NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India

(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Report on Mini Project

"Festivals of India"

Course Code: 18CS607

Course Name: Computer Graphics Lab

Semester: 6 Section: D

Submitted To,

Mr. Pradeep Kanchan

Asst Prof Gd III, Dept of CSE, NMAMIT

Submitted By:

Name: Sushmitha M Achar USN: 4NM18CS198
Name: Swathi Mithanthaya USN: 4NM18CS200
Name: U Raksha Shenoy USN: 4NM18CS204

Date of submission:

Signature of Course Instructor

Table of Content

Abstract

The project is a Computer Graphics mini project which is a simulation of 5 ‘Festivals of India’, namely, Christmas, Kumbha Mela, Guru Poornima, Makar Sankranti and Diwali. C programming in OpenGL in Codeblocks has been used for implementation purposes. Various function of Glut header files has been used to create different components and provide them the necessary dynamic movements. Every festival is provided with a description window explaining the importance of each festival. The simulation of Christmas involves trees, snowman and human beings with snowman and humans being in motion. The simulation of Kumbha Mela involves temple, humans, boats with boats and people being in motion. Furthermore, Guru Poornima involves simulation of statue, house and people. The Makar Sankranti simulation involves houses, people, kites, sweet pot with kites being in motion. Diwali consists of simulation of greetings and 3 crackers: namely rocket, flower pot, chakra.

Introduction

"Festivals of India" Computer Graphics project aims to promote the diversity of India and its culture. This project briefs about the reasons of celebrating some festivals in India and simulations for visualization purposes. It is made with a purpose of spreading awareness about the importance of celebrating some festivals in India. OpenGL software has been used to create the visualizations and it has been done using C programming language in Codeblocks.

The simulation uses movements of certain elements, audio, colors and images as added features. Many features of the project are user interactive i.e., it allows user to control the movements, the duration of visibility and the animations of every festival. Every festival covers the main events of a festival. Following festivals are implemented: Christmas, Kumbha Mela, Guru Poornima, Makar Sankranti and Diwali.

There were several challenges involved in the project. Firstly, to decide on the festivals to animate was difficult as there are many festivals celebrated in India. This was overcome by in-depth research and team discussion, Secondly, the structure and the flow of presenting the festivals was challenging as it should not look cluttered and confusing. This was solved by team discussion and planning. Furthermore, linking every festival to its description window and to other festivals required a lot of thinking and planning. To resolve we tried various methods like mouse functions, keyboard functions and other functions or methods and finally used the best method.

Implementation Details

a. Introduction Pages

/*Display project team details*/

```
void display0(){
   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(1.25,-0.85,0.0);
   glVertex3f(1.25,-0.95,0.0);
   glVertex3f(1.55,-0.95,0.0);
   glVertex3f(1.55,-0.85,0.0);
   glEnd();
   glPushMatrix();
   glTranslatef(1.31,-0.92,0.0);
   bitmap_output(0,0,0,"NEXT",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.6,0.8,0);
   bitmap_output(0,0,0,"COMPUTER GRAPHICS MINI PROJECT
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.5,0.6,0);
   bitmap_output(0,0,0,"FESTIVALS OF INDIA",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-1.0,0.5,0);
bitmap_output(0,0,0,"************************************************************",
GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
```

```
glTranslatef(-0.7,0.1,0);
bitmap_output(0,0,0,"SUBMITTED BY:",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glPushMatrix();
glTranslatef(-0.7,0.08,0);
bitmap_output(0,0,0,"_________________",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.1,0);
bitmap_output(0,0,0,"Swathi Mithanthaya",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.2,0);
bitmap_output(0,0,0,"U Raksha Shenoy",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(0.2,-0.1,0);
bitmap_output(0,0,0,"4NM18CS200",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(0.2,-0.2,0);
bitmap_output(0,0,0,"4NM18CS204",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.3,0);
bitmap_output(0,0,0,"Sushmitha M Achar",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(0.2,-0.3,0);
bitmap_output(0,0,0,"4NM18CS198",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.4,0);
bitmap_output(0,0,0,"UNDER THE GUIDANCE OF:",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.42,0);
bitmap_output(0,0,0,"_____________________________",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.6,0);
bitmap_output(0,0,0,"Mr.Pradeep Kanchan",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
glPushMatrix();
glTranslatef(-0.7,-0.7,0);
bitmap_output(0,0,0,"Assistant Professor Grade III",GLUT_BITMAP_TIMES_ROMAN_24);
glPopMatrix();
```

```
   glPushMatrix();
   glTranslatef(-0.7,-0.8,0);
   bitmap_output(0,0,0,"Department of CSE NMAMIT, Nitte",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPopMatrix();
   glFlush();
   glPopMatrix();
   glutPostRedisplay();}
```

/*Display the next screen to go to every festival*/

```
void display()
{   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
   glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.6,0.8,0);
   bitmap_output(0,0,0,"Press c for Christmas ",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.6,0.7,0);
   bitmap_output(0,0,0,"Press k for Kumbha Mela",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.6,0.6,0);
   bitmap_output(0,0,0,"Press g for Guru poornima",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.6,0.5,0);
   bitmap_output(0,0,0,"Press m for Makar Sankranthi",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glPushMatrix();
   glTranslatef(-0.6,0.4,0);
   bitmap_output(0,0,0,"Press d for Diwali",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glFlush();
   glPopMatrix(); glutPostRedisplay();}
```

Implementation Details
b. Christmas

```
/*Description window*/
void display1(){
   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
   glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.25,0.8,0);
   bitmap_output(0,0,0,"Merry Christmas",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(0.0,-0.85,0.0);
   glVertex3f(0.0,-0.95,0.0);
   glVertex3f(0.2,-0.95,0.0);
   glVertex3f(0.2,-0.85,0.0);
   glEnd();
   glPushMatrix();
   glTranslatef(0.0,-0.92,0.0);
   bitmap_output(0,0,0,"Click",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-1.0,0.6,0);
bitmap_output(0,0,0,"************************************************************",
GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.4,0);
   bitmap_output(0,0,0,"Christmas is celebrated on December 25 every year, and it is
a",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.3,0);
   bitmap_output(0,0,0,"cultural festival of the Christian community It marks the birth
",GLUT_BITMAP_TIMES_ROMAN_24);
```

```
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.2,0);
   bitmap_output(0,0,0,"of Jesus to Mother Mary It is said that Jesus was born in a
stable",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.1,0);
   bitmap_output(0,0,0,"and was a good shepherd. The life of Jesus was one of
hardships",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,0,0);
   bitmap_output(0,0,0,"and sacrifices, and Christmas aims to acknowledge that fact
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.1,0);
   bitmap_output(0,0,0,"Christmas is a public holiday in all the countries and is
celebrated",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.2,0);
   bitmap_output(0,0,0,"by non-Christians too as it is a festival that integrates people of
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.3,0);
   bitmap_output(0,0,0,"all cultures The central theme of Christmas celebrations is to
have",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.4,0);
   bitmap_output(0,0,0,"a spirit of sharing, kindness, and empathy towards one
another",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.6,0);
   bitmap_output(0,0,0,"Here is the simple simulation of christmas
festival,",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.7,0);
   bitmap_output(0,0,0,"Press on click to view the festival:",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glFlush();
   glPopMatrix();
   glutPostRedisplay();}
```

/*Christmas simulation*/

/*** Function to Draw Snow ******/**

```
void drawSnow(){
   glColor3d(0.95, 0.95, 0.95); // Color of the snow
   srand(time(NULL));
   int i;
   for(i=0; i < array_size; i++){
      glPushMatrix();
         glTranslated(snow[i][0], snow[i][1], -4);
         glRotated(65, -1.0, 0.0, 0.8);
         glutSolidCube(0.015);
      glPopMatrix();
      glPushMatrix();
         glTranslated(-snow[i][0], snow[i][1], -4);
         glRotated(65, -1.0, 0.0, 0.8);
         glutSolidCube(0.015);
      glPopMatrix();}
      for(j=0; j <array_size ; j++){
               snow[j][0] += 0.005;   // moving the snow in x direction
               snow[j][1] -= 0.005;}}   // moving the snow in y direction
```

/***** Funtion for Drawing lights on the tree ******/**

```
void drawLights(){
   // Eight Lights on the tree
   glColor3d(l1-0.01,l2,l3);
   glPushMatrix();
      glTranslated(-0.3,0.5, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l2,l3+0.06,l1);
   glPushMatrix();
      glTranslated(0.3, 0.95, -5);
      glRotated(65, -1.0, 0.0, 0.5);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l1, l3, l2-0.05);
   glPushMatrix();
      glTranslated(0, 1.6, -5);
      glRotated(65, -1.0, 0.0, 0.5);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l2-0.04, l1+0.02, l3);
```

```
   glPushMatrix();
      glTranslated(0.4, 0.25, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l3+0.02, l1-0.08, l2-0.06);
   glPushMatrix();
      glTranslated(-0.2,0.05, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l3-0.04, l2+0.07, l1-0.07);
   glPushMatrix();
      glTranslated(-0.5,1, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l3+0.06,l1,l1);
   glPushMatrix();
      glTranslated(-0.1,1.2, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();
   glColor3d(l2, l3+0.02, l3-0.04);
   glPushMatrix();
      glTranslated(0.2, 0.55, -5);
      glRotated(65, -1.0, 0.0, 0.8);
      glutSolidCube(0.1);
   glPopMatrix();}
```

/**** Function to Draw Snowman *******/**

```
void drawSnowMan(GLfloat x , GLfloat y , GLfloat z){

   glColor3f(0.0,0.0,0.0);
   glPushMatrix();       // Eyes
      glTranslatef(x,y+0.8,z+0.5);
      glutSolidSphere(0.05, 200, 200);
   glPopMatrix();
   glPushMatrix();
      glTranslatef(x +0.2,y+0.8,z + 0.5);
      glutSolidSphere(0.05, 200, 200);
   glPopMatrix();

   glColor3d(1, 0, 0);  // Nose
   glPushMatrix();
      glTranslatef(x+ 0.1,y + 0.65,z + 0.5);
      glRotatef(-260.0, 1.0, -1.5, 0.0);
```

```
      glutSolidCone(0.08, 0.2, 10, 2);
   glPopMatrix();

   glColor3f(1,1,1);  // Body Structure
   glPushMatrix();
glTranslatef(x, y + 0.1, z );
      glutSolidSphere(0.55, 200, 200);
   glPopMatrix();
   glColor3f(1,1,1);
   glPushMatrix();
      glTranslatef(x, y - 0.99, z);
      glutSolidSphere(0.95, 200, 200);
   glPopMatrix();
   glColor3f(1,1,1);
   glPushMatrix();
      glTranslatef(x, y+0.7, z);
      glutSolidSphere(0.35, 200, 200);
   glPopMatrix();}
```

/*** Function to draw the rotated snowman ******/**

```
void rotatingSnowMan(){
float    x = -2.7, y = 0.3  , z = -7;
   glPushMatrix();
   glTranslatef(x,y,z);
   glRotatef(angle,0.0,0.0,1.0);
   glTranslatef(-x,-y,-z);
   drawSnowMan(x,y,z);
   glPopMatrix();}
```

/**** Function to Draw the Tree ******/**

```
void drawTree(){
   glColor3d(0.1, 0.5, 0.1);  // Tree color
   //Tree Cones
   glPushMatrix();
      glTranslated(0.0, 1.1, -6);
      glRotated(65, -1.0, 0.0, 0.0);
      glutSolidCone(0.85, 1.6, 100, 100);
   glPopMatrix();
   glPushMatrix();
      glTranslated(0.0,0.6,-6);
      glRotated(65, -1.0, 0.0, 0.0);
      glutSolidCone(0.9, 1.5, 100, 100);
   glPopMatrix();
   glPushMatrix();
      glTranslated(0.0,0.25,-6);
      glRotated(65, -1.0, 0.0, 0.0);
```

```
      glutSolidCone(0.95, 1.5, 100, 100);
   glPopMatrix();
   // Tree base
   glColor3d(0.29, 0.13, 0.13);
   glPushMatrix();
      glTranslated(0.0,-1.15,-6);
      glRotated(65, -1.0, 0.0, 0.0);
      gluCylinder(quadratic, 0.3f, 0.3f, 1.25f, 32, 32);
   glPopMatrix();
   // Tree shadow
   glColor3d(0.6, 0.6, 0.6);
   glPushMatrix();
      glTranslated(0.0,-0.8,-6);
      glRotated(70, -1.0, 0.0, 0.0);
      glutSolidTorus(0.33, 0.60, 200, 200);
   glPopMatrix();}
```

/**** Function to Draw the ground/mountains ******/**

```
void drawGround()
{//Upper Ground Circle
   glColor3f(0.93,0.94,0.94);
   glPushMatrix();
      glTranslatef(-7,-10,-12.5);
      glutSolidSphere(10,200, 1000);
   glPopMatrix();
   //Middle Ground Circle
   glColor3f(0.96,0.97,0.95);
   glPushMatrix();
      glTranslatef(1.5,-10,-10.5);
      glutSolidSphere(10,200, 1000);
   glPopMatrix();
   //Back Ground Circle
   glColor3f(0.91,0.92,0.90);
   glPushMatrix();
      glTranslatef(13,-10,-19.5);
      glutSolidSphere(10,200, 1000);
   glPopMatrix();}
```

/*** Function to Draw Gifts *****/**

```
void drawGifts()
{//Gift One
glColor3f(1,0,0);
   glPushMatrix();
   glTranslatef(0,-0.62,-5);
   glRotatef(45,1,0,0);
   glRotatef(45,0,1,0);
```

```
   glutSolidCube(0.3);
   glPopMatrix();
   // Gift Two
   glColor3f(1,1,0);
   glPushMatrix();
   glTranslatef(0.5,-0.52,-5);
   glRotatef(45,1,0,0);
   glRotatef(45,0,1,0);
   glutSolidCube(0.3);
   glPopMatrix();
   // Gift Three
   glColor3f(0,1,0);
   glPushMatrix();
   glTranslatef(0.3,-1.00,-5);
   glRotatef(45,1,0,0);
   glRotatef(45,0,1,0);
   glutSolidCube(0.3);
   glPopMatrix();
   // Gift Four
   glColor3f(0,1,1);
   glPushMatrix();
   glTranslatef(-0.4,-0.9,-5);
   glRotatef(45,1,0,0);
   glRotatef(45,0,1,0);
   glutSolidCube(0.3);
   glPopMatrix();}
```

/*Function to draw people*/

```
void draw_people()
{ glPushMatrix();
   glLineWidth(2);
   glTranslatef(1.25, -0.2, -1.03);
   glBegin(GL_LINES);
   glVertex3f(0+shift, 0.0, 0+shift);
   glVertex3f(0+shift, -0.3, 0+shift);
   glVertex3f(0.0+shift, 0.0, 0.0+shift);
   glVertex3f(-0.09+shift,-0.15,0.0+shift);
   glVertex3f(0.0+shift, 0.0, 0.0+shift);
   glVertex3f(0.09+shift,-0.15,0.0+shift);
   glVertex3f(0.0+shift, -0.3, 0.0+shift);
   glVertex3f(-0.09+shift,-0.45,0.0+shift);
   glVertex3f(0.0+shift, -0.3, 0.0+shift);
   glVertex3f(0.09+shift,-0.45, 0.0+shift);
   glVertex3f(0.0+shift, 0.0, 0.0+shift);
   glVertex3f(0.08+shift, 0.07, 0.0+shift);
   glVertex3f(0.0+shift, 0.0, 0.0+shift);
   glVertex3f(-0.08+shift, 0.07, 0.0+shift);
```

```
   glVertex3f(0.08+shift, 0.07, 0.0+shift);
   glVertex3f(0.0+shift, 0.15, 0.0+shift);
   glVertex3f(-0.08+shift, 0.07, 0.0+shift);
   glVertex3f(0.0+shift, 0.15, 0.0+shift);
   glVertex3f(-0.023+shift, 0.095, 0.0+shift);
   glVertex3f(-0.03+shift, 0.095, 0.0+shift);
   glVertex3f(0.023+shift, 0.095, 0.0+shift);
   glVertex3f(0.03+shift, 0.095, 0.0+shift);
   glVertex3f(0.0+shift, 0.08, 0.0+shift);
   glVertex3f(0.0+shift, 0.06, 0.0+shift);
   glVertex3f(0.025+shift, 0.045, 0.0+shift);
   glVertex3f(-0.025+shift, 0.045, 0.0+shift);
   glEnd();
   glPopMatrix();}
```

/**** Display Function *******/**

```
void displayc(void){
   if(playThemeMusic1)
    { PlaySound(TEXT("themec.wav"), NULL, SND_ASYNC|SND_FILENAME|SND_LOOP);
                playThemeMusic1 = 0;}
glClearColor(0.45,0.45,1.0,1.0);
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
glLoadIdentity();
gluLookAt(0,0,1,0,0,0,0,1,0);
   drawGround();
   rotatingSnowMan();
   drawTree();
   drawSnow();
   drawLights();
   drawGifts();
   draw_people();}
```

/**** Idle Function which rotates the snowman ******/**

```
void idle(){
  if(right){
     angle += 0.9f ;
     if(angle > 7.0f)
  right = false;}
 if(!right){
angle -= 0.9f;
if(angle  < -4.0f) right = true;}
  srand(time(NULL));
  l1 = ((float) (rand()) / (float) (RAND_MAX));  // randomizing
  l2 = ((float) (rand()) / (float) (RAND_MAX))*0.99; // color of the
  l3 = ((float) (rand()) / (float) (RAND_MAX));      // lights
glutPostRedisplay();}
```

Implementation Details
c. Kumbha Mela

/*Decription Window*/

```
void display4(){
   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
    glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.25,0.8,0);
   bitmap_output(0,0,0,"Kumbha Mela",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(0.0,-0.85,0.0);
   glVertex3f(0.0,-0.95,0.0);
   glVertex3f(0.2,-0.95,0.0);
   glVertex3f(0.2,-0.85,0.0);
   glEnd();
   glPushMatrix();
   glTranslatef(0.0,-0.92,0.0);
   bitmap_output(0,0,0,"Click",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-1.0,0.6,0);
   bitmap_output(0,0,0,"********************",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.4,0);
   bitmap_output(0,0,0,"Kumbh Mela is one of the biggest and peaceful gatherings of
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
```

```
   glTranslatef(-0.8,0.3,0);
   bitmap_output(0,0,0,"Hindu pilgrims on the banks of holy rivers in India. ‘Ardh
Kumbh’",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.2,0);
   bitmap_output(0,0,0,"is held at every 6 year and ‘Kumbh’ is held at every 12
years",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.1,0);
   bitmap_output(0,0,0,"Lakhs of Hindu devotees take baths in the sacred water of river
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0,0);
   bitmap_output(0,0,0,"Yamuna, Ganga, Godavari and Shipra at four religious places in
a ",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.1,0);
   bitmap_output(0,0,0,"hope to attain salvation.The date and length of Kumbh Mela
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.2,0);
   bitmap_output(0,0,0,"celebration is decided by astronomy basis the zodiac locations
of  ",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.3,0);
   bitmap_output(0,0,0,"planets Jupiter, Sun and Moon  In 2019 Kumbh Mela
celebration",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.4,0);
   bitmap_output(0,0,0," was held for 55 days which began on 14th January and ended
on ",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.5,0);
   bitmap_output(0,0,0,"10th March.Maha Kumbh comes once in 144
years",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.7,0);
   bitmap_output(0,0,0,"Here is the simple simulation of Kumbh
mela,",GLUT_BITMAP_TIMES_ROMAN_24);
```

```
            glPopMatrix();
            glPushMatrix();
            glTranslatef(-0.8,-0.8,0);
            bitmap_output(0,0,0,"Press on click to view the
          festival:",GLUT_BITMAP_TIMES_ROMAN_24);
            glPopMatrix();
            glFlush();
            glPopMatrix();
            glutPostRedisplay();}
```

/*Kumbha Mela Simulation*/

/**** Function to draw sun ******/**

```
void drawSun()
{ glPushMatrix();
   glColor3f(3.0,0.5,0.0);
   DrawCircle(150,700,40,1000);
   glPopMatrix();}
```

/**** Function to draw tree ******/**

```
void tree(int x,int y)
{glPushMatrix();
glBegin(GL_POLYGON); //Tree
        glColor3f(0.6, 0.3, 0.0);
        glVertex2i(x+150, 450+y);
        glVertex2i(x+165, 450+y);
        glVertex2i(x+165, 550+y);
        glVertex2i(x+150, 550+y);
        glEnd();
        glColor3f(0.0, 0.7, 0.1);
        DrawCircle(x+130, 545+y, 40, 1000); // 4
        glColor3f(0.0, 0.7, 0.1);
        DrawCircle(x+155, 580+y, 40, 1000);
        glColor3f(0.0, 0.7, 0.1);
        DrawCircle(x+170, 550+y, 40, 1000);
        glPopMatrix();}
```

/****Function to draw background ******/**

```
void scene2()
{glPushMatrix();
        glBegin(GL_POLYGON); //Sky
        glColor3f(2.0, 0.7, 0.1);
        glVertex2i(0, 800);
```

```
                glVertex2i(1200, 800);
                glColor3f(0.7, 0.7, 1.0);
                glVertex2i(1200, 0);
                glVertex2i(0, 0);
                glEnd();
                glBegin(GL_POLYGON); //river
                glColor3f(0.2, 0.3, 1.1);
                glVertex2i(0, 200);
                glVertex2i(0, 430);
                glVertex2i(1300, 400);
                glVertex2i(1300, 50);
                glEnd();
                glBegin(GL_POLYGON); // ground
                glColor3f(0, 0.6, 0);
                glVertex2i(0, 200);
                glVertex2i(1200, 65);
                glVertex2i(1200,0);
                glVertex2i(0, 0);
                glEnd();
            glBegin(GL_POLYGON); // groundup
                glColor3f(0, 0.6, 0);
                glVertex2i(0, 430);
                glVertex2i(0, 500);
                glVertex2i(1200,500);
                glVertex2i(1200,400);
                glEnd();
                //tree();
                glPopMatrix();}
```

/**** Function to create people******/**

```
        void people(int x ,int y)
        {   glPushMatrix();
            glColor3f(0,0,0);
            DrawCircle(52+x,490+y,13,1000);
            glColor3f(0.8,0.7,0.5);
            DrawCircle(52+x,488+y,11,1000);
            glBegin(GL_LINES);
            glColor3f(0,0,0);
            glVertex2i(48+x,491+y);
            glVertex2i(48+x,494+y);
            glVertex2i(56+x,491+y);
            glVertex2i(56+x,494+y);
            glVertex2i(52+x,489+y);
            glVertex2i(52+x,485+y);
            glVertex2i(54+x,482+y);
            glVertex2i(50+x,482+y);
            glEnd();
```

```
glBegin(GL_POLYGON);
glColor3f(0,0,0.5);
glVertex2i(42+x,477+y);
glVertex2i(62+x,477+y);
glVertex2i(62+x,455+y);
glVertex2i(42+x,455+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.8,0.7,0.5);
glVertex2i(42+x,477+y);
glVertex2i(36+x,477+y);
glVertex2i(36+x,460+y);
glVertex2i(42+x,460+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.8,0.7,0.5);
glVertex2i(62+x,477+y);
glVertex2i(68+x,477+y);
glVertex2i(68+x,460+y);
glVertex2i(62+x,460+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.5,0,0.8);
glVertex2i(42+x,455+y);
glVertex2i(52+x,455+y);
glVertex2i(48+x,444+y);
glVertex2i(42+x,444+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.5,0,0.8);
glVertex2i(52+x,455+y);
glVertex2i(62+x,455+y);
glVertex2i(62+x,444+y);
glVertex2i(56+x,444+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.8,0.7,0.5);
glVertex2i(56+x,438+y);
glVertex2i(62+x,438+y);
glVertex2i(62+x,444+y);
glVertex2i(56+x,444+y);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.8,0.7,0.5);
glVertex2i(42+x,438+y);
glVertex2i(48+x,438+y);
glVertex2i(48+x,444+y);
glVertex2i(42+x,444+y);
```

```
        glEnd();
        glPopMatrix();}
```

