```c
#include <windows.h>
#include<string.h>
#include<stdarg.h>
#include<stdio.h>
#include <glut.h>
static double x=0.0;

void stroke_output(GLfloat x, GLfloat y, char *format,...)
{
        va_list args;
        char buffer[200], *p;
        va_start(args, format);
        vsprintf(buffer, format, args);
        va_end(args);
        glPushMatrix();
        glTranslatef(-2.5, y, 0);
        glScaled(0.003, 0.005, 0.005);
        for (p = buffer; *p; p++)
    glutStrokeCharacter(GLUT_STROKE_ROMAN, *p);
        glPopMatrix();
}

        //changing backgroun color
void d4()
{

        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glClearColor(0.0,0.0,0.0,1.0);
        glLoadIdentity();
        glTranslatef(0.0f,0.0f,-13.0f);
        stroke_output(-2.0, 1.7, "Wel Come");
        stroke_output(-2.0, 0.9, "To");
        stroke_output(-2.0, 0.0, "Project Created");
        stroke_output(-2.0, -0.9, "By");
        stroke_output(-2.0, -1.8, "");
        glFlush();
        glutSwapBuffers();
}


void d5()
{
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glClearColor(1.0,1.0,1.0,1.0);
        glLoadIdentity();
        glTranslatef(0.0f,0.0f,-13.0f);
        stroke_output(-2.0, 1.7, "Wel Come");
        stroke_output(-2.0, 0.9, "To");
        stroke_output(-2.0, 0.0, "Project Created");
        stroke_output(-2.0, -0.9, "By");
        stroke_output(-2.0, -1.8, "");
        glFlush();
        glutSwapBuffers();
}

void d6()
{
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glClearColor(0.7,0.3,.2,1.0);
        glLoadIdentity();
        glTranslatef(0.0f,0.0f,-13.0f);
        stroke_output(-2.0, 1.7, "Wel Come");
        stroke_output(-2.0, 0.9, "To");
        stroke_output(-2.0, 0.0, "Project Created");
        stroke_output(-2.0, -0.9, "By");
        stroke_output(-2.0, -1.8, "");
        glFlush();
        glutSwapBuffers();
}

void d7()
{
```

```
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glClearColor(0.5,0.7,0.3,1.0);
        glLoadIdentity();
        glTranslatef(0.0f,0.0f,-13.0f);
        stroke_output(-2.0, 1.7, "Wel Come");
        stroke_output(-2.0, 0.9, "To");
        stroke_output(-2.0, 0.0, "Project Created");
        stroke_output(-2.0, -0.9, "By");
        stroke_output(-2.0, -1.8, "");
        glFlush();
        glutSwapBuffers();
}



void flying(double ang)
{
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glLoadIdentity();
    glTranslatef(0.0f,-.5f,-13.0f);
    glRotatef(115,1.0f,0.0f,0.0f);

        //fan
        glPushMatrix();
        glRotatef(ang,0.0f,0.0f,1.0f);
        glScaled(2.9,0.2,0.1);
        glTranslatef(0.0,0.0,16.0);
        glRotatef(ang,0.0f,0.0f,1.0f);
        glutSolidSphere(0.7,20,60);
        glPopMatrix();

        //2nd blade
        glPushMatrix();
        glRotatef(ang,0.0f,0.0f,1.0f);
        glScaled(0.2,2.9,0.1);
        glTranslatef(0.0,0.0,16.0);
        glRotatef(ang,0.0f,0.0f,1.0f);
        glutSolidSphere(0.7,20,60);
        glPopMatrix();

        glPushMatrix();
        glutWireCone(4,3,80,120);
        glPopMatrix();

        glPushMatrix();
        glRotated(ang,0.0,1.0,0.0);
        glTranslatef(0.05,-3.0,0.0);
        glRotated(ang,0.0,1.0,0.0);
        glutSolidSphere(0.3,20,60);
        glPopMatrix();
        glFlush();
        glutSwapBuffers();
}


void fly()
{
        x += 5.30;
        flying(x);
}


void nofly()
{
        flying(0);
}


void doInit()
{

        /* Background and foreground color */
```

```
    glClearColor(0.0,0.0,0.0,0.0);
    glColor3f(.0,1.0,1.0);
    glViewport(0,0,640,480);

        /* Select the projection matrix and reset it then
     setup our view perspective */
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluPerspective(30.0f,(GLfloat)640/(GLfloat)480,0.1f,200.0f);
    /* Select the modelview matrix, which we alter with rotatef() */
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glClearDepth(2.0f);
    glEnable(GL_DEPTH_TEST);
    glDepthFunc(GL_LEQUAL);
}

void doDisplay()
{
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();
        glTranslatef(0.0f,0.0f,-13.0f);
        stroke_output(-2.0, 1.7, "Wel Come");
        stroke_output(-2.0, 0.9, "To");
        stroke_output(-2.0, 0.0, "Project Created");
        stroke_output(-2.0, -0.9, "By");
        stroke_output(-2.0, -1.8, "");

        GLfloat mat_ambient[]={0.0f,1.0f,2.0f,1.0f};
        GLfloat mat_diffuse[]={0.0f,1.5f,.5f,1.0f};
        GLfloat mat_specular[]={5.0f,1.0f,1.0f,1.0f};
        GLfloat mat_shininess[]={50.0f};
        glMaterialfv(GL_FRONT,GL_AMBIENT,mat_ambient);
        glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
        glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
        glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);

        /*GLfloat lightIntensity[]={3.7f,0.7f,0.7f,1.0f};     Orange
        GLfloat light_position[]={2.0f,5.0f,3.0f,1.0f};*/

        /*light source properties*/
        GLfloat lightIntensity[]={1.7f,1.7f,1.7f,1.0f};
        GLfloat light_position[]={2.0f,0.0f,0.0f,0.0f};
        glLightfv(GL_LIGHT0,GL_POSITION,light_position);
        GLfloat light_position2[]={0.0f,0.0f,8.0f,0.0f};
        glLightfv(GL_LIGHT0,GL_POSITION,light_position2);
        GLfloat light_position3[]={0.0f,5.0f,2.0f,0.5f};
        glLightfv(GL_LIGHT0,GL_POSITION,light_position3);
        glLightfv(GL_LIGHT0,GL_DIFFUSE,lightIntensity);
        glFlush();
        glutSwapBuffers();

}

void menu(int id)
{
        switch(id)
        {

        case 1:glutIdleFunc(fly);
                break;
        case 2:exit(0);
                break;

        }
        glFlush();
        glutSwapBuffers();
        glutPostRedisplay();
}

void mykey(unsigned char key,int x,int y)
{
```

```
217
218         if(key=='q'||key=='Q')
219         {
220                 exit(0);
221         }
222         if(key=='1')
223         {
224                 glutIdleFunc(d4);
225         }
226         if(key=='2')
227         {
228                 glutIdleFunc(d5);
229         }
230         if(key=='3')
231         {
232                 glutIdleFunc(d6);
233         }
234         if(key=='4')
235         {
236                 glutIdleFunc(d7);
237         }
238
239         if(key=='b')
240         {
241                 glutIdleFunc(fly);
242         }
243         if(key=='B')
244         {
245                 glutIdleFunc(nofly);
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 }
```