int y)
216 {

```

```

217     if(key=='q' || key=='Q')
218     {
219         exit(0);
220     }
221     if(key=='1')
222     {
223         glutIdleFunc(d4);
224     }
225     if(key=='2')
226     {
227         glutIdleFunc(d5);
228     }
229     if(key=='3')
230     {
231         glutIdleFunc(d6);
232     }
233     if(key=='4')
234     {
235         glutIdleFunc(d7);
236     }
237
238     if(key=='b')
239     {
240         glutIdleFunc(fly);
241     }
242     if(key=='B')
243     {
244         glutIdleFunc(nofly);
245     }
246 }
247
248 }
249
250 int main(int argc, char *argv[])
251 {
252     glutInit(&argc, argv);
253     glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB);
254     glutInitWindowSize(640,480);
255     glutInitWindowPosition(0,0);
256     glutCreateWindow("Flying ball");
257     glutDisplayFunc(doDisplay);
258     glEnable(GL_LIGHTING);
259     glEnable(GL_LIGHT0);
260     glShadeModel(GL_SMOOTH);
261     glEnable(GL_DEPTH_TEST);
262     glEnable(GL_NORMALIZE);
263     glutKeyboardFunc(mykey);
264     glutCreateMenu(menu);
265     glutAddMenuEntry("Flying ball",1);
266     glutAddMenuEntry("Exit",2);
267     glutAttachMenu(GLUT_RIGHT_BUTTON);
268     doInit();
269     glutMainLoop();
270     return 0;
271 }

```