/**** Function to draw boats with people ******/**

```
    void drawBoat(int x){
      glPushMatrix();
      glBegin(GL_POLYGON);
     glColor3f(0,0,0);//man1
      DrawCircle(52+x+shift,410,10,1000);
      glColor3f(0.8,0.7,0.5);
      DrawCircle(52+x+shift,409,8,1000);
      glColor3f(0,0,0);//man2
      DrawCircle(77+x+shift,410,10,1000);
      glColor3f(0.8,0.6,0.5);
      DrawCircle(77+x+shift,409,8,1000);
      glColor3f(0,0,0);//man3
      DrawCircle(102+x+shift,410,10,1000);
      glColor3f(0.8,0.8,0.5);
      DrawCircle(102+x+shift,409,8,1000);
      glColor3f(0,0,0);//man4
      DrawCircle(127+x+shift,410,10,1000);
      glColor3f(0.8,0.7,0.5);
      DrawCircle(127+x+shift,409,8,1000);
      glColor3f(0,0,0);//man5
      DrawCircle(152+x+shift,410,10,1000);
      glColor3f(0.8,0.6,0.5);
      DrawCircle(152+x+shift,409,8,1000);
            glBegin(GL_POLYGON);//boat down part
            glColor3f(0.4,0.0,0);
            glVertex2i(20+x+shift,380);
            glVertex2i(45+x+shift,350);
            glVertex2i(170+x+shift,350);
            glVertex2i(195+x+shift, 380);
            glVertex2i(170+x+shift,350);
            glVertex2i(45+x+shift,350);
            glVertex2i(20+x+shift,380);
            glEnd();
            glPushMatrix();
            glBegin(GL_POLYGON);//boat upper part
            glColor3f(0.7, 0.4, 0.0);
            glVertex2i(20+x+shift, 380);
            glVertex2i(45+x+shift, 400);
            glVertex2i(170+x+shift, 400);
            glVertex2i(195+x+shift, 380);
            glVertex2i(170+x+shift,360);
            glVertex2i(45+x+shift,360);
    glVertex2i(20+x+shift,380);
```

```
glEnd();
glBegin(GL_POLYGON);//stripe
glColor3f(0,0,0);
glVertex2i(55+x+shift,400);
glVertex2i(60+x+shift,400);
glVertex2i(60+x+shift,360);
glVertex2i(55+x+shift,360);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,1);
glVertex2i(47+x+shift,400);
glVertex2i(55+x+shift,400);
glVertex2i(55+x+shift,385);
glVertex2i(65+x+shift,385);
glVertex2i(65+x+shift,380);
glVertex2i(47+x+shift,380);
glVertex2i(47+x+shift,385);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2i(80+x+shift,400);
glVertex2i(85+x+shift,400);
glVertex2i(85+x+shift,360);
glVertex2i(80+x+shift,360);
glEnd();
glBegin(GL_POLYGON);//man2 body
glColor3f(1,0,1);
glVertex2i(72+x+shift,400);
glVertex2i(80+x+shift,400);
glVertex2i(80+x+shift,385);
glVertex2i(90+x+shift,385);
glVertex2i(90+x+shift,380);
glVertex2i(72+x+shift,380);
glVertex2i(72+x+shift,385);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2i(105+x+shift,400);
glVertex2i(110+x+shift,400);
glVertex2i(110+x+shift,360);
glVertex2i(105+x+shift,360);
glEnd();
glBegin(GL_POLYGON);//man3 body
glColor3f(1,0,0);
glVertex2i(97+x+shift,400);
glVertex2i(105+x+shift,400);
glVertex2i(105+x+shift,385);
glVertex2i(115+x+shift,385);
```

```
glVertex2i(115+x+shift,380);
glVertex2i(97+x+shift,380);
glVertex2i(97+x+shift,385);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2i(130+x+shift,400);
glVertex2i(135+x+shift,400);
glVertex2i(135+x+shift,360);
glVertex2i(130+x+shift,360);
glEnd();
glBegin(GL_POLYGON);//man4 body
glColor3f(0,1,1);
glVertex2i(122+x+shift,400);
glVertex2i(130+x+shift,400);
glVertex2i(130+x+shift,385);
glVertex2i(140+x+shift,385);
glVertex2i(140+x+shift,380);
glVertex2i(122+x+shift,380);
glVertex2i(122+x+shift,385);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2i(155+x+shift,400);
glVertex2i(160+x+shift,400);
glVertex2i(160+x+shift,360);
glVertex2i(155+x+shift,360);
glEnd();
glBegin(GL_POLYGON);//man5 body
glColor3f(0,1,0);
glVertex2i(147+x+shift,400);
glVertex2i(155+x+shift,400);
glVertex2i(155+x+shift,385);
glVertex2i(165+x+shift,385);
glVertex2i(165+x+shift,380);
glVertex2i(147+x+shift,380);
glVertex2i(147+x+shift,385);
glEnd();
glBegin(GL_LINES);
    glColor3f(0.0, 0.0, 0.0);
    glPointSize(2);
    glVertex2i(20+x+shift, 380);
    glVertex2i(45+x+shift, 400);
    glVertex2i(45+x+shift, 400);
    glVertex2i(170+x+shift, 400);
    glVertex2i(170+x+shift, 400);
    glVertex2i(195+x+shift, 380);
    glVertex2i(195+x+shift, 380);
```

```
                glVertex2i(170+x+shift,360);
                glVertex2i(170+x+shift,360);
                glVertex2i(45+x+shift,360);
                glVertex2i(45+x+shift,360);
                glVertex2i(20+x+shift,380);
                glVertex2i(50+x+shift,390);//stick1
                glVertex2i(65+x+shift,320);
                glVertex2i(75+x+shift,390);//stick2
                glVertex2i(90+x+shift,320);
                glVertex2i(100+x+shift,390);//stick3
                glVertex2i(115+x+shift,320);
                glVertex2i(125+x+shift,390);//stick4
                glVertex2i(140+x+shift,320);
                glVertex2i(150+x+shift,390);//stick5
                glVertex2i(165+x+shift,320);
                glEnd();
        glPopMatrix();}
```

/****Function to draw temple ******/**

```
        void temple(){
           glPushMatrix();
           glBegin(GL_POLYGON);
           glColor3f(0.5,0.5,0.5);
           glVertex2i(500,430);
           glVertex2i(500,530);
           glVertex2i(550,750);
           glVertex2i(650,750);
           glVertex2i(700,530);
           glVertex2i(700,430);
           glVertex2i(500,430);
           glEnd();
           glBegin(GL_LINES);
           glColor3f(0.7,0.6,1.5);
           glVertex2i(500,530);
           glVertex2i(700,530);
           glVertex2i(520,610);
           glVertex2i(680,610);
           glVertex2i(540,690);
           glVertex2i(660,690);
           glColor3f(0.7,0.2,0);
           glVertex2i(560,430);
           glVertex2i(530,410);
           glVertex2i(640,430);
           glVertex2i(670,410);
           glColor3f(0,0,0);
           glPointSize(5);
           glVertex2i(551,750);
```

```
glVertex2i(551,790);
glVertex2i(650,750);
glVertex2i(650,790);
glEnd();
glBegin(GL_POLYGON); //door1
glColor3f(1,1,0.5);
glVertex2i(560,430);
glVertex2i(560,520);
glVertex2i(640,520);
glVertex2i(640,430);
glEnd();
glBegin(GL_POLYGON); //door2
glColor3f(1,1,0.5);
glVertex2i(570,530);
glVertex2i(570,595);
glVertex2i(630,595);
glVertex2i(630,530);
glEnd();
glBegin(GL_POLYGON); //door3
glColor3f(1,1,0.5);
glVertex2i(580,610);
glVertex2i(580,670);
glVertex2i(620,670);
glVertex2i(620,610);
glEnd();
glBegin(GL_POLYGON); //door4
glColor3f(1,1,0.5);
glVertex2i(585,690);
glVertex2i(585,730);
glVertex2i(615,730);
glVertex2i(615,690);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1.0,0.2,0.2);
glVertex2i(550,760);
glVertex2i(550,790);
glVertex2i(530,770);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1.0,0.2,0.2);
glVertex2i(650,760);
glVertex2i(650,790);
glVertex2i(670,770);
glEnd();
glPopMatrix();}
```

/****Function to display ******/**

```
void display7(void)
{
      if(playThemeMusic4)
    {
                PlaySound(TEXT("themek.wav"), NULL,
SND_ASYNC|SND_FILENAME|SND_LOOP);
                playThemeMusic4 = 0;
    }

   glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
   glClearColor(0.0,0.498039,1.0,0.0);
   glMatrixMode(GL_PROJECTION);
         scene2();
         drawSun();
         tree(0,0);
   people(0+shift,0);
   drawBoat(50);
         glFlush();
         glutPostRedisplay();
         glutSwapBuffers();
}
```

Implementation Details
d. Guru Poornima

/*Description Window*/

```
void display5(){
   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
    glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.25,0.8,0);
   bitmap_output(0,0,0,"Guru Poornima",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(0.0,-0.85,0.0);
   glVertex3f(0.0,-0.95,0.0);
   glVertex3f(0.2,-0.95,0.0);
   glVertex3f(0.2,-0.85,0.0);
   glEnd();
   glPushMatrix();
   glTranslatef(0.0,-0.92,0.0);
   bitmap_output(0,0,0,"Click",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-1.0,0.6,0);
bitmap_output(0,0,0,"**************************************************
****",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.4,0);
   bitmap_output(0,0,0,"'Guru Purnima' is a famous festival of Hindus. It is celebrated
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
```

```
   glTranslatef(-0.8,0.3,0);
   bitmap_output(0,0,0,"on the full moon day in the month of Ashadh according
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.2,0);
   bitmap_output(0,0,0,"to Hindu Calendar.Guru Purnima is celebrated in the sacred
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.1,0);
   bitmap_output(0,0,0,"memory of the great sage Vyasa, the ancient saint the Srimad
who",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0,0);
   bitmap_output(0,0,0,"compiled the four Vedas, wrote 18 Puranas,the Mahabharata and
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.1,0);
   bitmap_output(0,0,0,"Bhagavata. The day is also known as 'Vyasa Purnima' This day
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.2,0);
   bitmap_output(0,0,0,"is celebrated as a mark of respect to the 'Guru' i.e a
teacher",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.3,0);
   bitmap_output(0,0,0,"The day is observed by devotees who offer pujas(worship)
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.4,0);
   bitmap_output(0,0,0,"to their beloved Gurus.Guru Purnima is celebrated to honour
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.5,0);
   bitmap_output(0,0,0," our teachers, who remove the darkness from our
minds",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.7,0);
   bitmap_output(0,0,0,"Here is the simple simulation of Guru
Poornima,",GLUT_BITMAP_TIMES_ROMAN_24);
```

```
    glPopMatrix();
    glPushMatrix();
    glTranslatef(-0.8,-0.8,0);
    bitmap_output(0,0,0,"Press on click to view the
festival:",GLUT_BITMAP_TIMES_ROMAN_24);
    glPopMatrix();
    glFlush();
    glPopMatrix();
    glutPostRedisplay();}
```

/*Guru Poornima simulation*/

/**** Function to draw people******/**

```
void peoplex(){
    glPushMatrix();
    glColor3f(0,0,0);
    DrawCircle1(52,190,13,1000);
    glColor3f(0.8,0.7,0.5);
    DrawCircle1(52,188,11,1000);
    glBegin(GL_LINES);
    glColor3f(0,0,0);
    glVertex2i(48,191);
    glVertex2i(48,194);
    glVertex2i(56,191);
    glVertex2i(56,194);
    glVertex2i(52,189);
    glVertex2i(52,185);
    glVertex2i(54,182);
    glVertex2i(50,182);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(0,0,0.5);
    glVertex2i(42,177);
    glVertex2i(62,177);
    glVertex2i(62,155);
    glVertex2i(42,155);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(0.8,0.7,0.5);
    glVertex2i(42,177);
    glVertex2i(36,177);
    glVertex2i(36,160);
    glVertex2i(42,160);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(0.8,0.7,0.5);
```

```
            glVertex2i(62,177);
            glVertex2i(68,177);
            glVertex2i(68,160);
            glVertex2i(62,160);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(0.5,0,0.8);
            glVertex2i(42,155);
            glVertex2i(52,155);
            glVertex2i(48,144);
            glVertex2i(42,144);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(0.5,0,0.8);
            glVertex2i(52,155);
            glVertex2i(62,155);
            glVertex2i(62,144);
            glVertex2i(56,144);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(0.8,0.7,0.5);
            glVertex2i(56,138);
            glVertex2i(62,138);
            glVertex2i(62,144);
            glVertex2i(56,144);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(0.8,0.7,0.5);
            glVertex2i(42,138);
            glVertex2i(48,138);
            glVertex2i(48,144);
            glVertex2i(42,144);
            glEnd();
            glPopMatrix();}

/******Function to display ******/

        void display8()
        { if(playThemeMusic3)
            {PlaySound(TEXT("themeg.wav"), NULL, SND_ASYNC|SND_FILENAME|SND_LOOP);
                        playThemeMusic3= 0; }
        glClear(GL_COLOR_BUFFER_BIT);
        sky();
        back();
        grass();
        tree();
        idle2();
        flower();
```

```
        man();
        sea();
        house1();
        peoplex();
        glRasterPos2f(250,50);
        glColor3f(1,0,0);
        glEnd();
        glFlush();}

/****** Function to draw sky ******/

        void sky(){
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(0,700,0,700);
        glBegin(GL_POLYGON);
        glColor3f(0.1,0.0,0.8);
        glVertex2f(0,400);
        glVertex2f(700,400);
        glColor3f(0.7,0.7,1);
        glVertex2f(700,700);
        glVertex2f(0,700);
        glVertex2f(0,400);
        glEnd();}

/****** Function to draw sea ******/

        void sea(){
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(0,700,0,700);
        glBegin(GL_POLYGON);
        glColor3f(0.5,1,0);
        glVertex2f(700,400);
        glColor3f(0,0.5,0);
        glVertex2f(600,350);
        glColor3f(0,0.5,0);
        glVertex2f(550,300);
        glVertex2f(500,250);
        glVertex2f(450,225);
        glVertex2f(350,0);
        glColor3f(0.0,0.0,0);
        glVertex2f(700,0);
        glVertex2f(700,400);
        glEnd();
        }

/****** Function to draw house ******/
```

```
void house1()
{glBegin(GL_POLYGON);
glColor3f(0.9,0.9,0.9);
glVertex2f(400+50,330-50);
glVertex2f(400+50,150-50);
glVertex2f(650+50,150-50);
glVertex2f(650+50,330-50);
glVertex2f(400+50,330-50);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.69,0.133,0.123);
glVertex2f(500+50,150-50);
glVertex2f(500+50,250-50);
glVertex2f(550+50,250-50);
glVertex2f(550+50,150-50);
glVertex2f(500+50,150-50);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.3,0.3,0.3);
glVertex2f(425+50,225-50);
glVertex2f(475+50,225-50);
glVertex2f(475+50,250-50);
glColor3f(0,0,0);
glVertex2f(425+50,250-50);
glVertex2f(425+50,225-50);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.3,0.3,0.3);
glVertex2f(575+50,225-50);
glVertex2f(625+50,225-50);
glVertex2f(625+50,250-50);
glColor3f(0,0,0);
glVertex2f(575+50,250-50);
glVertex2f(575+50,225-50);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.2,0.1,0);
glVertex2f(525+50,330-50);
glVertex2f(475+50,290-50);
glVertex2f(350+50,290-50);
glColor3f(0,0,0);
glVertex2f(525+50,475-50);
glColor3f(0.6,0.3,0.3);
glVertex2f(700+50,290-50);
glVertex2f(575+50,290-50);
glVertex2f(525+50,330-50);
glEnd();}
```

/**** Idle Function to draw the statue ******/**

```
void idle2()
{glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(30,375,-50,375);//body
        glBegin(GL_POLYGON);
                glColor3f(0.3,0.3,0.3);
                glVertex2f(171,155);
                glVertex2f(175,145);
                glVertex2f(178,144);
                glVertex2f(181,143);
                glVertex2f(180,130);
                glVertex2f(179.5,125);
                glVertex2f(179,120);
                glVertex2f(179,115);
                glVertex2f(179,110);
                glVertex2f(179.8,105);
                glVertex2f(180,102);
                glVertex2f(180,100);
                glVertex2f(154.8,100);
                glVertex2f(154.8,102);
                glVertex2f(155,105);
                glVertex2f(155.2,110);
                glVertex2f(155,115);
                glVertex2f(154.5,120);
                glVertex2f(154.3,125);
                glVertex2f(154,128);
                glVertex2f(153.5,129);
                glVertex2f(152.5,130);
                glVertex2f(153,142);
                glVertex2f(155,143);
                glVertex2f(158,144);
                glVertex2f(160,145);
                glVertex2f(160,145);
                glVertex2f(161.5,148);
                glVertex2f(161.5,148);
                glVertex2f(162,150);
                glVertex2f(162,150);
                glVertex2f(162,155);
                glVertex2f(162,155);
                glVertex2f(171,155);
        glEnd();
        glBegin(GL_LINES);
                glVertex2f(180,102);
                glVertex2f(154.8,102);
```

```
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2f(177+3,103-3);
glColor3f(0.3,0.3,0.3);
glVertex2f(190+3,103-3);
glVertex2f(190+3,70-3);
glVertex2f(135+3,70-3);
glVertex2f(135+3,103-3);
glVertex2f(177+3,103-3);
glEnd();
glBegin(GL_LINE_STRIP);
glColor3f(0,0,0);
glVertex2f(162,150);
glVertex2f(165,130);
glVertex2f(168,120);
glVertex2f(171,130);
glVertex2f(174,150);
glEnd();//head
glBegin(GL_POLYGON);
                glColor3f(0.3,0.3,0.3);
                glVertex2f(167,152);
                glVertex2f(169,153);
                glVertex2f(172,155);
                glVertex2f(173.5,158);
                glVertex2f(174,160);
                glVertex2f(174.5,164);
                glVertex2f(174,169);
                glVertex2f(173,172);
                glVertex2f(171,174);
                glVertex2f(169,174.8);
                glVertex2f(167,175);
                glVertex2f(164,174.5);
                glVertex2f(162,173.5);
                glVertex2f(160,171);
                glVertex2f(159,169);
                glVertex2f(159,165);
                glVertex2f(159,160);
                glVertex2f(159.5,158);
                glVertex2f(160,157);
                glVertex2f(161,156);
                glVertex2f(162.5,154.5);
                glVertex2f(163,154);
                glVertex2f(164,153);
                glVertex2f(167,152);
        glEnd();//nose
glBegin(GL_LINES);
        glColor3f(0.0,0.0,0.0);
```

```
            glVertex2f(166.9,165);
            glVertex2f(166.9,161);
        glEnd();//mouth
        glBegin(GL_LINE_STRIP);
            glColor3f(0.0,0.0,0.0);
            glVertex2f(165,159);
            glVertex2f(166,158.7);
            glVertex2f(167,158.5);
            glVertex2f(168,158.6);
            glVertex2f(169.8,159);
        glEnd();//hair
        glBegin(GL_POLYGON);
                    glColor3f(0.3,0.3,0.3);
                    glVertex2f(174.5,164);
                    glVertex2f(174,169);
                    glVertex2f(174,169);
                    glVertex2f(173,172);
                    glVertex2f(173,170);
                    glVertex2f(173,170);
                    glVertex2f(174,167);
                    glVertex2f(174,167);
                    glVertex2f(174,167);
                    glVertex2f(174.5,162);
                    glVertex2f(174.5,162);
                    glVertex2f(174.5,164);
                    glEnd();
        glBegin(GL_POLYGON);
                     glVertex2f(174,169);
                     glVertex2f(173,172);
                     glVertex2f(171,174);
                    glVertex2f(171,174);
                    glVertex2f(169,174.8);
                    glVertex2f(169,174.8);
                    glVertex2f(167,175);
                    glVertex2f(167,173);
                    glVertex2f(167,173);
                    glVertex2f(169,172.8);
                    glVertex2f(169,172.8);
                    glVertex2f(171,172);
                    glVertex2f(171,172);
                    glVertex2f(173,170);
                    glVertex2f(174,167);
        glEnd();
        glBegin(GL_POLYGON);
                    glVertex2f(167,175);
                    glVertex2f(164,174.5);
                    glVertex2f(164,174.5);
                    glVertex2f(162,173.5);
```

```
                    glVertex2f(162,173.5);
                    glVertex2f(160,171);
                    glVertex2f(160,171);
                    glVertex2f(159,169);
                    glVertex2f(159,169);
                    glVertex2f(159,167);
                    glVertex2f(159,167);
                    glVertex2f(160,169);
                    glVertex2f(160,169);
                    glVertex2f(162,171.5);
                    glVertex2f(162,171.5);
                    glVertex2f(164,172.5);
                    glVertex2f(164,172.5);
                   glVertex2f(167,173);
    glEnd();
    glBegin(GL_POLYGON);
    glVertex2f(160,158);
    glVertex2f(160.8,157);
    glVertex2f(160.8,154);
    glVertex2f(160.8,150);
    glVertex2f(160.8,145);
    glVertex2f(160.8,143);
    glVertex2f(160.8,136);
    glEnd();
    glBegin(GL_POLYGON);
    glVertex2f(161,157);
    glVertex2f(161.8,156);
    glVertex2f(161.8,154);
    glVertex2f(161.8,145);
    glVertex2f(161.8,143);
    glVertex2f(161.8,140);
    glVertex2f(161.8,135);
    glEnd();
    glBegin(GL_POLYGON);
    glVertex2f(172,156.5);
    glVertex2f(172.8,156);
    glVertex2f(172.8,154);
    glVertex2f(172.8,145);
    glVertex2f(172.8,143);
    glVertex2f(172.8,140);
    glVertex2f(172.8,135);
    glEnd();
    glBegin(GL_POLYGON);
    glVertex2f(173,158.5);
    glVertex2f(173.8,157);
    glVertex2f(173.8,154);
    glVertex2f(173.8,150);
    glVertex2f(173.8,145);
```

```
glVertex2f(173.8,143);
glVertex2f(173.8,136);
glEnd();//EYES
glBegin(GL_POLYGON);
        glColor3f(0.0,0.0,0.0);
                glVertex2f(160,165);
                glVertex2f(162,163.5);
                glVertex2f(163,163.5);
                glVertex2f(165,165);
                glVertex2f(163,166.5);
                glVertex2f(162,166.5);
        glEnd();
        glBegin(GL_POLYGON);
                glVertex2f(168,165);
                glVertex2f(170,163.5);
                glVertex2f(171,163.5);
                glVertex2f(173,165);
                glVertex2f(171,166.5);
                glVertex2f(170,166.5);
        glEnd();
glBegin(GL_LINE_STRIP);
                glVertex2f(160,166.5);
                glVertex2f(162,168.5);
                glVertex2f(163,168.5);
                glVertex2f(165,166.5);
        glEnd();
glBegin(GL_LINE_STRIP);
                glVertex2f(168,166.5);
                glVertex2f(171,168.5);
                glVertex2f(170,168.5);
                glVertex2f(173,166.5);
        glEnd();
        glColor3f(1.0,1.0,1.0);
                glColor3f(0.0,0.0,0.0);
        glBegin(GL_POLYGON);//hand1
                glColor3f(0.3,0.3,0.3);
                glVertex2f(153,142);
                glVertex2f(151,140.5);
                glVertex2f(150,139);
                glVertex2f(149,137);
                glVertex2f(148,135);
                glVertex2f(147.2,133);
                glVertex2f(147,130);
                glVertex2f(147,125);
                glVertex2f(147.1,120);
                glVertex2f(147.2,112);//HALF
                glVertex2f(151.8,105);
                glVertex2f(152,110);
```

```
                glVertex2f(152.5,115);
                glVertex2f(152.5,120);
                glVertex2f(152.5,123);
                glVertex2f(152.5,125);
                glVertex2f(152.5,130);
        glEnd();
    glBegin(GL_POLYGON);
        glColor3f(0.3,0.3,0.3);
        glVertex2f(147.2,112);
        glVertex2f(143.2,125);
        glVertex2f(139.2,125);
        glVertex2f(146.2,101);
        //glVertex2(87.2,100);
        glVertex2f(148.2,100.5);
        glVertex2f(149.2,101);
        glVertex2f(150.2,101.5);
        glVertex2f(151.2,102);
        glVertex2f(151.8,105);
    glEnd();
                glBegin(GL_POLYGON);
                glColor3f(0.3,0.3,0.3);
                glVertex2f(139.2,125);
                glVertex2f(137.6,129);
                glVertex2f(137.6,136);
                glVertex2f(139.2,138);
                glVertex2f(140.2,139);
                glVertex2f(141.2,139.1);
                glVertex2f(142.2,139.2);
                glVertex2f(143.2,139.3);
                glVertex2f(144.2,138);
                glVertex2f(144.2,137);
                glVertex2f(144.2,136);
                glVertex2f(144.2,135);
                glVertex2f(144,134);
                glVertex2f(143.8,133);
                glVertex2f(143.6,132);
                glVertex2f(143.6,130);
                glVertex2f(144.2,131);
                glVertex2f(144.3,132);
                glVertex2f(144.4,133);
                glVertex2f(144.8,134);
                glVertex2f(144.9,135);
                glVertex2f(145,136);
                glVertex2f(145.1,136);
                glVertex2f(145.3,136);
                glVertex2f(145.8,136);
                glVertex2f(146.2,136);
                glVertex2f(146.2,135);
```

```
                glVertex2f(146.2,133);
                glVertex2f(146,134);
                glVertex2f(145.8,132);
                glVertex2f(145.4,130);
                glVertex2f(145,128);
                glVertex2f(144.6,126);
                glVertex2f(143.6,124.2);
                glEnd();//rays
        if(s2<170)
        s2+=0.45;
if(s3<10)
s3+=0.045;
        glBegin(GL_POLYGON);
        glColor3f(1.0,1.0,0.0);
        glVertex2f(141.2,137);
        glColor3f(1.0,1.0,0.0);
        glVertex2f(141.2,127);
        glColor3f(1.0,0.0,0.0);
        glVertex2f(141-s2,100-s3);
        glColor3f(1.0,1.0,0.0);
        glVertex2f(141-s2,99.5-s3);
        glutPostRedisplay();
        glEnd();
        glBegin(GL_POLYGON);/*hand2*/
                glColor3f(0.3,0.3,0.3);
                glVertex2f(180,143);//
                glVertex2f(182,142);
                glVertex2f(182,142);
                glVertex2f(183,141);
                glVertex2f(183,141);
                glVertex2f(184,139);
                glVertex2f(184,139);
                glVertex2f(185,136.5);
                glVertex2f(185,136.5);
                glVertex2f(186,135);
                glVertex2f(186,135);
                glVertex2f(187,130);
                glVertex2f(187,130);
                glVertex2f(187.2,125);
                glVertex2f(187.2,125);
                glVertex2f(187.5,120);
                glVertex2f(187.5,120);
                glVertex2f(187.5,117);
                glVertex2f(187.5,117);
                glVertex2f(187.2,115);
                glVertex2f(187.2,115);
                glVertex2f(187.2,112);
                glVertex2f(187.2,112);
```

```
                glVertex2f(187.2,108);
                glVertex2f(187.2,108);
                glVertex2f(184,103);
                glVertex2f(184,103);
                glVertex2f(183.5,105);
                glVertex2f(183.5,105);
                glVertex2f(183,110);
                glVertex2f(183,110);
                glVertex2f(182.5,115);
                glVertex2f(182.5,115);
                glVertex2f(182,120);
                glVertex2f(182,120);
                glVertex2f(180.5,125);//121
                glVertex2f(180,125);
                glVertex2f(179,143);//K
        glEnd();
        glBegin(GL_POLYGON);
                glVertex2f(200,120);
                glVertex2f(203,125);
                glVertex2f(204,125);
                glVertex2f(207,121);
                glVertex2f(207,115);
                glVertex2f(204,112);
                glVertex2f(203,112);
                glVertex2f(200,113);
        glEnd();
                glBegin(GL_POLYGON);
                glVertex2f(187.2,108);
                 glVertex2f(200,120);
                glVertex2f(200,120);
                glVertex2f(200,113);
                glVertex2f(200,113);
                glVertex2f(187,98);
                glVertex2f(187,98);
                glVertex2f(186,98.5);
                glVertex2f(186,98.5);
                glVertex2f(185,99);
                glVertex2f(185,99);
                glVertex2f(184,103);
                glEnd();
                glBegin(GL_POLYGON);/*pony*/
                glColor3f(0.3,0.3,0.3);
                glVertex2f(164,185.6);
                glVertex2f(169,185.6);
                glVertex2f(170,182.8);
                glVertex2f(170,178.8);
                glVertex2f(169,174.8);
                glVertex2f(164,174.8);
```

```
                glVertex2f(163,178.8);
                glVertex2f(163,182.8);
                glEnd();}

/****** Function to draw flowers ******/

        void flower()
        {       glMatrixMode(GL_PROJECTION);
                glLoadIdentity();
                gluOrtho2D(30,375,-50,375);
                f+=0.4;
                if(f<150){
                glBegin(GL_POLYGON);/*flowers*/
                        glColor3f(1,0,0);
                        glVertex2f(150,190-f);
                        glVertex2f(153,190-f);
                        glVertex2f(153,186-f);
                        glVertex2f(150,186-f);
                        glVertex2f(150,190-f);
                        glEnd();
                glBegin(GL_POLYGON);
                glColor3f(1,1,0);
                        glVertex2f(150,190-f);
                        glVertex2f(149,191-f);
                        glVertex2f(152,193-f);
                        glVertex2f(154,191-f);
                        glVertex2f(153,190-f);
                        glVertex2f(150,190-f);
                        glEnd();
        glBegin(GL_POLYGON);
                glColor3f(1,1,0);
                        glVertex2f(153,190-f);
                        glVertex2f(154,191-f);
                        glVertex2f(155,188-f);
                        glVertex2f(154,185-f);
                        glVertex2f(153,186-f);
                        glVertex2f(153,190-f);
                        glEnd();
                glBegin(GL_POLYGON);/*flowers*/
                        glColor3f(1,1,0);
                        glVertex2f(160,200-f);
                        glVertex2f(163,200-f);
                        glVertex2f(163,196-f);
                        glVertex2f(160,196-f);
                        glVertex2f(160,200-f);
                        glEnd();
                        glBegin(GL_POLYGON);
                        glColor3f(1,0,0);
```

```
                glVertex2f(160,200-f);
                glVertex2f(159,201-f);
                glVertex2f(162,203-f);
                glVertex2f(164,201-f);
                glVertex2f(163,200-f);
                glVertex2f(160,200-f);
                glEnd();
        glBegin(GL_POLYGON);
                glColor3f(1,0,0);
                glVertex2f(163,200-f);
                glVertex2f(164,201-f);
                glVertex2f(165,198-f);
                glVertex2f(164,195-f);
                glVertex2f(163,196-f);
                glVertex2f(163,200-f);
                glEnd();
        glBegin(GL_POLYGON);/*flowers*/
                glColor3f(1,0,0);
                glVertex2f(170,190-f);
                glVertex2f(169,191-f);
                glVertex2f(172,193-f);
                glVertex2f(174,191-f);
                glVertex2f(173,190-f);
                glVertex2f(170,190-f);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(0,0,1);
                glVertex2f(170,190-f);
                glVertex2f(173,190-f);
                glVertex2f(173,186-f);
                glVertex2f(170,186-f);
                glVertex2f(170,190-f);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(1,0,0);
                glVertex2f(173,190-f);
                glVertex2f(174,191-f);
                glVertex2f(175,188-f);
                glVertex2f(174,185-f);
                glVertex2f(173,186-f);
                glVertex2f(173,190-f);
                glEnd();
        glBegin(GL_POLYGON);/*flowers*/
                glColor3f(0,1,1);
                glVertex2f(180,200-f);
                glVertex2f(183,200-f);
                glVertex2f(183,196-f);
                glVertex2f(180,196-f);
```

```
glVertex2f(180,200-f);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,1,0);
glVertex2f(180,200-f);
glVertex2f(179,201-f);
glVertex2f(182,203-f);
glVertex2f(184,201-f);
glVertex2f(183,200-f);
glVertex2f(180,200-f);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0,1,0);
glVertex2f(183,200-f);
glVertex2f(184,201-f);
glVertex2f(185,198-f);
glVertex2f(184,195-f);
glVertex2f(183,196-f);
glVertex2f(183,200-f);
glEnd();
glBegin(GL_POLYGON);/*flowers*/
glColor3f(1,0,1);
glVertex2f(170,210-f);
glVertex2f(173,210-f);
glVertex2f(173,206-f);
glVertex2f(170,206-f);
glVertex2f(170,210-f);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1,0,0);
glVertex2f(170,210-f);
glVertex2f(169,211-f);
glVertex2f(172,213-f);
glVertex2f(174,211-f);
glVertex2f(173,210-f);
glVertex2f(170,210-f);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1,0,0);
glVertex2f(173,210-f);
glVertex2f(174,211-f);
glVertex2f(175,208-f);
glVertex2f(174,205-f);
glVertex2f(173,206-f);
glVertex2f(173,210-f);
glEnd();
glBegin(GL_POLYGON);/*flowers*/
glColor3f(1,0,0);
```

```
                glVertex2f(190,190-f);
                glVertex2f(193,190-f);
                glVertex2f(193,186-f);
                glVertex2f(190,186-f);
                glVertex2f(190,190-f);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(1,1,0);
                glVertex2f(190,190-f);
                glVertex2f(189,191-f);
                glVertex2f(192,193-f);
                glVertex2f(194,191-f);
                glVertex2f(193,190-f);
                glVertex2f(190,190-f);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(1,1,0);
                glVertex2f(193,190-f);
                glVertex2f(194,191-f);
                glVertex2f(195,188-f);
                glVertex2f(194,185-f);
                glVertex2f(193,186-f);
                glVertex2f(193,190-f);
                glEnd();}
        else
        {       glBegin(GL_POLYGON);/*flowers*/
                glColor3f(1,0,0);
                glVertex2f(150,190-140);
                glVertex2f(153,190-140);
                glVertex2f(153,186-140);
                glVertex2f(150,186-140);
                glVertex2f(150,190-140);
                glEnd();
                glBegin(GL_POLYGON);
        glColor3f(1,1,0);
                glVertex2f(150,190-140);
                glVertex2f(149,191-140);
                glVertex2f(152,193-140);
                glVertex2f(154,191-140);
                glVertex2f(153,190-140);
                glVertex2f(150,190-140);
                glEnd();
glBegin(GL_POLYGON);
        glColor3f(1,1,0);
                glVertex2f(153,190-140);
                glVertex2f(154,191-140);
                glVertex2f(155,188-140);
                glVertex2f(154,185-140);
```

```
            glVertex2f(153,186-140);
            glVertex2f(153,190-140);
            glEnd();
        glBegin(GL_POLYGON);/*flowers*/
            glColor3f(1,1,0);
            glVertex2f(160,200-140);
            glVertex2f(163,200-140);
            glVertex2f(163,196-140);
            glVertex2f(160,196-140);
            glVertex2f(160,200-140);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(1,0,0);
            glVertex2f(160,200-140);
            glVertex2f(159,201-140);
            glVertex2f(162,203-140);
            glVertex2f(164,201-140);
            glVertex2f(163,200-140);
            glVertex2f(160,200-140);
            glEnd();
        glBegin(GL_POLYGON);
            glColor3f(1,0,0);
            glVertex2f(163,200-140);
            glVertex2f(164,201-140);
            glVertex2f(165,198-140);
            glVertex2f(164,195-140);
            glVertex2f(163,196-140);
            glVertex2f(163,200-140);
            glEnd();
        glBegin(GL_POLYGON);/*flowers*/
            glColor3f(0,0,1);
            glVertex2f(170,190-140);
            glVertex2f(173,190-140);
            glVertex2f(173,186-140);
            glVertex2f(170,186-140);
            glVertex2f(170,190-140);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(1,0,0);
            glVertex2f(170,190-140);
            glVertex2f(169,191-140);
            glVertex2f(172,193-140);
            glVertex2f(174,191-140);
            glVertex2f(173,190-140);
            glVertex2f(170,190-140);
            glEnd();
            glBegin(GL_POLYGON);
            glColor3f(1,0,0);
```

```
                glVertex2f(173,190-140);
                glVertex2f(174,191-140);
                glVertex2f(175,188-140);
                glVertex2f(174,185-140);
                glVertex2f(173,186-140);
                glVertex2f(173,190-140);
                glEnd();
        glBegin(GL_POLYGON);/*flowers*/
                glColor3f(0,1,1);
                glVertex2f(180,200-140);
                glVertex2f(183,200-140);
                glVertex2f(183,196-140);
                glVertex2f(180,196-140);
                glVertex2f(180,200-140);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(0,1,0);
                glVertex2f(180,200-140);
                glVertex2f(179,201-140);
                glVertex2f(182,203-140);
                glVertex2f(184,201-140);
                glVertex2f(183,200-140);
                glVertex2f(180,200-140);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(0,1,0);
                glVertex2f(183,200-140);
                glVertex2f(184,201-140);
                glVertex2f(185,198-140);
                glVertex2f(184,195-140);
                glVertex2f(183,196-140);
                glVertex2f(183,200-140);
                glEnd();
                glBegin(GL_POLYGON);/*flowers*/
                glColor3f(1,0,1);
                glVertex2f(170,210-140);
                glVertex2f(173,210-140);
                glVertex2f(173,206-140);
                glVertex2f(170,206-140);
                glVertex2f(170,210-140);
                glEnd();
                glBegin(GL_POLYGON);
                glColor3f(1,0,0);
                glVertex2f(170,210-140);
                glVertex2f(169,211-140);
                glVertex2f(172,213-140);
                glVertex2f(174,211-140);
                glVertex2f(173,210-140);
```

```
                        glVertex2f(170,210-140);
                        glEnd();
                        glBegin(GL_POLYGON);
                        glColor3f(1,0,0);
                        glVertex2f(173,210-140);
                        glVertex2f(174,211-140);
                        glVertex2f(175,208-140);
                        glVertex2f(174,205-140);
                        glVertex2f(173,206-140);
                        glVertex2f(173,210-140);
                        glEnd();
                        glBegin(GL_POLYGON);/*flowers*/
                        glColor3f(1,0,0);
                        glVertex2f(190,190-140);
                        glVertex2f(193,190-140);
                        glVertex2f(193,186-140);
                        glVertex2f(190,186-140);
                        glVertex2f(190,190-140);
                        glEnd();
                        glBegin(GL_POLYGON);
                        glColor3f(1,1,0);
                        glVertex2f(190,190-140);
                        glVertex2f(189,191-140);
                        glVertex2f(192,193-140);
                        glVertex2f(194,191-140);
                        glVertex2f(193,190-140);
                        glVertex2f(190,190-140);
                        glEnd();
                        glBegin(GL_POLYGON);
                        glColor3f(1,1,0);
                        glVertex2f(193,190-140);
                        glVertex2f(194,191-140);
                        glVertex2f(195,188-140);
                        glVertex2f(194,185-140);
                        glVertex2f(193,186-140);
                        glVertex2f(193,190-140);
                        glEnd();}
                        glutPostRedisplay();}
```

/**** Function to draw a man next to the statue******/**

```
        void man()
        {
        glMatrixMode(GL_PROJECTION);
                glLoadIdentity();
                gluOrtho2D(-10,100,-10,100);
        //left hand
        glBegin(GL_POLYGON);
```

```
glColor3f(0.9,0.9,0.8);
glVertex2f(22.8-4,47.7-3);
glVertex2f(25-4,46.2-3);
glVertex2f(25-4,46.2-3);
glVertex2f(24.5-4,43.7-3);
glVertex2f(24.5-4,43.7-3);
glVertex2f(23.3-4,44.2-3);
glVertex2f(23.3-4,44.2-3);
glVertex2f(22.8-4,47.7-3);
glEnd();
glBegin(GL_POLYGON);
//glColor3f(0.5,0.4,0.5);
glVertex2f(25-4,46.2-3);
glVertex2f(30-4,47.2-3);
glVertex2f(30-4,47.2-3);
glVertex2f(30.5-4,45.7-3);
glVertex2f(30.5-4,45.7-3);
glVertex2f(24.5-4,43.7-3);
glVertex2f(24.5-4,43.7-3);
glVertex2f(25-4,46.2-3);
glEnd();
glBegin(GL_POLYGON);
//glColor3f(0.5,0.4,0.5);
glVertex2f(30-4,47.2-3);
glVertex2f(32-4,48.7-3);
glVertex2f(32-4,48.7-3);
glVertex2f(32.5-4,48.2-3);
glVertex2f(32.5-4,48.2-3);
glVertex2f(33.5-4,47.2-3);
glVertex2f(33.5-4,47.2-3);
glVertex2f(32.5-4,46.2-3);
glVertex2f(32.5-4,46.2-3);
glVertex2f(30.5-4,45.7-3);
glVertex2f(30.5-4,45.7-3);
glVertex2f(30-4,47.2-3);
glEnd();
//right leg
glBegin(GL_POLYGON);
glColor3f(0.9,0.9,0.8);
glVertex2f(22-1,31-4);
glVertex2f(21.5-1,25-4);
glVertex2f(21.5-1,25-4);
glVertex2f(21-1,23-4);
glVertex2f(21-1,23-4);
glVertex2f(20.5-1,24.5-4);
glVertex2f(20.5-1,24.5-4);
glVertex2f(20-1,31-4);
glVertex2f(20-1,31-4);
```

```
glVertex2f(22-1,31-4);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(21.5-1,25-4);
glVertex2f(22.5-1,25-4);
glVertex2f(22.5-1,25-4);
glVertex2f(23-1,24.5-4);
glVertex2f(23-1,24.5-4);
glVertex2f(23-1,24-4);
glVertex2f(23-1,24-4);
glVertex2f(22.5-1,23-4);
glVertex2f(22.5-1,23-4);
glVertex2f(21-1,23-4);
glVertex2f(21-1,23-4);
glVertex2f(21.5-1,25-4);
glEnd();
//LEFT LEg
glBegin(GL_POLYGON);
glColor3f(0.9,0.9,0.8);
glVertex2f(19.5-1,30.5-4);
glVertex2f(20-1,23-4);
glVertex2f(20-1,23-4);
glVertex2f(20.5-1,23-4);
glVertex2f(20.5-1,23-4);
glVertex2f(21-1,23-4);
glVertex2f(21-1,23-4);
glVertex2f(20.5-1,24.5-4);
glVertex2f(20.5-1,24.5-4);
glVertex2f(20-1,31-4);
glVertex2f(20-1,31-4);
glVertex2f(19.5-1,30.5-4);
glEnd();
//head
glBegin(GL_POLYGON);
glColor3f(0,0,0);
glVertex2f(18,55+1);
glVertex2f(18.5,56.4+1);
glVertex2f(18.5,56.4+1);
glVertex2f(19.5,56.9+1);
glVertex2f(19.5,56.9+1);
glVertex2f(22.5,56+1);
glVertex2f(22.5,56+1);
glVertex2f(22.3,55+1);
glVertex2f(22.3,55+1);
glVertex2f(18,55+1);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.9,0.9,0.8);
```

```
glVertex2f(18,55);
glVertex2f(18.5,56);
glVertex2f(18.5,56);
glVertex2f(19.5,56.2);
glVertex2f(19.5,56.2);
glVertex2f(22.5,55.9);
glVertex2f(22.5,55.9);
glVertex2f(22.3,55);
glVertex2f(22.3,55);
glVertex2f(18,55);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(18,55);
glVertex2f(22.3,55);
glVertex2f(22.3,55);
glVertex2f(22.8,53);
glVertex2f(22.8,53);
glVertex2f(22.3,52.8);
glVertex2f(22.3,52.8);
glVertex2f(18.5,51);
glVertex2f(18.5,51);
glVertex2f(18,52);
glVertex2f(18,52);
glVertex2f(18,55);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(18.5,51);
glVertex2f(22.2,52.8);
glVertex2f(22.2,52.8);
glVertex2f(22.4,52);
glVertex2f(22.4,52);
glVertex2f(21.8,51.5);
glVertex2f(21.8,51.5);
glVertex2f(22.4,51.5);
glVertex2f(22.4,51.5);
glVertex2f(18.5,51);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(18.5,51);
glVertex2f(21.8,51.5);
glVertex2f(22.4,51.5);
glVertex2f(22.4,51.5);
glVertex2f(22.2,50.5);
glVertex2f(22.2,50.5);
glVertex2f(21.6,50.2);
glVertex2f(21.6,50.2);
glVertex2f(21.4,50);
glVertex2f(21.4,50);
```

```
glVertex2f(21.8,49);
glVertex2f(21.8,49);
glVertex2f(18.5,51);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(19,49+3);
glVertex2f(21,49+3);
glVertex2f(21,45+3);
glVertex2f(19,45+3);
glVertex2f(19,49+3);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.9,0.5,0.9);
glVertex2f(19,48);
glVertex2f(17,46);
glVertex2f(17,36);
glVertex2f(23,36);
glVertex2f(23,46);
glVertex2f(21,48);
glVertex2f(19,48);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.5,0.3,0.5);
glVertex2f(17,36);
glVertex2f(23,36);
glVertex2f(23,32);
glVertex2f(17,32);
glVertex2f(17,36);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1,0.25,0);
glVertex2f(17,32);
glVertex2f(20,32);
glVertex2f(20,27);
glVertex2f(19.5,25);
glVertex2f(17.5,25);
glVertex2f(17,27);
glVertex2f(17,32);
glEnd();
glBegin(GL_POLYGON);
glColor3f(1,0.25,0);
glVertex2f(20,32);
glVertex2f(23,32);
glVertex2f(23,27);
glVertex2f(22.5,25);
glVertex2f(19.5,25);
glVertex2f(20,32);
glEnd();
```

```
//right hand
glBegin(GL_POLYGON);
glColor3f(0.9,0.9,0.8);
glVertex2f(21.5,44.8);
glVertex2f(22.5,42.8);
glVertex2f(22.5,42.8);
glVertex2f(22,40.3);
glVertex2f(22,40.3);
glVertex2f(20,43.3);
glVertex2f(20,43.3);
glVertex2f(21.5,44.8);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(22.5,42.8);
glVertex2f(26.5,46.8);
glVertex2f(26.5,46.8);
glVertex2f(27,45.3);
glVertex2f(27,45.3);
glVertex2f(22,40.3);
glVertex2f(22,40.3);
glVertex2f(22.5,42.8);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(26.5,46.8);
glVertex2f(28.5,48.3);
glVertex2f(28.5,48.3);
glVertex2f(30,46.8);
glVertex2f(30,46.8);
glVertex2f(29,45.8);
glVertex2f(29,45.8);
glVertex2f(27,45.3);
glVertex2f(27,45.3);
glVertex2f(26.5,46.8);
glEnd();
//eyes
glBegin(GL_POLYGON);
glColor3f(0.0,0.0,0.0);
glVertex2f(21.3,54.7);
glVertex2f(21.8,54.2);
glVertex2f(22.3,54.7);
glVertex2f(21.8,55.2);
glEnd();
glBegin(GL_LINE_STRIP);
glVertex2f(21.3,55);
glVertex2f(21.8,55.5);
glVertex2f(22.3,55);
```

```
        glEnd();}

/****** Function to draw tree******/

        void tree()
        {glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(10,100.0,10,100.0);
        glBegin(GL_POLYGON);
        glColor3f(0,0.8,0.1);
        glVertex2f(28,65);
        glVertex2f(30,64.5);
         glVertex2f(32.5,64);
        glVertex2f(34,64.5);
         glVertex2f(35.5,65);
         glVertex2f(39,66);
         glVertex2f(37,67.5);
         glVertex2f(37,68);
         glVertex2f(39,69.5);
        glVertex2f(39,71.5);
        glVertex2f(39.3,72);
        glVertex2f(39.6,73);
        glVertex2f(39.9,73.5);
        glVertex2f(40,74);
         glVertex2f(40.5,75);
        glVertex2f(40,76.5);
         glVertex2f(43,77.5);
         glVertex2f(42.5,79);
         glVertex2f(43,80);
         glVertex2f(41.5,82.5);
         glVertex2f(40.5,82);
        glVertex2f(40,81.5);
        glVertex2f(41,82.5);
        glVertex2f(41.5,83.5);
        glVertex2f(42,85);
        glVertex2f(39,87);
        glVertex2f(37,88.5);
        glVertex2f(33,86);
        glVertex2f(32,84);
        glVertex2f(33,90);
        glVertex2f(28,65);
        glEnd();
        //root
        glBegin(GL_POLYGON);
        glColor3f(0.65,0.50,0.39);
        glVertex2f(26.5,49);
        glVertex2f(28,48.5);
         glVertex2f(30,50);
```

```
 glVertex2f(28.5,52.5);
 glVertex2f(28,55);
 glVertex2f(28.5,67.5);
 glVertex2f(28,55);
 glVertex2f(28,60);
 glVertex2f(28.5,67.5);
 glVertex2f(28,65);
 glVertex2f(28.5,67.5);
 glVertex2f(29.5,71);
 glVertex2f(30,72.5);
 glVertex2f(25,80);
 glVertex2f(25,50);
 glVertex2f(26.5,49);
 glEnd();
//tree 1 branch
glShadeModel(GL_SMOOTH);
glBegin(GL_POLYGON);
glColor3f(0.0,0.9,0.2);
glVertex2f(16.5,70);
glVertex2f(12.5,70);
 glVertex2f(11,71.5);
 glVertex2f(10,72.5);
glColor3f(0.0,0.85,0.1);
 glVertex2f(10,75);
 glVertex2f(7.5,75);
 glVertex2f(6,76);
 glVertex2f(5,78.5);
 glVertex2f(5,80);
 glVertex2f(7.5,82.5);
 glVertex2f(5,85);
 glVertex2f(2.7,87);
 glVertex2f(2.75,90);
glColor3f(0.1,0.6,0.1);
 glVertex2f(2.8,91);
 glVertex2f(3.5,92.5);
 glVertex2f(5,95);
 glVertex2f(7,97.5);
 glVertex2f(10,98);
 glVertex2f(15,99);
 glColor3f(0.1,0.6,0.2);
 glVertex2f(20,98.5);
glVertex2f(25,98);
 glVertex2f(30,97.5);
 glVertex2f(30.5,97);
 glVertex2f(30,95);
 glVertex2f(32.5,92.5);
 glVertex2f(34,87.5);
glVertex2f(35,85);
```

```
glVertex2f(34.5,82.5);
glVertex2f(34,79);
glVertex2f(34.5,77.5);
 glColor3f(0.1,0.6,0.0);
glVertex2f(33,76);
glVertex2f(32.5,74.5);
glVertex2f(31,73);
glVertex2f(30,72.5);
glVertex2f(27.5,72.5);
glVertex2f(26,73);
glVertex2f(25,75);
glEnd();
//tree1 root
glBegin(GL_POLYGON);
glColor3f(0.8,0.6,0);
glVertex2f(17.5,56);
glVertex2f(17.0,62.5);
glVertex2f(17.5,67.5);
glVertex2f(16.5,70.0);
glVertex2f(16.0,72.5);
glVertex2f(15.5,75.0);
glVertex2f(15.0,80.0);
glVertex2f(20.0,85.0);
glVertex2f(21.5,77.0);
glVertex2f(26.0,87.0);
glVertex2f(30.0,85.0);
glVertex2f(28.0,80.0);
glVertex2f(25.5,75.0);
glVertex2f(25,72.5);
glVertex2f(25,70.0);
glVertex2f(25.0,55.0);
glVertex2f(25.5,52.5);
glVertex2f(26.0,50.0);
glVertex2f(26.5,48.0);
glVertex2f(27.0,47.5);
glVertex2f(28.5,46.0);
glVertex2f(29,44);
glVertex2f(30.0,40.0);
glVertex2f(20.0,47.5);
glVertex2f(17.5,47.0);
glVertex2f(18.0,50.0);
glVertex2f(17.0,54.0);
glVertex2f(20.0,53.0);
glVertex2f(17.5,56.0);
glEnd();}
```

/**** Function for background******/**

```
void back()
{glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(-20,100.0,-20,100.0);
glBegin(GL_POLYGON);
glColor3f(0.0,0.8,0.1);
glVertex2f(28,65);
glVertex2f(30,64.5);
 glVertex2f(32.5,64);
glVertex2f(34,64.5);
 glVertex2f(35.5,65);
 glVertex2f(39,66);
 glVertex2f(37,67.5);
 glVertex2f(37,68);
 glVertex2f(39,69.5);
glVertex2f(39,71.5);
glVertex2f(39.3,72);
glVertex2f(39.6,73);
glVertex2f(39.9,73.5);
glVertex2f(40,74);
 glVertex2f(40.5,75);
glVertex2f(40,76.5);
 glVertex2f(43,77.5);
 glVertex2f(42.5,79);
 glVertex2f(43,80);
 glVertex2f(41.5,82.5);
 glVertex2f(40.5,82);
glVertex2f(40,81.5);
glVertex2f(41,82.5);
glVertex2f(41.5,83.5);
glVertex2f(42,85);
glVertex2f(39,87);
glVertex2f(37,88.5);
glVertex2f(33,86);
glVertex2f(32,84);
glVertex2f(33,90);
glVertex2f(28,65);
glEnd();
//root
glBegin(GL_POLYGON);
glColor3f(0.65,0.50,0.39);
glVertex2f(26.5,49);
glVertex2f(28,48.5);
 glVertex2f(30,50);
 glVertex2f(28.5,52.5);
 glVertex2f(28,55);
 glVertex2f(28.5,67.5);
 glVertex2f(28,55);
```

```
 glVertex2f(28,60);
 glVertex2f(28.5,67.5);
 glVertex2f(28,65);
 glVertex2f(28.5,67.5);
 glVertex2f(29.5,71);
 glVertex2f(30,72.5);
 glVertex2f(25,80);
 glVertex2f(25,50);
 glVertex2f(26.5,49);
 glEnd();
//tree 1 branch
glShadeModel(GL_SMOOTH);
glBegin(GL_POLYGON);
glColor3f(0.0,0.9,0.2);
glVertex2f(16.5,70);
glVertex2f(12.5,70);
 glVertex2f(11,71.5);
 glVertex2f(10,72.5);
glColor3f(0.0,0.85,0.1);
 glVertex2f(10,75);
 glVertex2f(7.5,75);
 glVertex2f(6,76);
 glVertex2f(5,78.5);
 glVertex2f(5,80);
 glVertex2f(7.5,82.5);
 glVertex2f(5,85);
 glVertex2f(2.7,87);
 glVertex2f(2.75,90);
glColor3f(0.1,0.6,0.1);
 glVertex2f(2.8,91);
 glVertex2f(3.5,92.5);
 glVertex2f(5,95);
 glVertex2f(7,97.5);
 glVertex2f(10,98);
 glVertex2f(15,99);
 glColor3f(0.1,0.6,0.2);
 glVertex2f(20,98.5);
glVertex2f(25,98);
 glVertex2f(30,97.5);
 glVertex2f(30.5,97);
 glVertex2f(30,95);
 glVertex2f(32.5,92.5);
 glVertex2f(34,87.5);
glVertex2f(35,85);
 glVertex2f(34.5,82.5);
 glVertex2f(34,79);
 glVertex2f(34.5,77.5);
  glColor3f(0.1,0.6,0.0);
```

```
 glVertex2f(33,76);
 glVertex2f(32.5,74.5);
 glVertex2f(31,73);
 glVertex2f(30,72.5);
 glVertex2f(27.5,72.5);
 glVertex2f(26,73);
 glVertex2f(25,75);
glEnd();
//tree1 root
glBegin(GL_POLYGON);
glColor3f(0.8,0.6,0.0);
glVertex2f(17.5,56);
glVertex2f(17.0,62.5);
 glVertex2f(17.5,67.5);
glVertex2f(16.5,70.0);
 glVertex2f(16.0,72.5);
 glVertex2f(15.5,75.0);
 glVertex2f(15.0,80.0);
 glVertex2f(20.0,85.0);
 glVertex2f(21.5,77.0);
glVertex2f(26.0,87.0);
glVertex2f(30.0,85.0);
 glVertex2f(28.0,80.0);
glVertex2f(25.5,75.0);
 glVertex2f(25,72.5);
 glVertex2f(25,70.0);
 glVertex2f(25.0,55.0);
 glVertex2f(25.5,52.5);
 glVertex2f(26.0,50.0);
glVertex2f(26.5,48.0);
glVertex2f(27.0,47.5);
glVertex2f(28.5,46.0);
glVertex2f(29,44);
glVertex2f(30.0,40.0);
glVertex2f(20.0,47.5);
glVertex2f(17.5,47.0);
glVertex2f(18.0,50.0);
glVertex2f(17.0,54.0);
 glVertex2f(20.0,53.0);
 glVertex2f(17.5,56.0);
 glEnd();
//tree2half
glBegin(GL_POLYGON);
glColor3f(0.65,0.50,0.39);
glVertex2f(75,48);
 glVertex2f(75.5,58);
 glVertex2f(75.6,60);
 glVertex2f(75.5,61);
```

```
 glVertex2f(75,65);
 glVertex2f(74.5,66);
 glVertex2f(73,69);
 glVertex2f(72.5,70);
 glVertex2f(72.5,77.5);
 glVertex2f(75,80);
 glVertex2f(81,67.5);
 glVertex2f(78,48);
glEnd();
//tree branch 2
glBegin(GL_POLYGON);
glColor3f(0.0,0.6,0.3);
glVertex2f(75,75);
glVertex2f(72.5,72.5);
 glVertex2f(71,74.0);
 glVertex2f(70.5,74.5);
 glVertex2f(70,75);
 glVertex2f(69.5,76);
 glVertex2f(69.5,77);
 glVertex2f(70,80);
 glVertex2f(68,81);
 glVertex2f(67.5,82);
 glVertex2f(66,83.5);
 glVertex2f(66.3,84.5);
 glVertex2f(66.5,85);
 glVertex2f(67,87);
 glVertex2f(67.5,87.5);
 glVertex2f(70,90);
 glVertex2f(71,91.5);
 glVertex2f(71.5,92.5);
 glVertex2f(72.5,94);
 glVertex2f(75,95);
 glVertex2f(77.5,96);
 glVertex2f(80,96.5);
 glVertex2f(82.5,96);
 glVertex2f(83,95);
  glVertex2f(83.5,94.5);
  glVertex2f(84.5,92.5);
 glVertex2f(84.7,90);
 glVertex2f(86,91);
 glVertex2f(87.0,91.5);
 glVertex2f(87.5,90);
 glVertex2f(87.5,89);
 glVertex2f(89,89);
 glVertex2f(89.5,87.5);
 glVertex2f(89,85);
glVertex2f(88,82.5);
 glVertex2f(87,82);
```

```
 glVertex2f(86,81);
 glVertex2f(85,80);
 glVertex2f(85.5,75.5);
 glVertex2f(84,75);
glVertex2f(82.5,75);
glEnd();
//tree 2
glBegin(GL_POLYGON);
glColor3f(0.8,0.6,0.0);
glVertex2f(77.5,48);
glVertex2f(77,60);
 glVertex2f(77.5,62.5);
 glVertex2f(77,65);
 glVertex2f(77,67.5);
 glVertex2f(76,70);
 glVertex2f(75,75);
 glVertex2f(75,78);
 glVertex2f(80,75);
 glVertex2f(82.5,78);
 glVertex2f(82,75);
 glVertex2f(82,72.5);
 glVertex2f(81.5,70);
 glVertex2f(81,65);
 glVertex2f(81.5,62);
 glVertex2f(84,48);
glEnd();}
```

/**** Function for ground ******/**

```
void grass()
{
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0,150.0,0.0,150.0);
//grass 2
//rightt grass
glBegin(GL_POLYGON);
glColor3f(0.0,0.9,0.3);
glVertex2f(0,7.5);
 glVertex2f(5,13);
 glVertex2f(0,15);
 glVertex2f(5,13);
 glVertex2f(0,20);
 glVertex2f(5,15);
 glVertex2f(10,18);
 glVertex2f(15,25);
 glVertex2f(13,20); glEnd();}
```

Implementation Details
e. Makar Sankranthi

//Decription Window

```
void display3()
{glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
   glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.25,0.8,0);
   bitmap_output(0,0,0,"Makara Sankranthi",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(0.0,-0.85,0.0);
   glVertex3f(0.0,-0.95,0.0);
   glVertex3f(0.2,-0.95,0.0);
   glVertex3f(0.2,-0.85,0.0);
   glEnd();
   glPushMatrix();
   glTranslatef(0.0,-0.92,0.0);
   bitmap_output(0,0,0,"Click",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-1.0,0.6,0);
bitmap_output(0,0,0,"**************************************************
****",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.4,0);
   bitmap_output(0,0,0,"Makar Sankranti is the celebration of the harvest
festival",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.3,0);
```

```
   bitmap_output(0,0,0,"We all know that spring is the most pleasant of all
seasons",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.2,0);
   bitmap_output(0,0,0,"It is accompanied by pleasant weather, calm winds, and a
sunny",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.1,0);
   bitmap_output(0,0,0,"but not scorching weather All of these reasons together make
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0,0);
   bitmap_output(0,0,0,"spring the perfect season to grow crops and sustain high crop
yield",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.1,0);
   bitmap_output(0,0,0,"The fest is also called the Kite festival in several parts of our
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.2,0);
   bitmap_output(0,0,0,"country India Kites are flown on this day to honor the Sun God
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.3,0);
   bitmap_output(0,0,0,"It is celebrated a day after the Lohri festival celebrated mostly
in",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.4,0);
   bitmap_output(0,0,0,"Punjab and Chandigarh Sweets and clothes are exchanged
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.5,0);
   bitmap_output(0,0,0,"among families",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.7,0);
   bitmap_output(0,0,0,"Here is the simple simulation of Makar Sankranthi(Kite
Festival),",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
```

```
        glTranslatef(-0.8,-0.8,0);
        bitmap_output(0,0,0,"Press on click to view the
    festival:",GLUT_BITMAP_TIMES_ROMAN_24);
        glPopMatrix();
        glFlush();
        glPopMatrix();
        glutPostRedisplay();}
```

/*Makar Sankranti Simulation*/

/**** Function to draw sun ******/**

```
void drawSunm()
{
   glPushMatrix();
   glColor3f(3,1.5,0.0);
   DrawCircles(150,700,45,1000);
   glColor3f(3,1,0.5);
   DrawCircles(150,700,40,1000);
   glPopMatrix();

}
```

/**** Function to draw tree ******/**

```
void treem(int x,int y)
{       glPushMatrix();
        glBegin(GL_POLYGON); //Tree
        glColor3f(0.6, 0.3, 0.0);
        glVertex2i(x+150, 450+y);
        glVertex2i(x+165, 450+y);
        glVertex2i(x+165, 550+y);
        glVertex2i(x+150, 550+y);
        glEnd();
        glColor3f(0.0, 0.7, 0.1);
        DrawCircles(x+130, 545+y, 40, 1000); // 4
        glColor3f(0.0, 0.7, 0.1);
        DrawCircles(x+155, 580+y, 40, 1000);
        glColor3f(0.0, 0.7, 0.1);
        DrawCircles(x+170, 550+y, 40, 1000);
        glPopMatrix();}
```

/**** Function for background ******/**

```
void scene1()
```

```
{       glPushMatrix();
        glBegin(GL_POLYGON); //Sky
        glColor3f(1.3, 0.8, 0.5);
        glVertex2i(0, 800);
        glVertex2i(1200, 800);
        glColor3f(0.7, 0.7, 1.0);
        glVertex2i(1200, 0);
        glVertex2i(0, 0);
        glEnd();
        glBegin(GL_POLYGON); // ground
        glColor3f(0.3, 1, 0.3);
        glVertex2i(0, 500);
        glVertex2i(1200,500);
        glVertex2i(1200,0);
        glVertex2i(0, 0);
        glEnd();
        glBegin(GL_POLYGON); //Road
        glColor3f(0.7, 0.3, 0.3);
        glVertex2i(400, 0);
        glVertex2i(500, 500);
        glVertex2i(600, 500);
        glVertex2i(700, 0);
        glEnd();
        glColor3f(0.0,0.9,0);//river
        DrawCircles(1200,0,300,1000);
        DrawCircles(1000,0,200,1000);
        glColor3f(0.5,2.3,3);//river
        DrawCircles(1200,0,290,1000);
        DrawCircles(1000,0,190,1000);
        glBegin(GL_POLYGON); //Road
        glColor3f(0.3,0.3,0.3);
        glVertex2i(410, 0);
        glVertex2i(510, 500);
        glVertex2i(590, 500);
        glVertex2i(690, 0);
        glEnd();
   glBegin(GL_LINES);
   glColor3f(1,1,1);
   glPointSize(3);
   glLineWidth(10);
   glVertex2i(550,500);
   glVertex2i(550,450);
   glVertex2i(550,400);
   glVertex2i(550,350);
   glVertex2i(550,300);
   glVertex2i(550,250);
   glVertex2i(550,200);
   glVertex2i(550,150);
```

```
   glVertex2i(550,100);
   glVertex2i(550,20);
   glEnd();
         glPopMatrix();}

/****** Function for house ******/

void home()
{   glPushMatrix();
   glColor3f(0.6,0.5,1.5);
   glBegin(GL_POLYGON);
   glVertex2i(10,350);
   glVertex2i(70,450);
   glVertex2i(130,350);
   glVertex2i(130,200);
   glVertex2i(10,200);
   glEnd();
   glColor3f(0.2,0.2,2);
   glBegin(GL_POLYGON);
   glVertex2i(130,200);
   glVertex2i(250,250);
   glVertex2i(250,350);
   glVertex2i(130,350);
   glEnd();
   glColor3f(0.7,0,0);
   glBegin(GL_POLYGON);
   glVertex2i(130,350);
   glVertex2i(70,450);
   glVertex2i(200,410);
   glVertex2i(250,350);
   glEnd();
   glColor3f(2.8,0.7,0.2);
   glBegin(GL_POLYGON);
   glVertex2i(150,208);
   glVertex2i(150,340);
   glVertex2i(190,340);
   glVertex2i(190,225);
   glEnd();
   glBegin(GL_POLYGON);
   glVertex2i(210,290);
   glVertex2i(210,320);
   glVertex2i(235,320);
   glVertex2i(235,295);
   glEnd();
   glColor3f(0.3,0,0);
   DrawCircles(70,400,10,1000);
   glBegin(GL_POLYGON);
   glColor3f(2.8,1.7,0.6);
```

```
glVertex2i(440,480);
glVertex2i(330,480);
glVertex2i(330,380);
glVertex2i(440,380);
glEnd();
glBegin(GL_POLYGON);//door
glColor3f(0.5,0,0);
glVertex2i(400,460);
glVertex2i(370,460);
glVertex2i(370,380);
glVertex2i(400,380);
glEnd();
glBegin(GL_POLYGON);//window
glVertex2i(360,450);
glVertex2i(340,450);
glVertex2i(340,430);
glVertex2i(360,430);
glEnd();
glBegin(GL_POLYGON);//window
glVertex2i(410,450);
glVertex2i(430,450);
glVertex2i(430,430);
glVertex2i(410,430);
glEnd();
glBegin(GL_POLYGON);
glColor3f(0.7,0,0);
glVertex2i(320,480);
glVertex2i(385,550);
glVertex2i(450,480);
glEnd();
glBegin(GL_POLYGON);
glColor3f(2.8,0.5,1.6);
glVertex2i(840,580);
glVertex2i(730,580);
glVertex2i(730,480);
glVertex2i(840,480);
glEnd();
glBegin(GL_POLYGON);//door
glColor3f(0.5,0,0);
glVertex2i(800,560);
glVertex2i(770,560);
glVertex2i(770,480);
glVertex2i(800,480);
glEnd();
glBegin(GL_POLYGON);//window
glVertex2i(760,550);
glVertex2i(740,550);
glVertex2i(740,530);
```

```
   glVertex2i(760,530);
   glEnd();
   glBegin(GL_POLYGON);//window
   glVertex2i(810,550);
   glVertex2i(830,550);
   glVertex2i(830,530);
   glVertex2i(810,530);
   glEnd();
   glBegin(GL_POLYGON);
   glColor3f(0.7,0,0);
   glVertex2i(720,580);
   glVertex2i(785,650);
   glVertex2i(850,580);
   glEnd();
   glPopMatrix();}
```

/**** Function for sweet pot ******/**

```
void pot()
{   glPushMatrix();
   glColor3f(0.5,0,0);
   glBegin(GL_POLYGON);
   glVertex2i(250,50);
   glVertex2i(350,50);
   glVertex2i(320,90);
   glVertex2i(280,90);
   glEnd();
   glColor3f(3.0,1.0,0.0);
   glBegin(GL_POLYGON);
   glVertex2i(270,50);
   glVertex2i(330,50);
   glVertex2i(315,90);
   glVertex2i(285,90);
   glEnd();
   glColor3f(1,1,1);
   DrawCircles(300,140,20,1000);
   glColor3f(3.0,0.5,0.0);
   DrawCircles(300,115,30,1000);
   glBegin(GL_POLYGON);
   glVertex2i(280,135);
   glVertex2i(280,145);
   glVertex2i(320,145);
   glVertex2i(320,135);
   glEnd();
   glColor3f(0.5,0,0);
   glBegin(GL_POLYGON);
   glVertex2i(260,32);
   glVertex2i(260,30);
```

```
glVertex2i(180,50);
glVertex2i(180,52);
glEnd();
glColor3f(0.0,0.2,0);
glBegin(GL_POLYGON);
glVertex2i(180,55);
glVertex2i(130,52.5);
glVertex2i(180,50);
glEnd();
glBegin(GL_POLYGON);
glVertex2i(180,50);
glVertex2i(150,52.5);
glVertex2i(190,40);
glVertex2i(160,42);
glEnd();
glColor3f(0.5,0,0);
glBegin(GL_POLYGON);
glVertex2i(260,37);
glVertex2i(260,35);
glVertex2i(180,45);
glVertex2i(180,47);
glEnd();
glColor3f(0.0,0.2,0);
glBegin(GL_POLYGON);
glVertex2i(180,60);
glVertex2i(130,55.5);
glVertex2i(180,55);
glEnd();
glBegin(GL_POLYGON);
glVertex2i(180,55);
glVertex2i(150,57.5);
glVertex2i(190,45);
glVertex2i(160,47);
glEnd();
glColor3f(0.5,0,0);
glBegin(GL_POLYGON);
glVertex2i(260,52);
glVertex2i(260,50);
glVertex2i(180,70);
glVertex2i(180,72);
glEnd();
glColor3f(0.0,0.2,0);
glBegin(GL_POLYGON);
glVertex2i(180,75);
glVertex2i(130,72.5);
glVertex2i(180,70);
glEnd();
glBegin(GL_POLYGON);
```

```
   glVertex2i(180,70);
   glVertex2i(150,72.5);
   glVertex2i(190,60);
   glVertex2i(160,62);
   glEnd();
   glPopMatrix();}

/****** Function for drawing people ******/

void drawpeople(int x ,int y)
{   glPushMatrix();
   glColor3f(0,0,0);
   DrawCircles(52+x,190+y,18,1000);
   glColor3f(1.0,0.7,0.5);
   DrawCircles(52+x,188+y,16,1000);
   glBegin(GL_LINES);
   glColor3f(0,0,0);
   glVertex2i(48+x,191+y);
   glVertex2i(48+x,194+y);
   glVertex2i(56+x,191+y);
   glVertex2i(56+x,194+y);
   glVertex2i(52+x,189+y);
   glVertex2i(52+x,185+y);
   glVertex2i(54+x,182+y);
   glVertex2i(50+x,182+y);
   glEnd();
   glBegin(GL_POLYGON);//body
   glColor3f(0,0,0.5);
   glVertex2i(37+x,172+y);
   glVertex2i(67+x,172+y);
   glVertex2i(67+x,135+y);
   glVertex2i(37+x,135+y);
   glEnd();
   glBegin(GL_POLYGON);//lefthand
   glColor3f(1.0,0.7,0.5);
   glVertex2i(37+x,172+y);
   glVertex2i(27+x,172+y);
   glVertex2i(27+x,140+y);
   glVertex2i(37+x,140+y);
   glEnd();
   glBegin(GL_POLYGON);//righthand
   glColor3f(1.0,0.7,0.5);
   glVertex2i(77+x,172+y);
   glVertex2i(67+x,172+y);
   glVertex2i(67+x,140+y);
   glVertex2i(77+x,140+y);
   glEnd();
   glBegin(GL_POLYGON);
```

```
    glColor3f(0.5,0,0.8);
    glVertex2i(37+x,135+y);
    glVertex2i(52+x,135+y);
    glVertex2i(48+x,105+y);
    glVertex2i(37+x,105+y);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(0.5,0,0.8);
    glVertex2i(52+x,135+y);
    glVertex2i(67+x,135+y);
    glVertex2i(67+x,105+y);
    glVertex2i(56+x,105+y);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(1.0,0.7,0.5);
    glVertex2i(56+x,95+y);
    glVertex2i(67+x,95+y);
    glVertex2i(67+x,105+y);
    glVertex2i(56+x,105+y);
    glEnd();
    glBegin(GL_POLYGON);
    glColor3f(1.0,0.7,0.5);
    glVertex2i(37+x,95+y);
    glVertex2i(48+x,95+y);
    glVertex2i(48+x,105+y);
    glVertex2i(37+x,105+y);
    glEnd();
    glPopMatrix();}

void drawpeople1(int x ,int y)
{ glPushMatrix();
    glColor3f(0,0,0);
    DrawCircles(152+x,190+y,18,1000);
    glColor3f(1.0,0.7,0.5);
    DrawCircles(152+x,188+y,16,1000);
    glBegin(GL_LINES);
    glColor3f(0,0,0);
    glVertex2i(148+x,191+y);
    glVertex2i(148+x,194+y);
    glVertex2i(156+x,191+y);
    glVertex2i(156+x,194+y);
    glVertex2i(152+x,189+y);
    glVertex2i(152+x,185+y);
    glVertex2i(154+x,182+y);
    glVertex2i(150+x,182+y);
    glEnd();
    glBegin(GL_POLYGON);//body
    glColor3f(0.5,0,0.4);
```

```
   glVertex2i(137+x,172+y);
   glVertex2i(167+x,172+y);
   glVertex2i(167+x,135+y);
   glVertex2i(137+x,135+y);
   glEnd();
   glBegin(GL_POLYGON);//lefthand
   glColor3f(1.0,0.7,0.5);
   glVertex2i(137+x,172+y);
   glVertex2i(127+x,172+y);
   glVertex2i(127+x,140+y);
   glVertex2i(137+x,140+y);
   glEnd();
   glBegin(GL_POLYGON);//righthand
   glColor3f(1.0,0.7,0.5);
   glVertex2i(177+x,172+y);
   glVertex2i(167+x,172+y);
   glVertex2i(167+x,140+y);
   glVertex2i(177+x,140+y);
   glEnd();
   glBegin(GL_POLYGON);
   glColor3f(3,0.2,0.3);
   glVertex2i(137+x,135+y);
   glVertex2i(167+x,135+y);
   glVertex2i(167+x,105+y);
   glVertex2i(137+x,105+y);
   glEnd();
   glBegin(GL_POLYGON);//rightleg
   glColor3f(1.0,0.7,0.5);
   glVertex2i(156+x,95+y);
   glVertex2i(167+x,95+y);
   glVertex2i(167+x,105+y);
   glVertex2i(156+x,105+y);
   glEnd();    glPopMatrix();}
```

/**** Function for display ******/**

```
void displaym(void)
{ if(playThemeMusic2)
   {    PlaySound(TEXT("themel.wav"), NULL, SND_ASYNC|SND_FILENAME|SND_LOOP);
              playThemeMusic2 = 0; }
        scene1();
        drawSunm();
        home();
        pot();
        treem(-20,0);
        glFlush();
        glutPostRedisplay();
        glutSwapBuffers();}
```

Implementation Details
f. Diwali

/*Description Window*/

```
void display2()
{
   glClear(GL_COLOR_BUFFER_BIT );
   glPushMatrix();
   glColor3f(0.80,0.0,0.6);
   glBegin(GL_POLYGON);
   glVertex2f(1.0,1.0);
   glColor3f(0.90,0.4,0.0);
   glVertex2f(1.0,-1.0);
   glColor3f(0.60,0.10,0.0);
   glVertex2f(-1.0,-1.0);
   glColor3f(0.90,0.0,0.40);
   glVertex2f(-1.0,1.0);
   glEnd();
   glColor3f(1.0,1.0,1.0);
   glPushMatrix();
   glTranslatef(-0.25,0.8,0);
   bitmap_output(0,0,0,"Happy Diwali",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glColor3f(1.0,1.0,1.0);
   glBegin(GL_LINE_LOOP);
   glVertex3f(0.0,-0.85,0.0);
   glVertex3f(0.0,-0.95,0.0);
   glVertex3f(0.2,-0.95,0.0);
   glVertex3f(0.2,-0.85,0.0);
   glEnd();

   glPushMatrix();
   glTranslatef(0.0,-0.92,0.0);
   bitmap_output(0,0,0,"Click",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();

   glPushMatrix();
   glTranslatef(-1.0,0.6,0);

bitmap_output(0,0,0,"************************************************************",
GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.4,0);
```

```
   bitmap_output(0,0,0,"Diwali is one of the most important festivals in Hindu culture
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.3,0);
   bitmap_output(0,0,0,"It is one of the most important holidays in the Hindu
calendar",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.2,0);
   bitmap_output(0,0,0,"Diwali gets its name from the Sanskrit word ‘Deepavali’ that
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0.1,0);
   bitmap_output(0,0,0,"roughly translates to ‘a row of lights'It is a great time to have
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,0,0);
   bitmap_output(0,0,0,"fun with your family and friends People decorate their houses with
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.1,0);
   bitmap_output(0,0,0,"oil lamps and fairy lights at night They cook delicious food
and",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.2,0);
   bitmap_output(0,0,0,"share them with their neighbors. It is a festival of lights; every
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.3,0);
   bitmap_output(0,0,0,"street is lit up with beautiful lights & people burst crackers and
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.4,0);
   bitmap_output(0,0,0,"celebrate it joyfully It spreads cheer and joy among people and
",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glPushMatrix();
   glTranslatef(-0.8,-0.5,0);
   bitmap_output(0,0,0,"makes them fall in love with their
culture",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
```

```
   glPushMatrix();
   glTranslatef(-0.8,-0.7,0);
   bitmap_output(0,0,0,"Here is the simple simulation of Diwali
Festival,",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
    glPushMatrix();
   glTranslatef(-0.8,-0.8,0);
   bitmap_output(0,0,0,"Press on click to view the festival:",GLUT_BITMAP_TIMES_ROMAN_24);
   glPopMatrix();
   glFlush();
   glPopMatrix();
   glutPostRedisplay();
}
```

//Diwali Simulation

/**** Functions for background******/**

```
void fillLowerBG()
{   GLubyte upperGreenColor[] = {20, 71, 1};
    GLubyte lowerGreenColor[] = {68, 210, 12};
    glBegin(GL_POLYGON) ;
    glColor3ubv(upperGreenColor);
    glVertex3i(-5, -1,-1);
    glVertex3i(5, -1,-1);
    glColor3ubv(lowerGreenColor);
    glVertex3i(5, -5,-1);
    glVertex3i(-5, -5,-1);
    glEnd();}

void fillUpperBG()
{
    glBegin(GL_POLYGON);
                glColor3ub(0,0,0);
                glVertex3f(-10,10,-5);
                glColor3ub(60,60,60);
                glVertex3f(10,10,-5);
                glColor3ub(1,5,2);
                glVertex3f(10,-10,-5);
                glColor3ub(3,3,1);
                glVertex3f(-10,-10,-5);
    glEnd();
    glColor3f(1,1,1);
    int dir;
    int i;
    float j;
    glPointSize(1.5);
```

```
    glEnable(GL_POINT_SMOOTH);
    glBegin(GL_POINTS);
                for(i=0; i<5; i++)
                { glVertex2f(-5 + 10*(rand()%100 / 100.0), 3*(rand()%100 / 100.0))}
    glEnd();
    glDisable(GL_POINT_SMOOTH);}

/****** Functions for background grass******/

void drawGrass()
{   GLubyte lightGreenColor[] = {37, 186, 7};
    GLubyte darkGreenColor[] = {20, 71, 1};
    float beginX = -5;
    float beginY = -0.5;
    glBegin(GL_TRIANGLES) ;
    while(beginX<5){
     glColor3ubv(lightGreenColor);
       glVertex2f(beginX, beginY);
       glColor3ubv(darkGreenColor);
       glVertex3f(beginX - 0.025, beginY - 0.5,0);
       glVertex3f(beginX + 0.025, beginY - 0.5,0);
       beginX += 0.05;}
    glEnd();}


void drawFence()
{ float beginX = -5;
    float beginY = -0.3;
    while(beginX < 5)
    { glPushMatrix();/****** Functions for drawing fence background******/

       glTranslatef(beginX, beginY, -1);
       glScalef(0.3,0.3,0.2);
       drawUnitFence();
       glPopMatrix();
       beginX += 0.4;}}

void drawBackground()
{    glDisable(GL_LIGHTING);
    fillLowerBG();
    fillUpperBG();
    glPushMatrix();
        glScalef(1.1,1.2,1.1);
        glTranslatef(0,0.1,-1.2);
        drawGrass();
    glPopMatrix();
    drawFence();
    glEnable(GL_LIGHTING);}
```

```
/****** Functions for title******/

void titleBelow()
{    pushUp = -2.3;
     glDisable(GL_LIGHTING);
     glEnable(GL_TEXTURE_2D);
     glEnable(GL_BLEND);
     glBlendFunc (GL_ONE, GL_ONE);
titleImg = LoadTextureRAW("hdtitle.raw",300,156);
     glBindTexture (GL_TEXTURE_2D, titleImg);
     if(pushBelow >= -2.3)         //IDEAL -2.42
               pushBelow -= 0.1;
     glPushMatrix();
        glTranslatef(0,pushBelow,0.1);
        glScalef(1.7,1.7,1);
        glBegin(GL_QUADS);
                        glTexCoord3f(-1,1,0.3);
                        glVertex3f(-1,1,0.2);
                        glTexCoord3f(0,1,0.3);
                        glVertex3f(1,1,0.2);
                        glTexCoord3f(0,0,0.3);
                        glVertex3f(1,0,0.2);
                        glTexCoord3f(-1,0,0.3);
                        glVertex3f(-1,0,0.2);
        glEnd();
     glPopMatrix();
     glDisable(GL_TEXTURE_2D);
     glDisable(GL_BLEND);
     glEnable(GL_LIGHTING);}


void titleOnTop()
{
     pushBelow = 0;
     glDisable(GL_LIGHTING);
     glEnable(GL_TEXTURE_2D);
     glEnable(GL_BLEND);
     glBlendFunc (GL_ONE, GL_ONE);
titleImg = LoadTextureRAW("hdtitle.raw",300,156);
     glBindTexture (GL_TEXTURE_2D, titleImg);
     if(pushUp <= 0 )         //IDEAL 0
               pushUp += 0.1;
     glPushMatrix();
        glTranslatef(0,pushUp,0.1);
        glScalef(1.7,1.7,1);
        glBegin(GL_QUADS);
                        glTexCoord3f(-1,1,0.3);
```

```
                    glVertex3f(-1,1,0.2);
                    glTexCoord3f(0,1,0.3);
                    glVertex3f(1,1,0.2);
                    glTexCoord3f(0,0,0.3);
                    glVertex3f(1,0,0.2);
                    glTexCoord3f(-1,0,0.3);
                    glVertex3f(-1,0,0.2);
       glEnd();
    glPopMatrix();
    glDisable(GL_TEXTURE_2D);
    glDisable(GL_BLEND);
    glEnable(GL_LIGHTING);}
```

/**** Functions for creating menus for crackers and greetings******/**

```
void displayMenu()
{    if(playThemeMusic)
    {           PlaySound(TEXT("theme.wav"), NULL, SND_ASYNC|SND_FILENAME|SND_LOOP);
                playThemeMusic = 0; }
    glutSetCursor(GLUT_CURSOR_INHERIT);
    drawBackground();
    //LEFT MENU
   glPushMatrix();
            GLfloat left_diffuse[] = {1,0.16,0.16};
            GLfloat left_ambient[] = {1,0.68,0.68};
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, left_diffuse);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, left_ambient);
            glTranslatef(-1.75,2.4,0);
            glScalef(3.4,0.4,1);
            glutSolidCube(1);
   glPopMatrix();
   //RIGHT MENU
   glPushMatrix();
            GLfloat right_diffuse[] = {0.17,0.74,0.27};
            GLfloat right_ambient[] = {0.78,1,0.68};
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, right_diffuse);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, right_ambient);
            glTranslatef(1.75,2.4,0);
            glScalef(3.4,0.4,1);
            glutSolidCube(1);
   glPopMatrix();
   //EXIT BUTTON
   glPushMatrix();
            GLfloat exit_diffuse[] = {0,0.06,0.4};     //INITIAL : {0,0.06,0.4};
            GLfloat exit_ambient[] = {1,0.99,0.53};
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, exit_diffuse);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, exit_ambient);
            glTranslatef(0,-2.8,-0.01);
```

```
            glScalef(1,0.3,0.2);
            glutSolidCube(1);
    glPopMatrix();

    if(brickColor>=1)
             brickDir = 1;
    else if(brickColor <= 0)
             brickDir = 0;
    if(brickDir==0)
            brickColor+=0.05;
    else
            brickColor-=0.05;
    GLfloat details_diffuse[] = {0,0.16,brickColor};
    GLfloat details_ambient[] = {1,0.99,0.53};
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, details_diffuse);
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, details_ambient);
    glDisable(GL_LIGHTING);
    glColor3ub(255,255,255);
    glRasterPos3f(-1.7,1.6,2);
    printString("BURST SOME CRACKERS");
   glColor3ub(0,34,6);
    glRasterPos3f(0.5,1.6,2);
    printString("SEND SOME GREETINGS");
    glColor3f(1,1,1);
    renderBitmapString(-0.1,-2.57,0.5,(void *)smallFont,"Exit");
    glEnable(GL_LIGHTING);}

/****** Functions for left menu for crackers******/

void leftMenu()
{   //LEFT POP DOWN
    glPushMatrix();
            GLfloat left_down_diffuse[] = {0,0.12,0.35};
            GLfloat left_down_ambient[] = {0.62,0.75,1};
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, left_down_diffuse);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, left_down_ambient);
            glTranslatef(-1.75,1,0);
            glScalef(3.4,3.5,1);
            glutSolidCube(1);
    glPopMatrix();
    glPushMatrix();
            glScalef(0.5,0.5,0.5);
            glTranslatef(-1.3,1,3);
            rocketicon();
            glTranslatef(-3,0,0);
            flowerpoticon();
            glTranslatef(1.5,2,0);
            chakriicon();
```

```
   glPopMatrix();}

/****** Functions for right menu for greetings******/

void rightMenu()
{//RIGHT POP DOWN
   glPushMatrix();
            GLfloat right_down_diffuse[] = {0,0.12,0.35};
            GLfloat right_down_ambient[] = {0.62,0.75,1};
            glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, right_down_diffuse);
            glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, right_down_ambient);
            glTranslatef(1.75,1.4,0);
            glScalef(3.4,2.0,1);
            glutSolidCube(1);
   glPopMatrix();
   glPushMatrix();
            glScalef(0.5,0.5,0.5);
            glTranslatef(1.5,2.8,3);
            greeting1icon();
            glTranslatef(2.5,0,0);
            greeting2icon();
   glPopMatrix();}

/****** Functions for sparkler******/

void sparkler()
{       glDisable(GL_LIGHTING);
        glPushMatrix();
        glTranslatef(passiveX,passiveY,3);
        glRotatef(75,0,0,1);
        glScalef(0.1,0.2,1);
        glBegin(GL_POLYGON);
                    glColor3ub(3,42,108);
                    glVertex2f(-0.1,0);
                    glColor3ub(255,255,255);
                    glVertex2f(0,0.2);
                    glColor3ub(3,42,108);
                    glVertex2f(0.1,0);
                    glColor3ub(0,0,0);
                    glVertex2f(0.1,-2);
                    glVertex2f(-0.1,-2);
        glEnd();
        glBegin(GL_POLYGON);
                    glColor3ub(83,28,3);
                    glVertex2f(-0.01,-2);
                    glVertex2f(0.01,-2);
                    glColor3f(1,1,1);
```

```
                    glVertex2f(0.01,-3);
                    glVertex2f(-0.01,-3);
        glEnd();
        glPopMatrix();;
        glEnable(GL_LIGHTING);}

/****** Functions for chakra cracker******/

void steadychakri()
{    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();
   backButton();
   glDisable(GL_LIGHTING);
    glBegin(GL_POLYGON);
                glColor3ub(12,32,2);
                glVertex3f(-50,50,-50);
                glColor3ub(0,0,0);
                glVertex3f(50,50,-50);
                glColor3ub(32,77,10);
                glVertex3f(50,-50,-50);
                glColor3ub(0,0,0);
                glVertex3f(-50,-50,-50);
    glEnd();
    glEnable(GL_LIGHTING);
    glPushMatrix();
    glRotatef(-45,1,0,0);
                glPushMatrix();
                glScalef(0.2,0.2,0.2);
                //TORUS
                GLfloat torus_diffuse[] = {0.8,0,0};
                glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, torus_diffuse);
                glPushMatrix();
                glScalef(0.25,0.25,0.25);
                glutWireTorus(1,10,15,30);
                glPopMatrix();
                //LIGHTING DISABLED
                glDisable(GL_LIGHTING);
                //CHAKRI CENTER
                 float radius=2.4;
                 glBegin(GL_POLYGON);
                 glColor3f(0,0,0);
                 glVertex2f(0,0);
                 glColor3f(1,0.54,0);
                 int i;
                 float degInRad;
                for (i=0; i <= 361; i++)
                {degInRad = i*3.14/180;
                   glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0);   }
```

```
                glEnd(); //GREEN LINES
                glColor3f(0,1,0);
                radius=2.0;
                 glBegin(GL_LINE_LOOP);
                for (i=0; i <= 361; i++)
                {  degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.02) ;}
                glEnd();
                radius=1.8;
                 glBegin(GL_LINE_LOOP);
                for (i=0; i <= 361; i++)
                {   degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.02);  }
                glEnd();
                radius=0.7;
                 glBegin(GL_POLYGON);
                 glColor3f(0,0.7,0);
                 glVertex2f(0,0);
                 glColor3f(0,0.3,0);
                for (i=0; i <= 361; i++)
                {  degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.03);}
                glEnd();
                //RED CENTRE
                radius=1.2;
                 glBegin(GL_POLYGON);
                 glColor3f(0.5,0,0);
                 glVertex2f(0,0);
                 glColor3f(1,0,0);
                for (i=0; i <= 361; i++)
                {  degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),0.02);}
                glEnd();
                 glPopMatrix();
                  glEnable(GL_LIGHTING);
    glPopMatrix();}

void chakri()
{
    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();
    backButton();
    glDisable(GL_LIGHTING);
    glBegin(GL_POLYGON);
                 glColor3ub(12,32,2);
                 glVertex3f(-50,50,-50);
                 glColor3ub(0,0,0);
                 glVertex3f(50,50,-50);
```

```
                glColor3ub(32,77,10);
                glVertex3f(50,-50,-50);
                glColor3ub(0,0,0);
                glVertex3f(-50,-50,-50);
glEnd();
glEnable(GL_LIGHTING);
glPushMatrix();
glRotatef(-45,1,0,0);
                int dir = 0;
                glPushMatrix();
                glScalef(0.2,0.2,0.2);
                glTranslatef(chakri_axes[0],chakri_axes[1],chakri_axes[2]);
               //TORUS
               GLfloat torus_diffuse[] = {0.8,0,0};
               glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, torus_diffuse);
               glPushMatrix();
               glScalef(0.25,0.25,0.25);
               glutWireTorus(1,10,15,30);
               glPopMatrix();
               //LIGHTING DISABLED
               glDisable(GL_LIGHTING);
               //CHAKRI CENTER
                float radius=2.5;
                glBegin(GL_POLYGON);
                glColor3f(0,0,0);
                glVertex2f(0,0);
                glColor3f(1,0.54,0);
                int i;
                float degInRad;
               for (i=0; i <= 361; i++)
               {   degInRad = i*3.14/180;
                 glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0); }
               glEnd();
               //GREEN LINES
               glColor3f(0,1,0);
               radius=2.0;
                glBegin(GL_LINE_LOOP);
               for (i=0; i <= 361; i++)
               {   degInRad = i*3.14/180;
                 glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.02);}
               glEnd();
               radius=1.8;
                glBegin(GL_LINE_LOOP);
               for (i=0; i <= 361; i++)
               {  degInRad = i*3.14/180;
                 glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.02);}
               glEnd();
```

```
                radius=0.7;
                 glBegin(GL_POLYGON);
                 glColor3f(0,0.7,0);
                 glVertex2f(0,0);
                 glColor3f(0,0.3,0);
                for (i=0; i <= 361; i++)
                {   degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.03);}
                glEnd();
                //RED CENTRE
               radius=1.2;
                 glBegin(GL_POLYGON);
                 glColor3f(0.5,0,0);
                 glVertex2f(0,0);
                 glColor3f(1,0,0);
                for (i=0; i <= 361; i++)
                {degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),0.02); }
                glEnd();
                 glFermatSpiral(0.05, 0.5, cangle+1, cangle+75.0, 80);
                 glPopMatrix();
                 cangle -= 0.1;
                 dir = rand()%2;
                 int incAxis = rand()%3;
                 if(dir)
                 {if(chakri_axes[incAxis] < 4)
                        chakri_axes[incAxis]++;                    }
                 else
                 {  if(chakri_axes[incAxis] > -4)
                        chakri_axes[incAxis]--; }
                  glEnable(GL_LIGHTING);
    glPopMatrix();}
```

/**** Functions for flowerpot cracker******/**

```
void steadyflowerpot() {
    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();
    drawBackground();
    GLfloat flowerpot_diffuse[] = {0.02,0,1};
    GLfloat flowerpot_ambient[] = {1,0,1};
    GLfloat flowerpot_shininess[] = { 100 };
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, flowerpot_diffuse);
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, flowerpot_ambient);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, flowerpot_shininess);
    backButton();
    glPushMatrix();
       glRotatef(-100.0, 1,0,0);
```

```
        glTranslatef(0,0,-2);
        glutSolidCone(0.2,0.5,20,20);
    glPopMatrix();}

void flowerpot() {
    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();
    drawBackground();
    GLfloat flowerpot_diffuse[] = {0.02,0,1};
    GLfloat flowerpot_ambient[] = {1,0,1};
    GLfloat flowerpot_shininess[] = { 100 };
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, flowerpot_diffuse);
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, flowerpot_ambient);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, flowerpot_shininess);
    backButton();
    glPushMatrix();
        glRotatef(-100.0, 1,0,0);
        glTranslatef(0,0,-2);
        glutSolidCone(0.2,0.5,20,20);
    glPopMatrix();
    glPushMatrix();
        glTranslatef(0,1,0);
        fireworks();
    glPopMatrix();}

void fireworks(void) {
    glEnable(GL_TEXTURE_2D);
    GLfloat sparks_diffuse[] = {0.2,0.5,1};
    GLfloat sparks_ambient[] = {1,1,0};
    GLfloat sparks_shininess[] = { 100 };
    GLfloat temp_ambient[] = {1,1,1};
    glLightfv (GL_LIGHT0, GL_AMBIENT, temp_ambient);
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, sparks_diffuse);
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, sparks_ambient);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, sparks_shininess);
    glUpdateParticles();
      glPushMatrix();
            glScalef(0.7,0.9,1);
            glTranslatef(0,0.7,0);
            glUpdateParticles();
            glDrawParticles();
      glPopMatrix();
     glDisable(GL_TEXTURE_2D);}

/****** Functions for rocket cracker******/

void steadyrocket(){
    playBlastMusic = 1;
```

```
    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();
    drawBackground();
    glPushMatrix();
    glScalef(0.6,0.6,0.6);
                GLfloat cyl_red[] = {1,0,0};
                GLfloat cyl_yellow[] = {1,1,0};
                GLfloat cyl_dark_red[] = {0.4,0,0};
                GLfloat cyl_dark_yellow[] = {0.5,0.5,0};
                GLfloat cyl_white[] = {0.8,0.8,0.8};
                GLfloat rocket_ambient[] = {0.2,0.2,0.2};
            glMaterialfv (GL_FRONT_AND_BACK, GL_AMBIENT, rocket_ambient);
            glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_white);
            glPushMatrix();
                    glTranslatef(-2,-2,0);          //-1 0 1
                    glRotatef(90,1,0,0);
                    gluCylinder(qobj, 0.4, 0.4, 1.4, 15, 16);
            glPopMatrix();
            glPushMatrix();
              glTranslatef(-2,-1,0);                //-1 0 0
              glRotatef(-1*rocketangle, 1,0,1);
              glPushMatrix();
                    glRotatef(90,1,0,0);
                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_red);
                    gluCylinder(qobj, 0.2, 0.2, 1.0, 15, 16);
                    glPushMatrix();
                            glTranslatef(0,0,0.22);
                            glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_dark_yellow);
                            gluCylinder(qobj, 0.21, 0.21, 0.5, 15, 16);
                    glPopMatrix();
                    glRotatef(180,1,0,0);
                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_dark_red);
                    glutSolidCone(0.3,0.5,20,20);
                    glTranslatef(-0.18,0,-2);

                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_yellow);
                    gluCylinder(qobj, 0.01, 0.01, 1.0, 15, 16);
              glPopMatrix();
            glPopMatrix();
    glPopMatrix();
    backButton();}

void rocketBlinkScreen()
{
    if(playBlastMusic)
    {
       PlaySound(TEXT("blast.wav"), NULL, SND_ASYNC|SND_FILENAME);
       playBlastMusic = 0;  }
```

```
    sparkler();
    backButton();
    glDisable(GL_LIGHTING);
    glColor3ub(rand()%255,rand()%255,rand()%255);
    glBegin(GL_QUADS);
                glVertex3f(-6,4,0);
                glVertex3f(6,4,0);
                glVertex3f(6,-4,0);
                glVertex3f(-6,-4,0);
    glEnd();
    glEnable(GL_LIGHTING);
    blinkTime += 0.1;
    if(blinkTime >= 10)
    { status = DISPLAY_MENU;
          resetMaterialProperties();
          glDisable(GL_TEXTURE_2D);
          rocketx = -2;
          rockety = -1;
          rocketz = 0;
          rocketangle = 0;
          glutSetCursor(GLUT_CURSOR_INHERIT); }}

void rocket()
{
    glutSetCursor(GLUT_CURSOR_NONE);
    sparkler();

    drawBackground();

    GLfloat rocket_ambient[] = {0.2,0.2,0.2};
    glMaterialfv (GL_FRONT_AND_BACK, GL_AMBIENT, rocket_ambient);

    glPushMatrix();
    glScalef(0.6,0.6,0.6);
                GLfloat cyl_red[] = {1,0,0};
                GLfloat cyl_yellow[] = {1,1,0};
                GLfloat cyl_dark_red[] = {0.4,0,0};
                GLfloat cyl_dark_yellow[] = {0.5,0.5,0};
                GLfloat cyl_white[] = {0.8,0.8,0.8};

              glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_white);
              glPushMatrix();
                      glTranslatef(-2,-2,0);
                      glRotatef(90,1,0,0);
                      gluCylinder(qobj, 0.4, 0.4, 1.4, 15, 16);
              glPopMatrix();
              glPushMatrix();
                glTranslatef(rocketx,rockety, rocketz);
```

```
                glRotatef(-1*rocketangle, 1,0,1);
                glPushMatrix();
                    glRotatef(90,1,0,0);
                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_red);
                    gluCylinder(qobj, 0.2, 0.2, 1.0, 15, 16);
                    glPushMatrix();
                        glTranslatef(0,0,0.22);
                        glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_dark_yellow);
                        gluCylinder(qobj, 0.21, 0.21, 0.5, 15, 16);
                    glPopMatrix();
                    glRotatef(180,1,0,0);
                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_dark_red);
                    glutSolidCone(0.3,0.5,20,20);
                    glTranslatef(-0.18,0,-2);
                    glMaterialfv (GL_FRONT_AND_BACK, GL_DIFFUSE, cyl_yellow);
                    gluCylinder(qobj, 0.01, 0.01, 1.0, 15, 16);
                glPopMatrix();
            glPopMatrix();
               rocketx += 0.1;
               rockety += 0.4;
               rocketz -= 0.2;
               if(rocketangle < 90)
                  rocketangle += 1;
    glPopMatrix();
    backButton();
    if(rockety >= 10)
            status=ROCKETBLINK;}

void rocketicon()
{glDisable(GL_LIGHTING);
       int i;
         float degInRad;
         float radius = 0.9;
         glPushMatrix();
         glTranslatef(0,-0.8,0);
            glBegin(GL_POLYGON);
            glColor3f(1,1,1);
            glVertex2f(0,0);
            glColor3ub(255,253,87);
           for (i=0; i <= 361; i++)
           {degInRad = i*3.14/180;
             glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),-0.01); }
           glEnd();
         glPopMatrix();
         glBegin(GL_POLYGON);
                    glColor3ub(221,0,0);
                    glVertex2f(0,0);
                    glColor3ub(228,78,78);
```

```
                glVertex2f(-0.2,-0.4);
                glVertex2f(0.2,-0.4);
        glEnd();
        glBegin(GL_POLYGON);
                glColor3ub(162,76,0);
                glVertex2f(-0.15,-0.4);
                glVertex2f(0.15,-0.4);
                glColor3ub(247,117,0);
                glVertex2f(0.15,-1);
                glVertex2f(-0.15,-1);
        glEnd();
        glBegin(GL_LINES);
                glColor3f(0,0,0);
                glVertex2f(-0.15,-1);
                glColor3f(1,1,0);
                glVertex2f(-0.15,-1.5);
        glEnd();
        glEnable(GL_LIGHTING);}
```

/**** Functions for left menu for crackers icon******/**

```
void flowerpoticon()
{glDisable(GL_LIGHTING);
        int i;
        float degInRad;
        float radius = 0.9;
        glPushMatrix();
        glTranslatef(0,-0.8,0);
           glBegin(GL_POLYGON);
           glColor3f(1,1,1);
           glVertex2f(0,0);
           glColor3ub(255,253,87);
          for (i=0; i <= 361; i++)
          {degInRad = i*3.14/180;
            glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),-0.01); }
          glEnd();
        glPopMatrix();
        glBegin(GL_POLYGON);
                glColor3ub(25, 42, 119);
                glVertex2f(0,0);
                glColor3ub(0,48,255);
                glVertex2f(-0.5,-1.3);
                glColor3ub(60,97,255);
                glVertex2f(0.5,-1.3);
        glEnd();
        glEnable(GL_LIGHTING);}

void chakriicon(){
```

```
        glDisable(GL_LIGHTING);
        int i;
        float degInRad;
        float radius = 0.9;
        glPushMatrix();
        glTranslatef(0,-0.8,0);
           glBegin(GL_POLYGON);
           glColor3f(1,1,1);
           glVertex2f(0,0);
           glColor3ub(255,253,87);
          for (i=0; i <= 361; i++)
          {degInRad = i*3.14/180;
            glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),-0.01); }
          glEnd();
          radius = 0.5;
          glBegin(GL_POLYGON);
           glColor3ub(255,84,0);
           glVertex2f(0,0);
           glColor3ub(148,30,2);
          for (i=0; i <= 361; i++)
          { degInRad = i*3.14/180;
            glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),0.00); }
          glEnd();
          radius = 0.2;
          glBegin(GL_POLYGON);
           glColor3ub(40,163,16);
           glVertex2f(0,0);
           glColor3ub(11,66,0);
          for (i=0; i <= 361; i++)
          {degInRad = i*3.14/180;
            glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),0.01); }
          glEnd();
        glPopMatrix();
        glEnable(GL_LIGHTING);}
```

/**** Functions for right menu for greeting ******/**

```
void diya()
{    int i;
    float radius;
   float degInRad;
   float centerX,centerY;
       radius=0.6;
                centerX=-1.5, centerY=0;
                 glBegin(GL_POLYGON);
                 glColor3ub(21,6,1);
                 glVertex3f(centerX,centerY,0.1);
                 glColor3ub(91,35,5);
```

```
                for (i=180; i <= 360; i++)
                {degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius +centerX,sin(degInRad)*radius+centerY,0.1); }
                glEnd();
        glPushMatrix();
                 glScalef(0.2,0.7,1);
                 radius=0.6;
                 centerX= -4.5, centerY=0.7;
                 glBegin(GL_POLYGON);
                 glColor3ub(255,248,0);
                 glVertex3f(centerX,centerY,0.1);
                 glColor3ub(255,127,0);
                 degInRad;
                for (i=0; i <= 361; i++)
                {degInRad = i*3.14/180;
                  glVertex3f(cos(degInRad)*radius +centerX,sin(degInRad)*radius+centerY,0.1);}
                glEnd();
        glPopMatrix();}

/****** Functions for greetings******/

void greeting1()
{ backButton();
   int i;
   float diyaRotationRadius=0.5;
   float radius;
   float degInRad;
    glDisable(GL_LIGHTING);
    glBegin(GL_QUADS);
                glColor3f(1,1,1);
                glVertex3f(-5,5,-0.2);
                glColor3f(50,1,1);
                glVertex3f(5,5,-0.2);
                glColor3ub(colRed[0],colRed[0],colRed[0]);
                glVertex3f(5,-5,-0.2);
                glColor3f(0,0.5,0.5);
                glVertex3f(-5,-5,-0.2);
    glEnd();
    glBegin(GL_QUADS);
                glColor3ub(colRed[0],0,0);
                glVertex3f(0,2,0);
                glColor3ub(colRed[1],1,0);
                glVertex3f(3,2,0);
                glColor3ub(colRed[2],3,1);
                glVertex3f(3,-2,0);
                glColor3ub(colRed[3],2,1);
                glVertex3f(0,-2,0);
    glEnd();
```

```
        //TEXT
        glColor3ub(255,200,0);
        renderBitmapString(0.8,1,0.1,(void *)bigFont,"Happy Diwali");
        glColor3ub(255,201,106);
        renderBitmapString(0.4,0.5,0.1,(void *)smallFont,"With a hope that you attain");
        renderBitmapString(0.7,0.25,0.1,(void *)smallFont,"Success and Bliss,");
        renderBitmapString(0.4,0,0.1,(void *)smallFont,"With every light that is lit");
        renderBitmapString(0.62,-0.25,0.1,(void *)smallFont,"on the Day of Diwali.");
        glColor3ub(225,82,54);
        renderBitmapString(0.85,-0.75,0.1,(void *)smallFont,"Best Regards,");
        glColor3ub(226,74,0);
        if(status==GREETING1)
        {       renderBitmapString(0.6,-0.95,0.1,(void *)smallFont,"Raksha,Swathi,Susmitha");}
        else
        {renderBitmapString(1.1,-0.95,0.1,(void *)smallFont,name1); }
        glColor3f(0.3,0,0);
        glBegin(GL_LINES);
                glVertex3f(0,2,0.01);
                glVertex3f(0,-2,0.01);
        glEnd();
       glPushMatrix();
          glRotatef(openAngle1,0,1,0);
          glBegin(GL_QUADS);
          glColor3f(1,1,1);
                glColor3ub(colRed[0],0,0);
                glVertex3f(0,2,0);
                glColor3ub(colRed[1],1,0);
                glVertex3f(-3,2,0);
                glColor3ub(colRed[2],3,1);
                glVertex3f(-3,-2,0);
                glColor3ub(colRed[3],2,1);
                glVertex3f(0,-2,0);
          glEnd();
              glPushMatrix();
                glTranslatef(cos(degInRad)*diyaRotationRadius,sin(degInRad)*diyaRotationRadius,0);
                diya();
                diyaRotator -= 5;
                if(diyaRotator <= 0)
                        diyaRotator = 361;
              glPopMatrix();
        glPopMatrix();
if(openAngle1>30)
                openAngle1 -= 3;
        for(i=0;i<4;i++)
        {
     if(cDir[i]==0)
                colRed[i] += 10;
```

```
                else
                  colRed[i] -= 10;
                if(colRed[i] >= 255)
                {cDir[i] = 1;
                  colRed[i] = 255;  }
                else if(colRed[i] <= 0)
                { cDir[i] = 0;
                  colRed[i] = 0; } }
    glEnable(GL_LIGHTING);}

/****** Functions for greetings icon******/

void greeting1icon()
{glDisable(GL_LIGHTING);
 int i;
         float degInRad;
         float radius = 0.9;
         glPushMatrix();
         glTranslatef(0,-0.8,0);
            glBegin(GL_POLYGON);
            glColor3f(1,1,1);
            glVertex2f(0,0);
            glColor3ub(255,253,87);
           for (i=0; i <= 361; i++)
           {degInRad = i*3.14/180;
             glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),-0.01); }
           glEnd();
           glScalef(0.7,0.7,0.7);
           glTranslatef(1.5,-0.2,0);
           diya();
         glPopMatrix();
         glEnable(GL_LIGHTING);}

void sideLight(int color){
int i;
    float radius;
    float degInRad;
   GLubyte red[]  = { 255, 0, 0};
   GLubyte green[]  = {   0, 255,   0};
   GLubyte blue[]  = { 0,  0, 255};
   glPushMatrix();
       radius=0.6;
       glScalef(0.07,0.3,1);
       glBegin(GL_POLYGON);
          switch(color)
          {  case 0:
                        glColor3ubv(red);
                        break;
```

```
                    case 1:
                        glColor3ubv(green);
                        break;
                    case 2:
                        glColor3ubv(blue);
                        break;
                    default:
                        glColor3ubv(red);
                        break; }
            glVertex3f(0,0,0.1);
            glColor3ub(0,0,0);
            for (i=0; i <= 361; i++)
            {    degInRad = i*3.14/180;
                glVertex3f(cos(degInRad)*radius,sin(degInRad)*radius,0.1);}
            glEnd();
    glPopMatrix();}

void greeting2img()
{  int i;
    glDisable(GL_LIGHTING);
    glEnable(GL_TEXTURE_2D);
    ganpati = LoadTextureRAW("ganpati.raw",400,400);
    glBindTexture (GL_TEXTURE_2D, ganpati);
    glPushMatrix();
      glTranslatef(1.2,0,0);
      glScalef(1.5,1.5,1);
      glBegin(GL_POLYGON);
                    glTexCoord3f(-1,1,0.3);
                    glVertex3f(-1,1,0.2);
                    glTexCoord3f(0,1,0.3);
                    glVertex3f(1,1,0.2);
                    glTexCoord3f(0,0,0.3);
                    glVertex3f(1,-1,0.2);
                    glTexCoord3f(-1,0,0.3);
                    glVertex3f(-1,-1,0.2);
      glEnd();
    glPopMatrix();
    happydiwali = LoadTextureRAW("happydiwali.raw",300,300);
    glBindTexture( GL_TEXTURE_2D,happydiwali );
    glPushMatrix();
      glTranslatef(-1.55,0,0);
      glScalef(1.3,1.3,1);
      glBegin(GL_POLYGON);
                    glTexCoord3f(-1,1,0.3);
                    glVertex3f(-1,1,0.2);
                    glTexCoord3f(0,1,0.3);
                    glVertex3f(1,1,0.2);
                    glTexCoord3f(0,0,0.3);
```

```
                    glVertex3f(1,-1,0.2);
                    glTexCoord3f(-1,0,0.3);
                    glVertex3f(-1,-1,0.2);
        glEnd();
    glPopMatrix();
    glEnable(GL_LIGHTING);
    glDisable(GL_TEXTURE_2D);}

void greeting2()
{
    backButton();
    greeting2img();
    float i;
    int toggler = 0;
    glDisable(GL_LIGHTING);
        //BACKGROUND
            glBegin(GL_QUADS);
                glColor3f(0,0,0);
                glVertex3f(-5,5,-0.2);
                glColor3f(50,50,1);
                glVertex3f(5,5,-0.2);
                glColor3ub(153,153,255);
                glVertex3f(5,-5,-0.2);
                glColor3f(0,0.5,0.5);
                glVertex3f(-5,-5,-0.2);
           glEnd();
        glBegin(GL_POLYGON);
             glColor3ub(252,255,5);
             glVertex3f(-3.2,2.5,0);
             glColor3ub(248,228,0);
             glVertex3f(3.2,2.5,0);
             glColor3ub(68,63,2);
             glVertex3f(3.2,-2.5,0);
             glColor3ub(68,63,2);
             glVertex3f(-3.2,-2.5,0);
        glEnd();
        glPushMatrix();
        glTranslatef(0,2.4,0);
        glRotatef(triangleAngle2,1,0,0);
  glBegin(GL_TRIANGLES);
             for(i=-3.2; i<=3.2; i=i+0.4 )
             {glColor3ub(107,3,3);
                glVertex3f(i,0,0.01);
                glVertex3f(i+0.4,0,0.01);
                if(toggler)
                {glColor3ub(255,0,0);
                   toggler = 0; }
                else
```

```
            {glColor3ub(255,96,0);
               toggler = 1; }
            glVertex3f(i+0.2,-0.6,0.01); }
      glEnd();
   glPopMatrix();
   //LIGHTS
   int colorIterator = startColor;
   int numLights;
   //LEFT
   glPushMatrix();
            glTranslatef(-3,1.4,0);
            for(numLights = 11; numLights>0; numLights-=1)
            { colorIterator=(colorIterator + 1)%3;
                     sideLight(colorIterator);
                     glTranslatef(0,-0.35,0); }
   glPopMatrix();
   //RIGHT
   glPushMatrix();
            glTranslatef(3,1.4,0);
            for(numLights = 11; numLights>0; numLights-=1)
            {colorIterator=(colorIterator + 1)%3;
                     sideLight(colorIterator);
                     glTranslatef(0,-0.35,0);}
   glPopMatrix();
   //BOTTOM
   glPushMatrix();
            glTranslatef(-2.8,-2.3,0);
            glRotatef(90,0,0,1);
            for(numLights = 17; numLights>0; numLights-=1)
            {colorIterator=(colorIterator + 1)%3;
                     sideLight(colorIterator);
                     glTranslatef(0,-0.35,0);    }
   glPopMatrix();
   startColor = (startColor + 1)%3;
   if(triangleAngle2>=0)
       dir = 1;
    else if(triangleAngle2<=-45)
       dir = 0;
    if(dir==0)
       triangleAngle2 += 2;
    else
       triangleAngle2 -= 2;
//TEXT
glColor3ub(255,255,255);
if(status==GREETING2)
{  renderBitmapString(-1.4,-1.8,0.1,(void *)bigFont,"Swathi,Raksha,Sushmitha");}
else
{       glRasterPos3f(-1,-1.8,0.1);
```

```
        renderBitmapString(-2,-1.8,0.1,(void *)bigFont,name2)}
    glEnable(GL_LIGHTING);}

void greeting2icon()
{sleepCount += 1;
         glDisable(GL_LIGHTING);
         if(sleepCount >= 10)
         {iconColorIter = (iconColorIter+1)%3;
                  sleepCount = 0; }
        int i;
         float degInRad;
         float radius = 0.9;
         glPushMatrix();
         glTranslatef(0,-0.8,0);
            glBegin(GL_POLYGON);
            glColor3f(1,1,1);
            glVertex2f(0,0);
            glColor3ub(255,253,87);
           for (i=0; i <= 361; i++)
           { degInRad = i*3.14/180;
             glVertex3f(cos(degInRad)*(1.0*radius),sin(degInRad)*(1.0*radius),-0.01); }
           glEnd();
         glTranslatef(-0.05,-0.05,0);
         glScalef(14.28,3.33,1);
         sideLight(iconColorIter);
         glPopMatrix();
         glEnable(GL_LIGHTING);}

/****** Functions for display******/
void displayd (void) {
   glDisable( GL_TEXTURE_2D );
   glClearColor (0.0,0.0,0.0,1.0); //clear the screen to black
   glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
 //clear the color buffer and the depth buffer
   glLoadIdentity();
   GLfloat DiffuseLight[] = {1,1,1}; //set DiffuseLight[] to the specified values
   GLfloat AmbientLight[] = {0,0,0}; //set AmbientLight[] to the specified values
   glLightfv (GL_LIGHT0, GL_DIFFUSE, DiffuseLight); //change the light accordingly
   glLightfv (GL_LIGHT0, GL_AMBIENT, AmbientLight); //change the light accordingly
   GLfloat LightPosition[] = {lx, ly, lz, lw}; //set the LightPosition to the specified values
   glLightfv (GL_LIGHT0, GL_POSITION, LightPosition); //change the light accordingly
   gluLookAt (0.0, 0.0, 5.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0);
//camera position, x,y,z, looking at x,y,z, Up Positions of the camera
   if(status==DISPLAY_MENU)
   {displayMenu();
      titleOnTop(); }
   else if(status==LEFT_MENU)
   {       displayMenu();
```

```
        titleBelow();
        leftMenu();    }
    else if(status==RIGHT_MENU)
    {  displayMenu();
        titleBelow();
        rightMenu();}
    else if(status==CHAKRI_STEADY)
    {   steadychakri();}
    else if(status==CHAKRI)
    {playThemeMusic = 1;      //When back to display, play theme music
        chakri();  }
    else if(status==FLOWERPOT_STEADY)
    { steadyflowerpot();}
    else if(status==FLOWERPOT)
    {playThemeMusic = 1;      //When back to display, play theme music
        flowerpot();}
    else if(status==ROCKET_STEADY)
    {steadyrocket();}
    else if(status==ROCKET)
    {playThemeMusic = 1;      //When back to display, play theme music
        rocket();  }
    else if(status==ROCKETBLINK)
    {rocketBlinkScreen();}
    else if(status==GREETING1 || status==GREETING1SIGNED)
    { greeting1();}
    else if(status==GREETING2 || status==GREETING2SIGNED)
    {  greeting2();  }
    glutSwapBuffers(); //swap the buffers
    if(status!=0 && status !=1 && status !=2)
        Sleep(35);}
```

/**** Functions for users to click the crackers******/**

```
void mouseClick(int button, int state, int xClick, int yClick)
{  if(button == GLUT_LEFT_BUTTON && state==GLUT_DOWN)
    {        float x, y;
             float sceneWidth, sceneHeight;
               sceneHeight = scr_height;
               sceneWidth = scr_height * 4.0/3.0;
               x = (xClick - (scr_width - sceneWidth)/2)/sceneWidth * 800.0;
               y = yClick/sceneHeight * 600.0;
             if(status==DISPLAY_MENU)
             { if(x>0 && x<390 && y>0 && y<45)
                     status = LEFT_MENU;
               else if(x>405 && x<800 && y>0 && y<45)
                     status = RIGHT_MENU;
               else if(x>350 && x<455 && y>570 && y<600)
                       exit(0);  }
```

```
else if(status==LEFT_MENU)
{if(x>125 && x<260 && y>65 && y<205)
        status = CHAKRI_STEADY;
   else if(x>15 && x<150 && y>215 && y<350)
        status = FLOWERPOT_STEADY;
   else if(x>235 && x<370 && y>215 && y<350)
        status = ROCKET_STEADY;
   else if(x>405 && x<800 && y>0 && y<45)
        status = RIGHT_MENU;
   else if(x>350 && x<455 && y>570 && y<600)
        exit(0);
   else
        status = DISPLAY_MENU;}
else if(status==RIGHT_MENU)
{if(x>0 && x<390 && y>0 && y<45)
      status = LEFT_MENU;
   else if(x>445 && x<580 && y>85 && y<220)
        status = GREETING1;
   else if(x>630 && x<763 && y>85 && y<220)
        status = GREETING2;
   else if(x>350 && x<455 && y>570 && y<600)
        exit(0);
   else
        status = DISPLAY_MENU; }
else if(status==CHAKRI_STEADY)
{  if(x>347 && x<452 && y>247 && y<353)
   {PlaySound(TEXT("chakri.wav"), NULL, SND_ASYNC|SND_FILENAME);
        status = CHAKRI;     }
   else if(x>670 && x<750 && y>570 && y<590)
   {status = DISPLAY_MENU;
         resetMaterialProperties();
         glutSetCursor(GLUT_CURSOR_INHERIT); }   }
else if(status==FLOWERPOT_STEADY)
{ if(x>375 && x<425 && y>460 && y<525)
   {PlaySound(TEXT("flowerpot.wav"), NULL, SND_ASYNC|SND_FILENAME);
        status = FLOWERPOT;
        glCreateParticles();
        texture[0] = LoadTextureRAW( "particle_mask.raw",256,256); //load our texture
        texture[1] = LoadTextureRAW( "particle.raw",256,256); }
   else if(x>670 && x<750 && y>570 && y<590)
   {status = DISPLAY_MENU;
         resetMaterialProperties();
         glutSetCursor(GLUT_CURSOR_INHERIT);
         FreeTexture(texture[0]);
         FreeTexture(texture[1]); }}
else if(status==ROCKET_STEADY)
{if(x>250 && x<300 && y>330 && y<530)
   {
```

```
            PlaySound(TEXT("whoosh.wav"), NULL, SND_ASYNC|SND_FILENAME);
            Sleep(50);
            status = ROCKET;   }
    else if(x>670 && x<750 && y>570 && y<590)
    {status = DISPLAY_MENU;
             resetMaterialProperties();
             glutSetCursor(GLUT_CURSOR_INHERIT);    }   }
else if(status==CHAKRI)
{  if(x>670 && x<750 && y>570 && y<590)
    {       status = DISPLAY_MENU;
             resetMaterialProperties();
             cangle = 50;
             chakri_axes[0] = 0;
             chakri_axes[1] = 0;
             chakri_axes[2] = 0;
             glutSetCursor(GLUT_CURSOR_INHERIT); }}
else if(status==FLOWERPOT)
{if(x>670 && x<750 && y>570 && y<590)
    {status = DISPLAY_MENU;
             resetMaterialProperties();
             glutSetCursor(GLUT_CURSOR_INHERIT);
             FreeTexture(0);
             FreeTexture(1);
             FreeTexture(2);
             GLfloat restore_ambient[] = {0,0,0};
             glLightfv (GL_LIGHT0, GL_AMBIENT, restore_ambient);}}
else if(status==ROCKET)
{  if(x>670 && x<750 && y>570 && y<590)
    {status = DISPLAY_MENU;
             resetMaterialProperties();
             glDisable(GL_TEXTURE_2D);
             rocketx = -2;
             rockety = -1;
             rocketz = 0;
             rocketangle = 0;
             glutSetCursor(GLUT_CURSOR_INHERIT);}}
 else if(status==ROCKETBLINK)
{if(x>670 && x<750 && y>570 && y<590)
    {status = DISPLAY_MENU;
             resetMaterialProperties();
             glDisable(GL_TEXTURE_2D);
             rocketx = -2;
             rockety = -1;
             rocketz = 0;
             rocketangle = 0;
             glutSetCursor(GLUT_CURSOR_INHERIT);}}
else if(status==GREETING1 || status==GREETING1SIGNED)
{            if(x>670 && x<750 && y>570 && y<590)
```

```
                {status = DISPLAY_MENU;
                        openAngle1 = 135;
                        name1[0] = '\0';
                        name1Iter = 0; }}
                else if(status==GREETING2 || status==GREETING2SIGNED)
                {if(x>670 && x<750 && y>570 && y<590) {
                        status = DISPLAY_MENU;
                        name2[0] = '\0';
                        name2Iter = 0;
                        FreeTexture(ganpati);
                        FreeTexture(happydiwali)

        }

        }

        }

        }
```

Conclusion

“Festivals of India” computer graphics project is an interactive and interesting way of conveying the culture and festivals of India. It provides information and a simulation to prove the importance of some festivals in the form of text and animations.

The animations are created using OpenGL and C programming language. Various glut functions and C functions has been invoked to create the features and movements of certain elements in the project. The use of images, texts and audio has been added using Load raw texture function, render bitmap string function and Play sound functions respectively. The projected is compiled, built and run in codeblocks IDE which provides a detailed error description if any helped us to solve any syntax, semantic, logical or runtime errors in our code.

Overall, the graphics is interactive, informative and colorful way to communicate the significance of the festivals. We implemented the following festivals: Christmas, Kumbha Mela, Guru Poornima, Makar Sankranti and Diwali.

Reference

- www.khronos.org
- www.stackoverflow.com
- www.computergraphics.stackexchange.com
- gamedev.stackexchange.com
- www.github.com
- www.graphics.stanford.edu
- www.tutorialspoint.com

Appendix

Fig.1. Introduction page

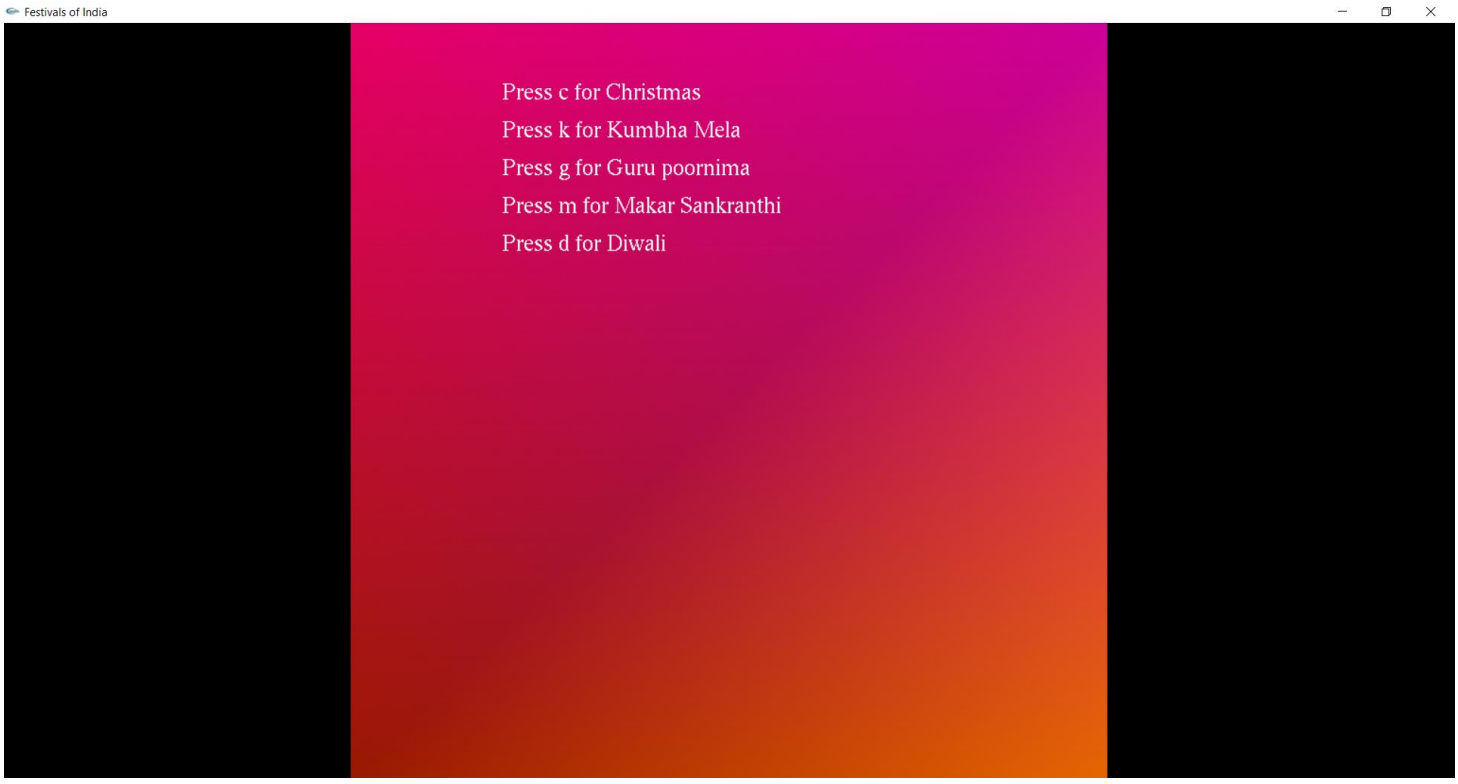

Fig.2. Instruction page

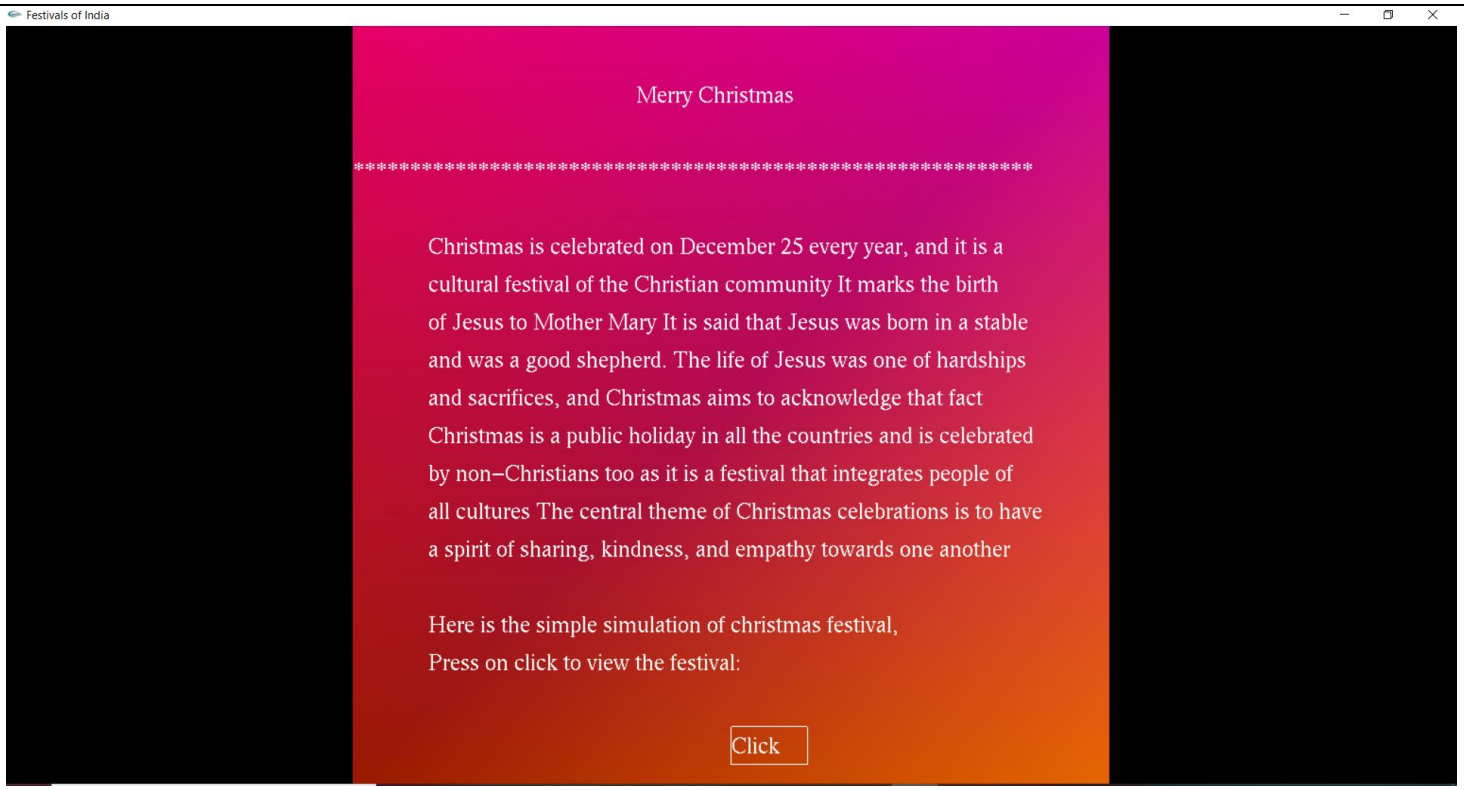

Fig.3. Christmas Description window

Fig.4. Christmas Simulation

Fig.5. Christmas Simulation

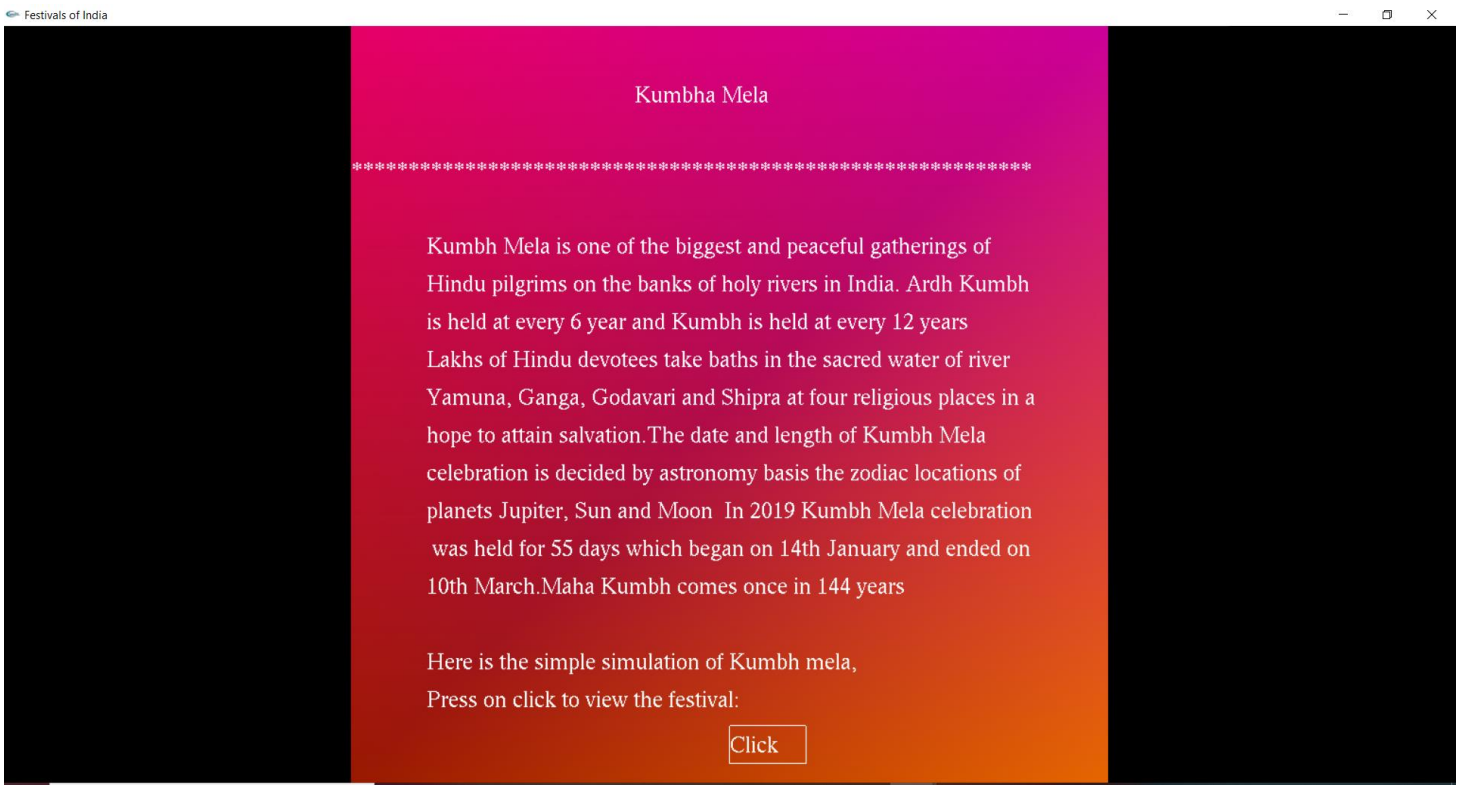

Fig.6 Kumbha Mela description window

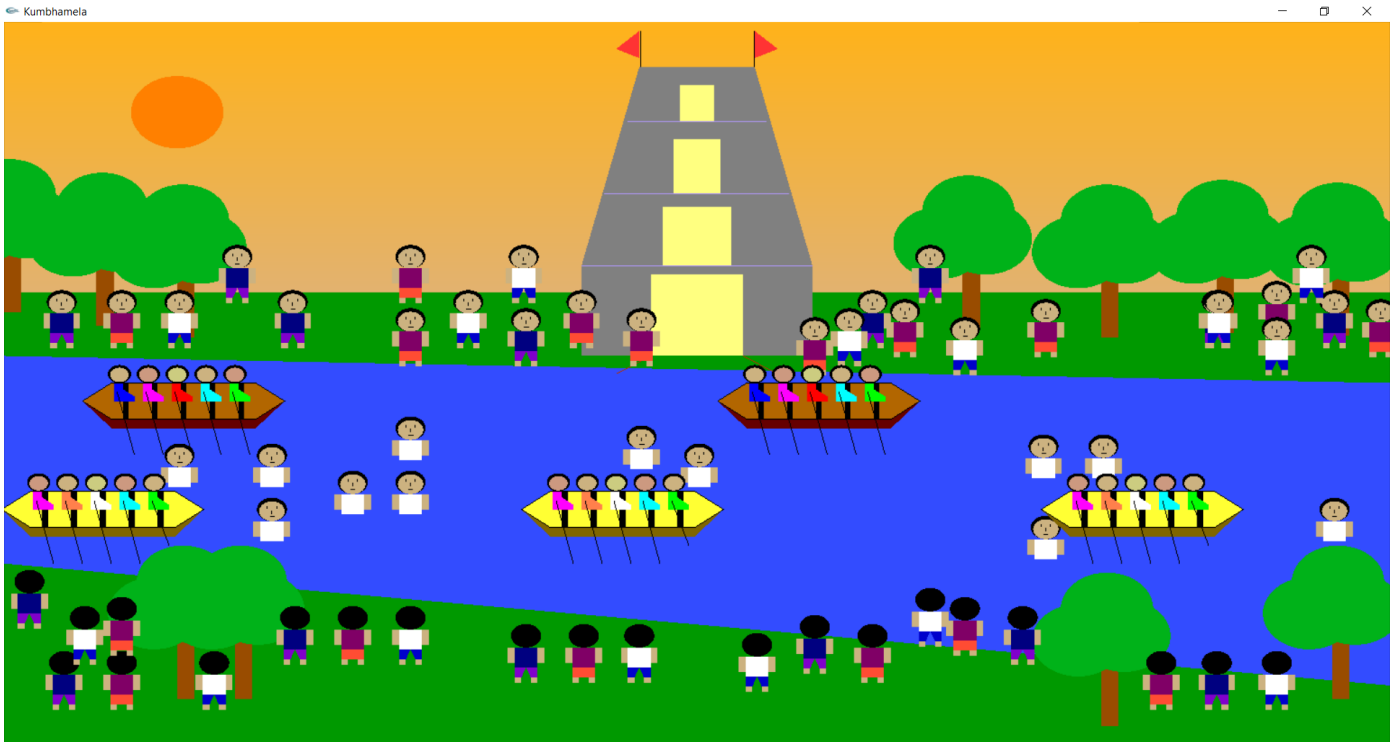

Fig.7. Kumbha Mela simulation

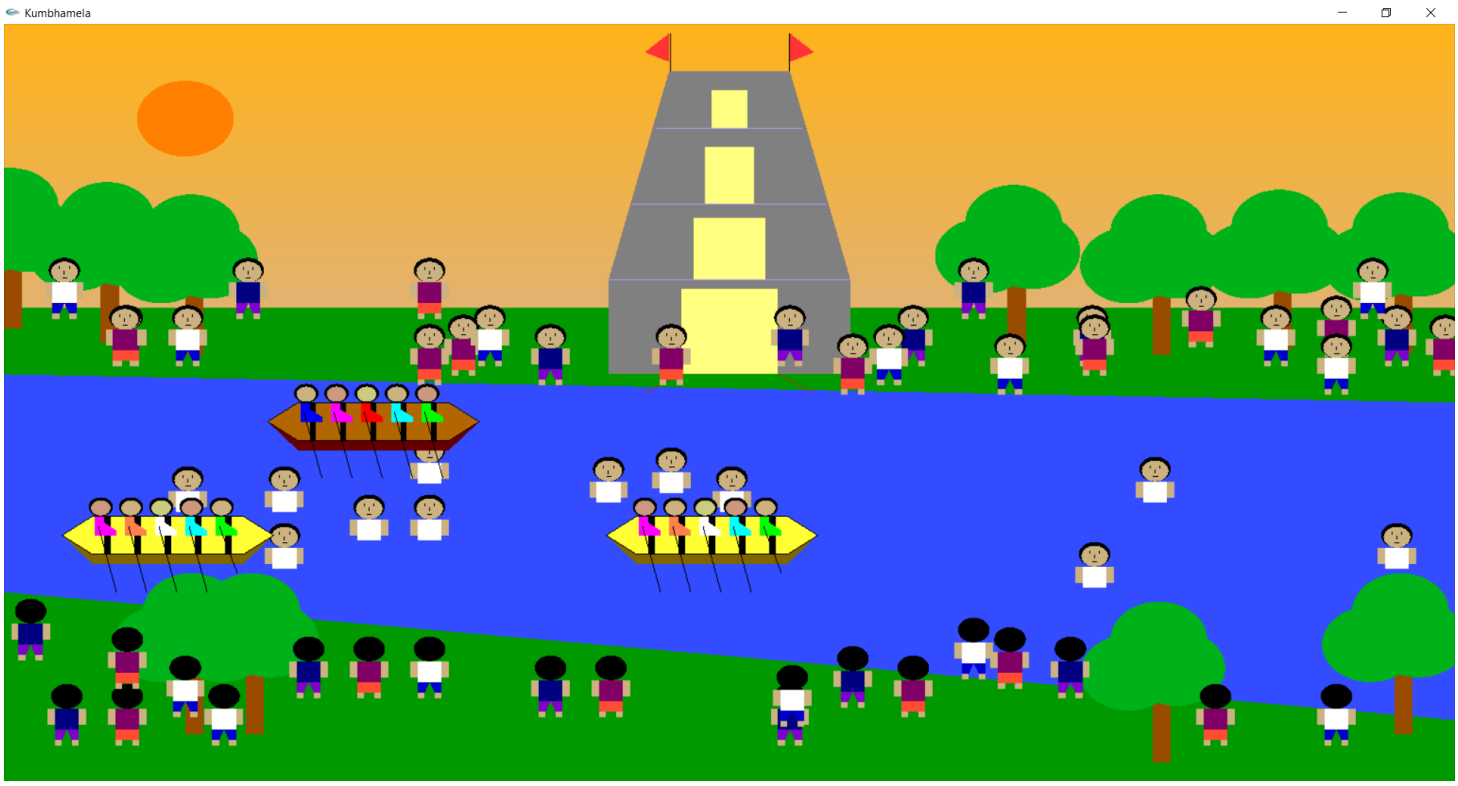

Fig.8. Kumbha Mela simulation

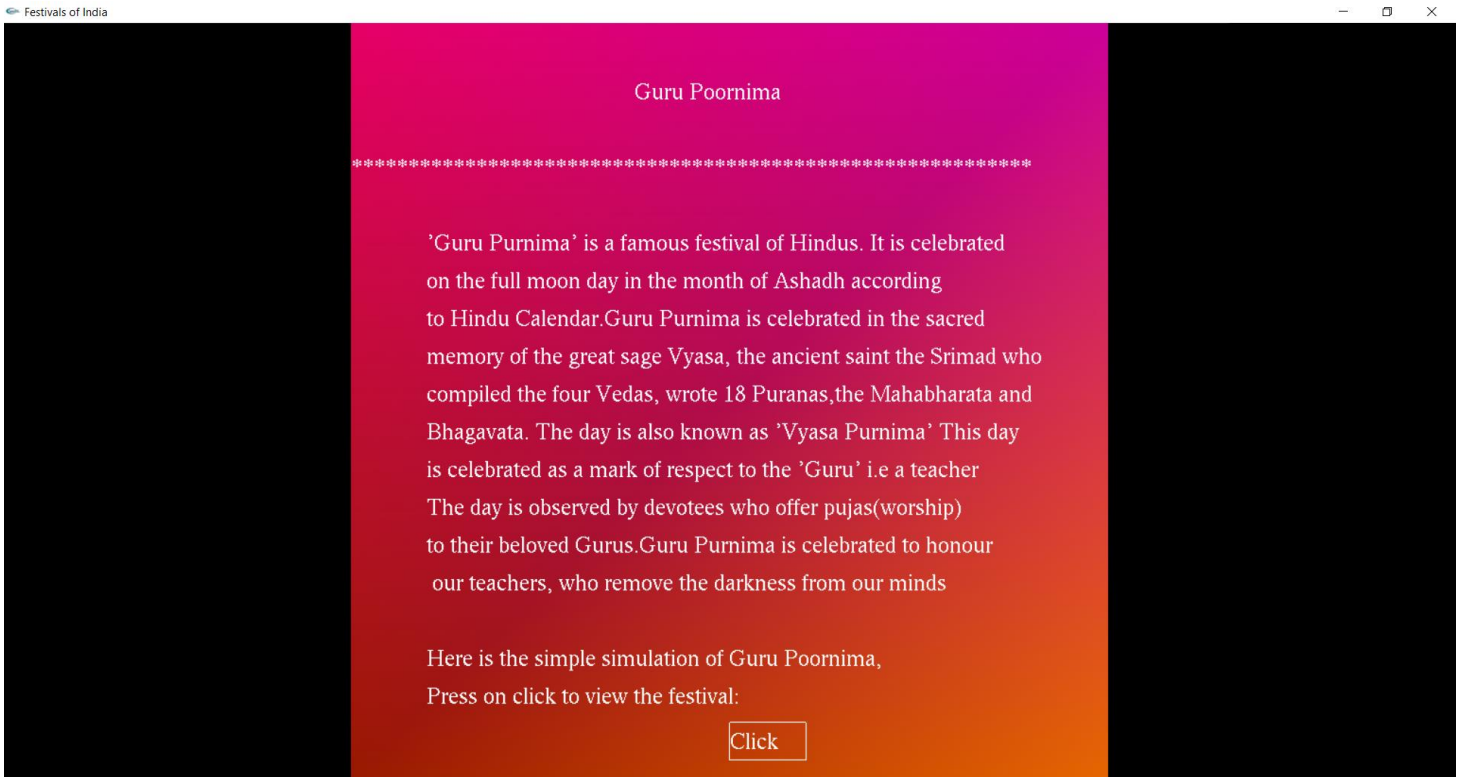

Fig.9. Guru Poornima description window

Fig.10. Guru Poornima simulation

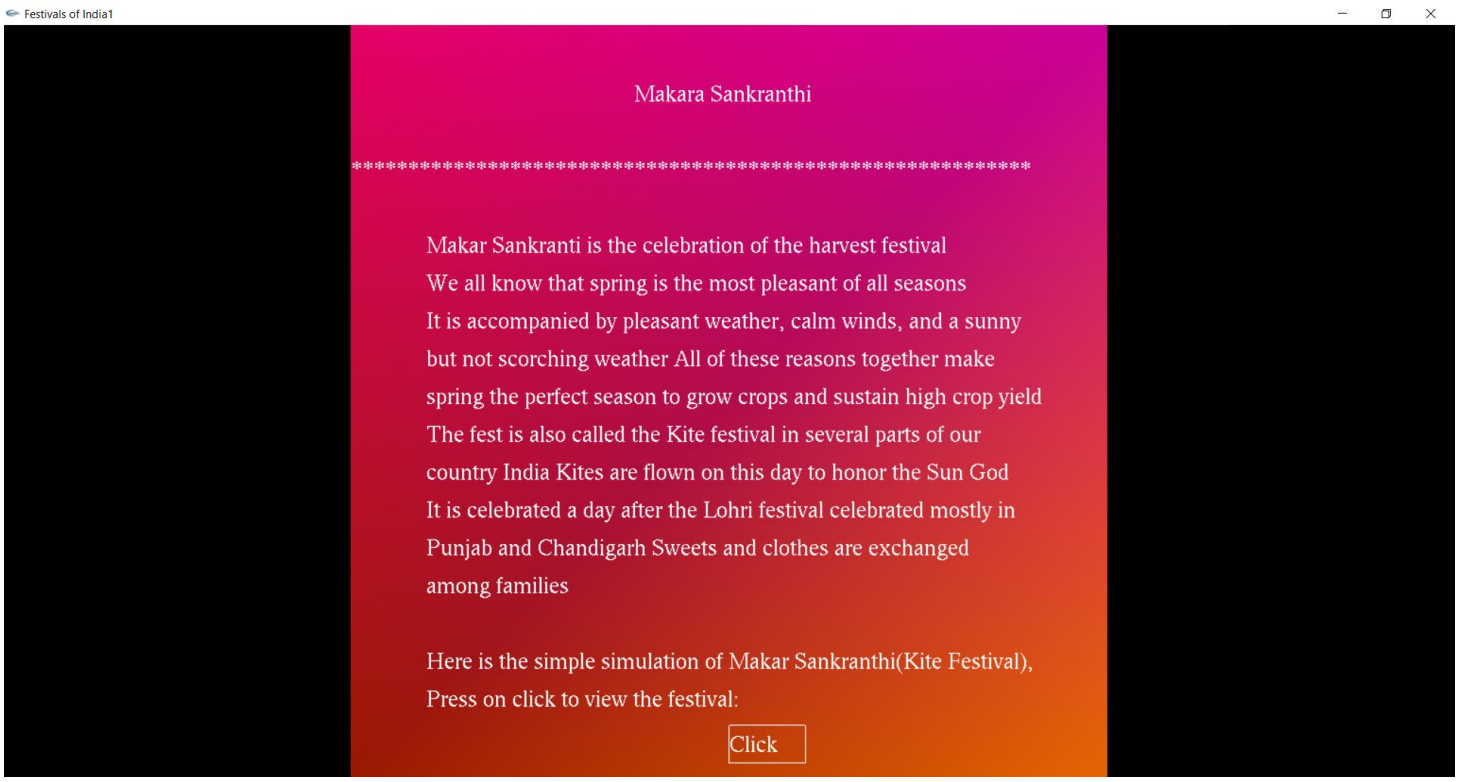

Fig.11. Makar Sankranthi description window

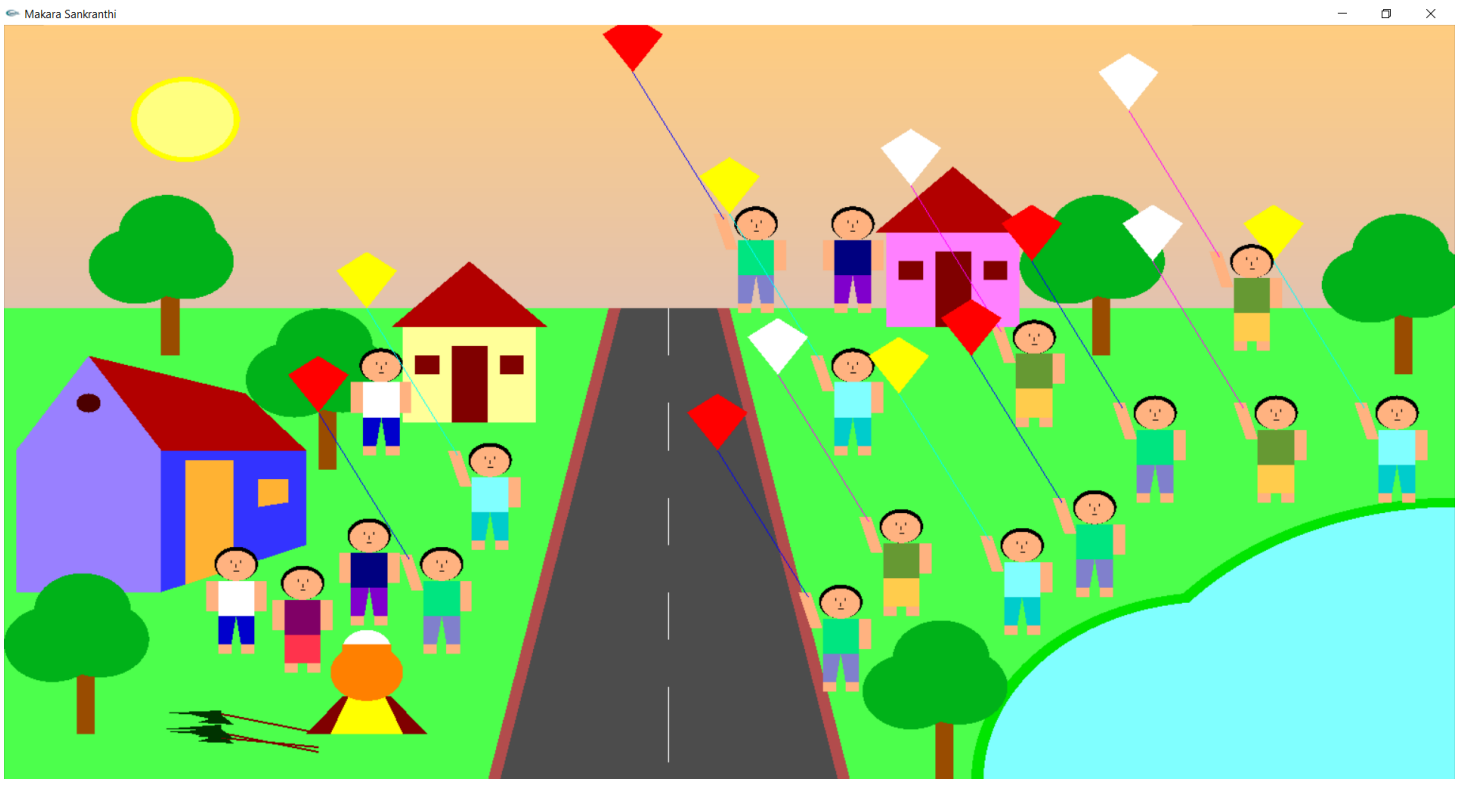

Fig. 12. Makar Sankranthi simulation

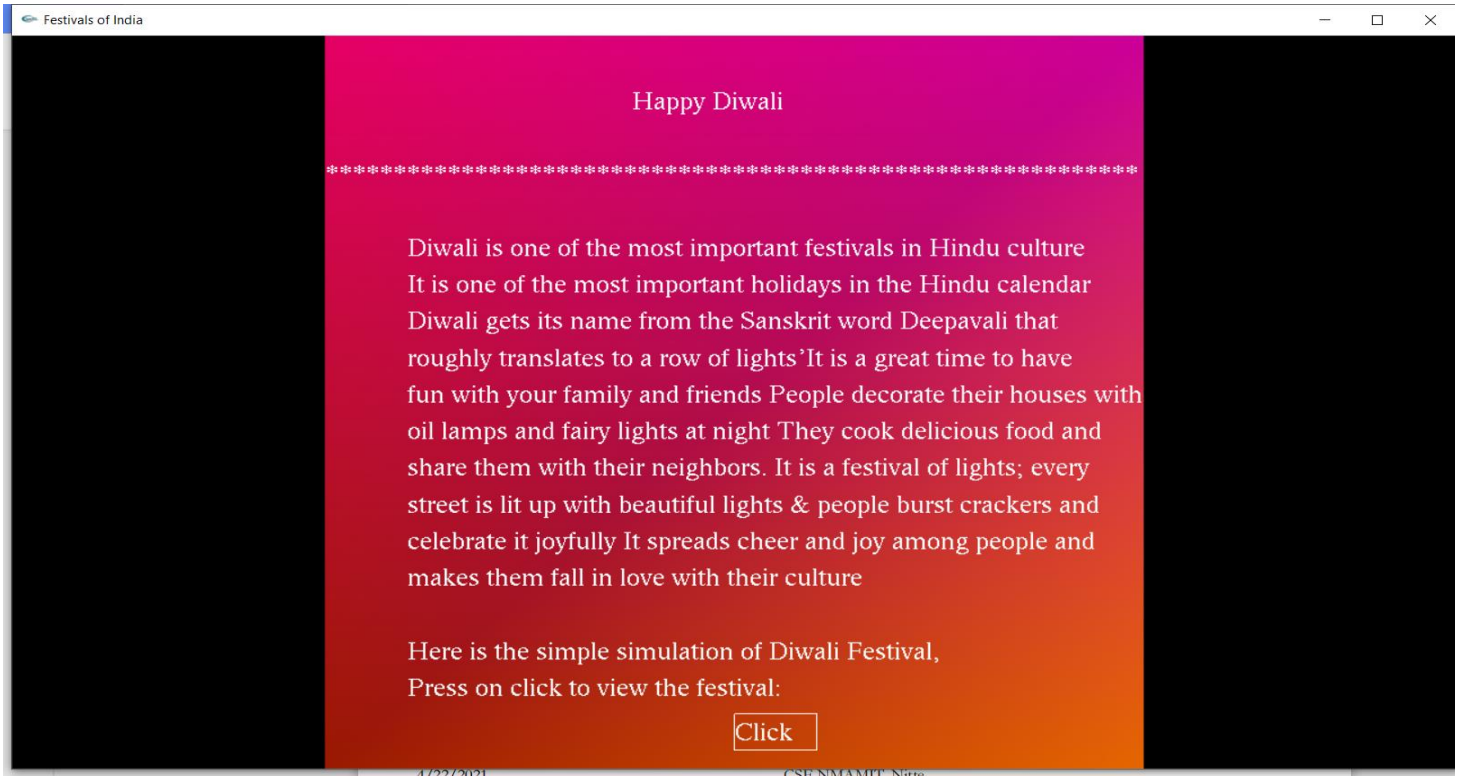

Fig.13. Diwali description window

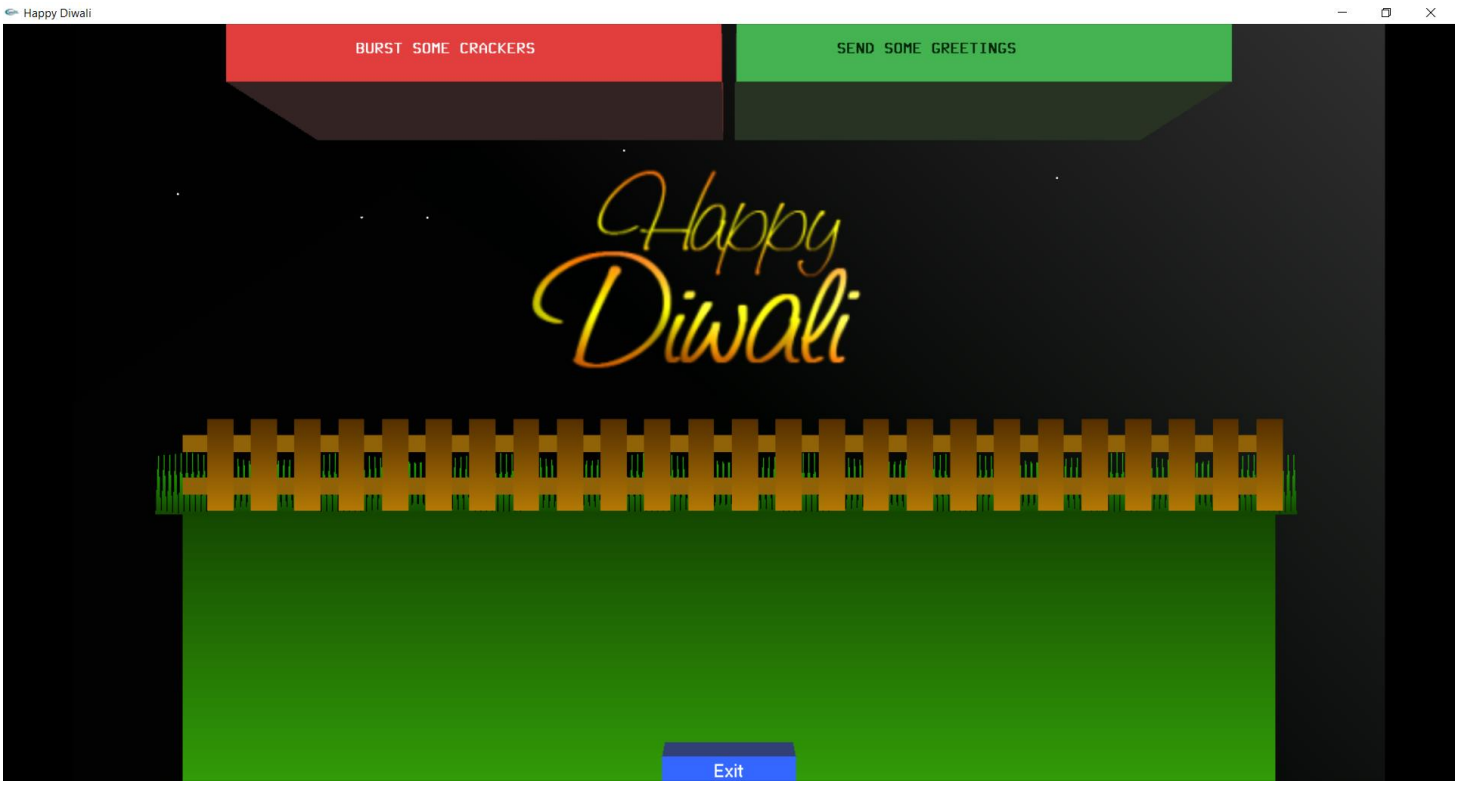

Fig.14. Diwali title page

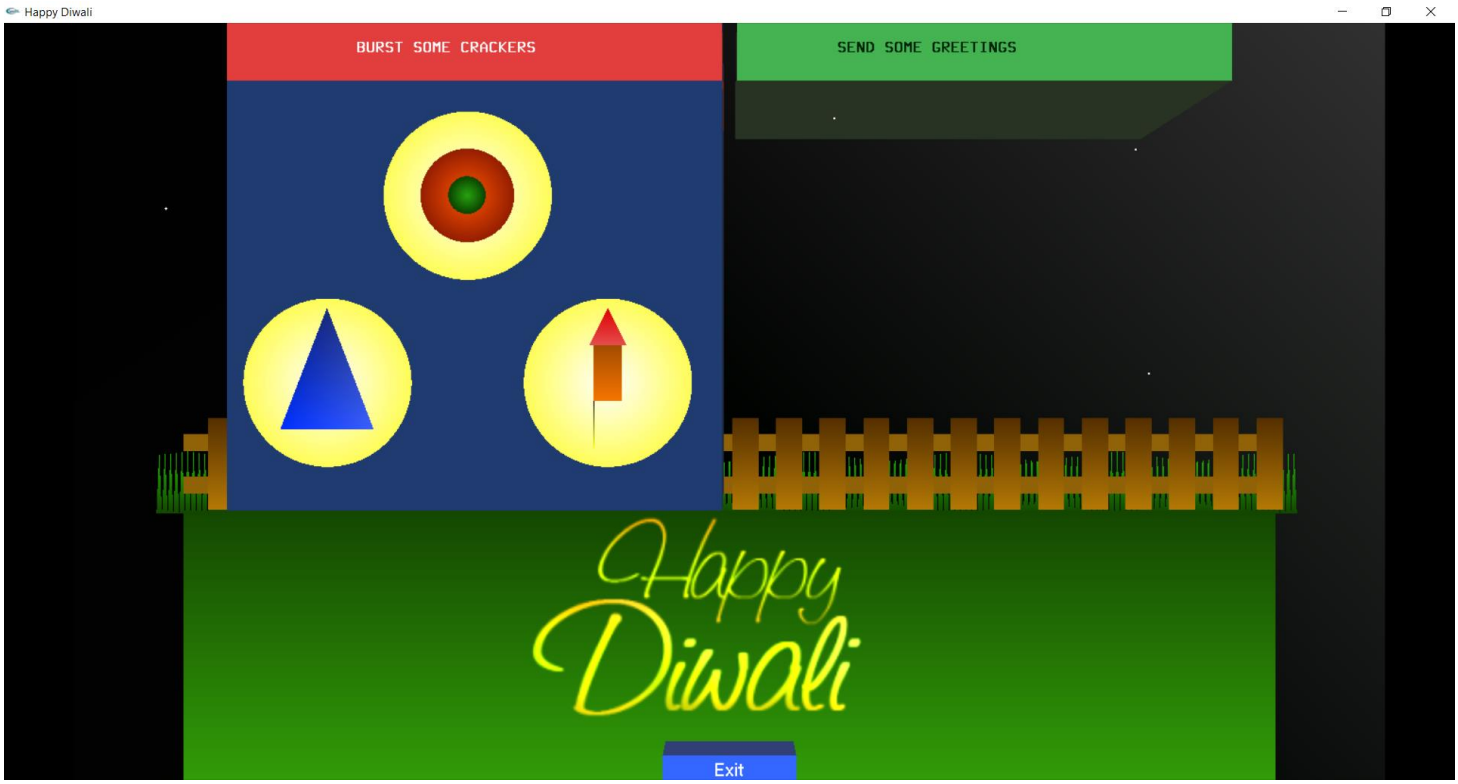

Fig.15. Left menu (Diwali)

Fig.16. Steady chakra and sparkle

Fig.17. Chakra and Sparkle simulation

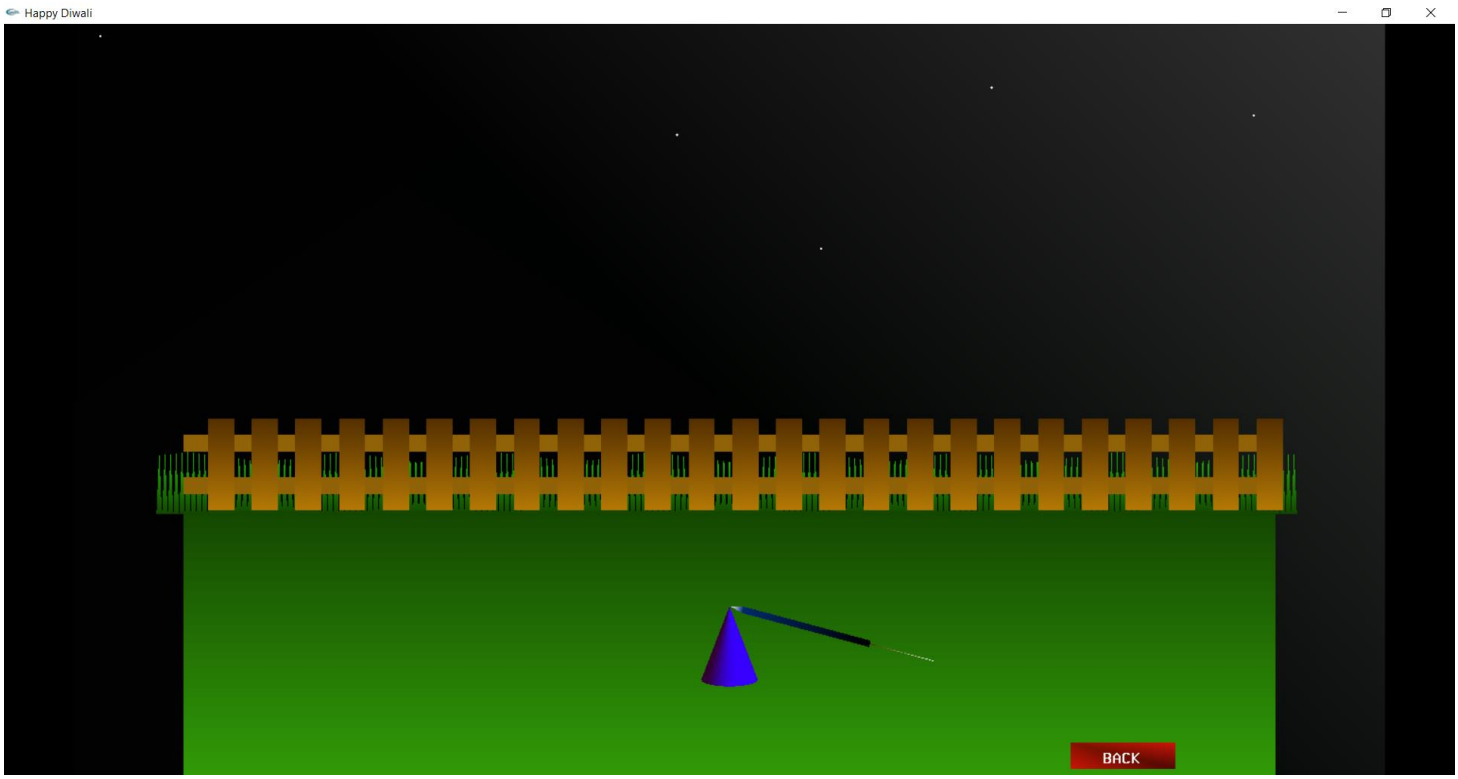

Fig.18. Steady flowerpot and sparkle

Fig.19. Flowerpot and Sparkle simulation

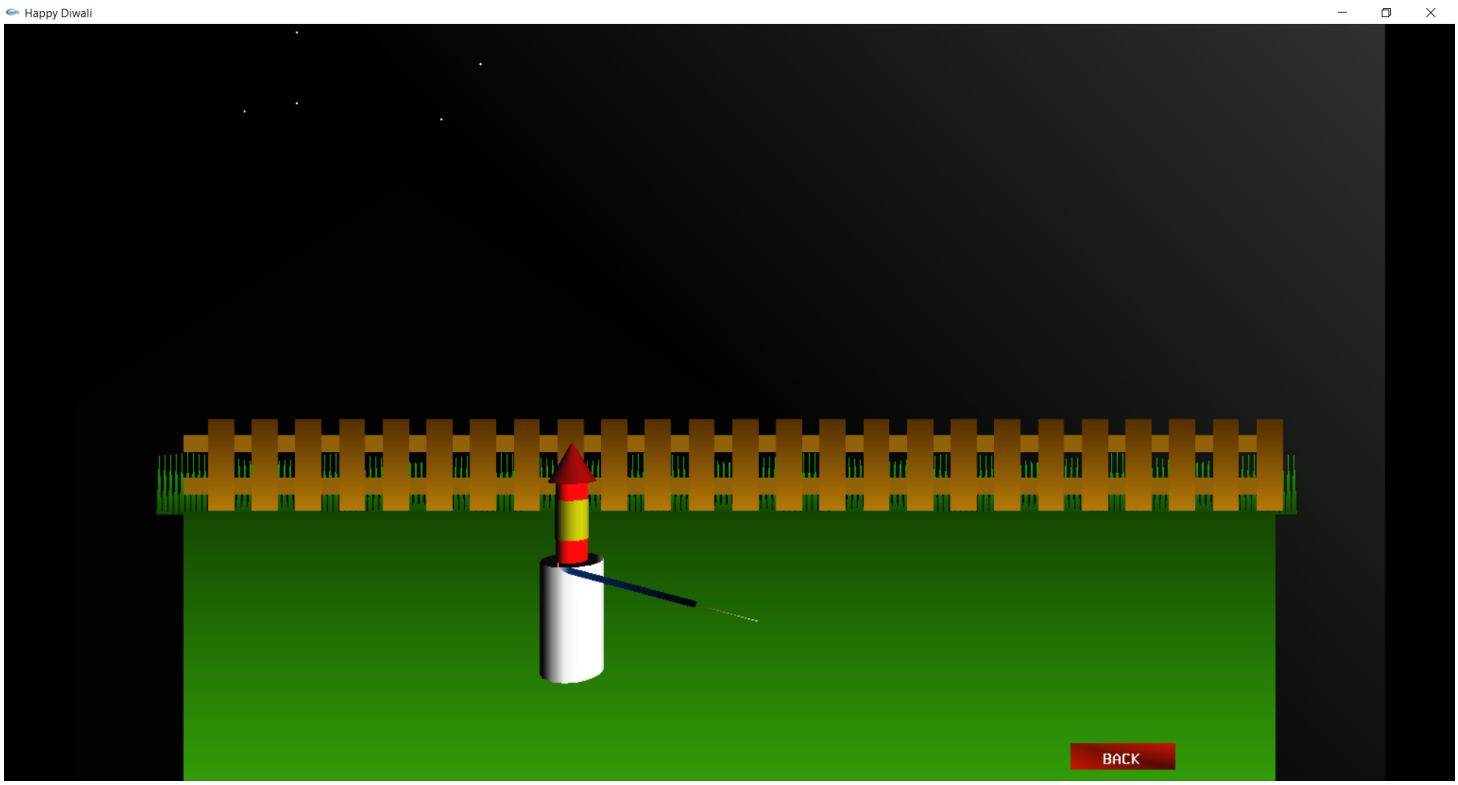

Fig.20. Steady rocket and sparkle

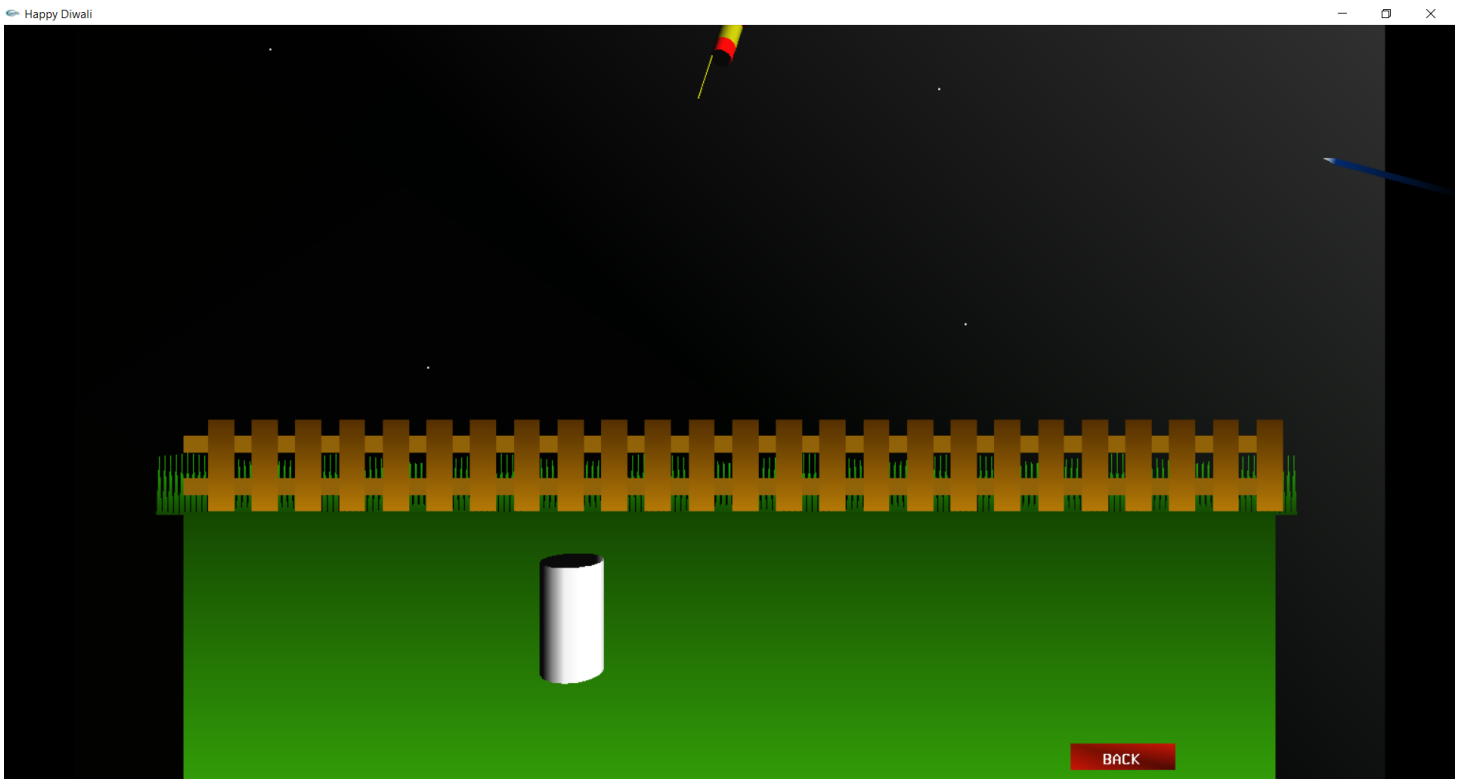

Fig.21. Rocket and sparkle simulation

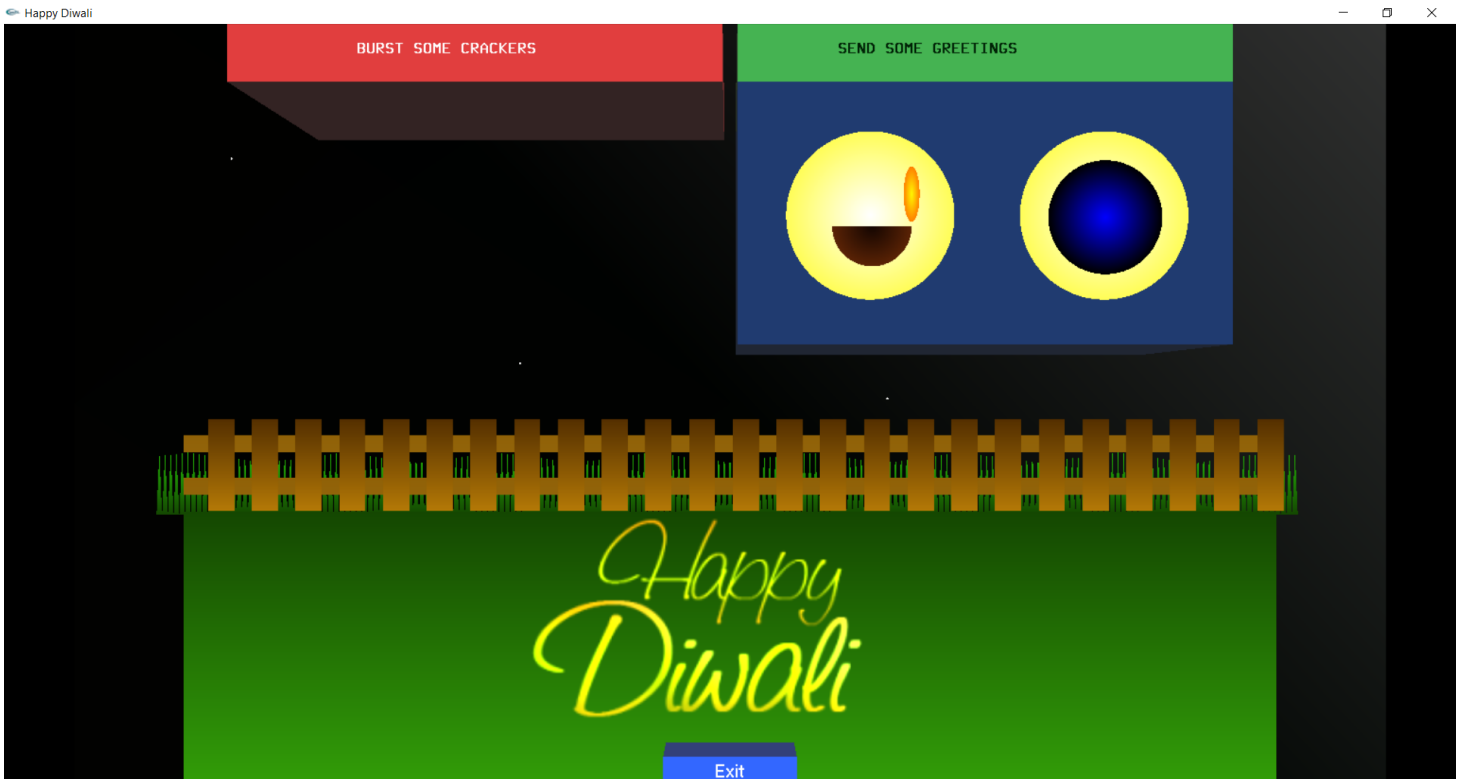

Fig.22. Right menu (Diwali)

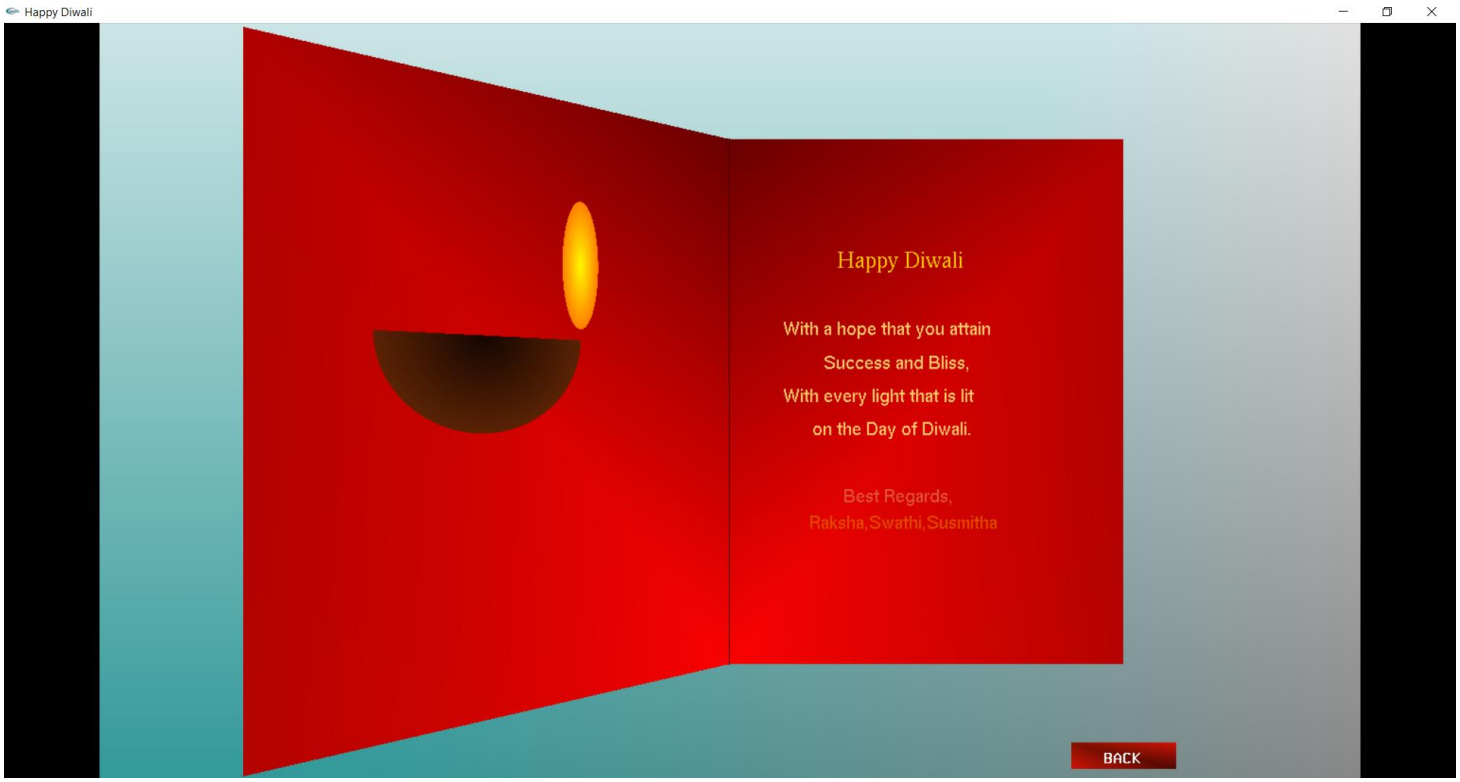

Fig.23. Diwali Greetings 1

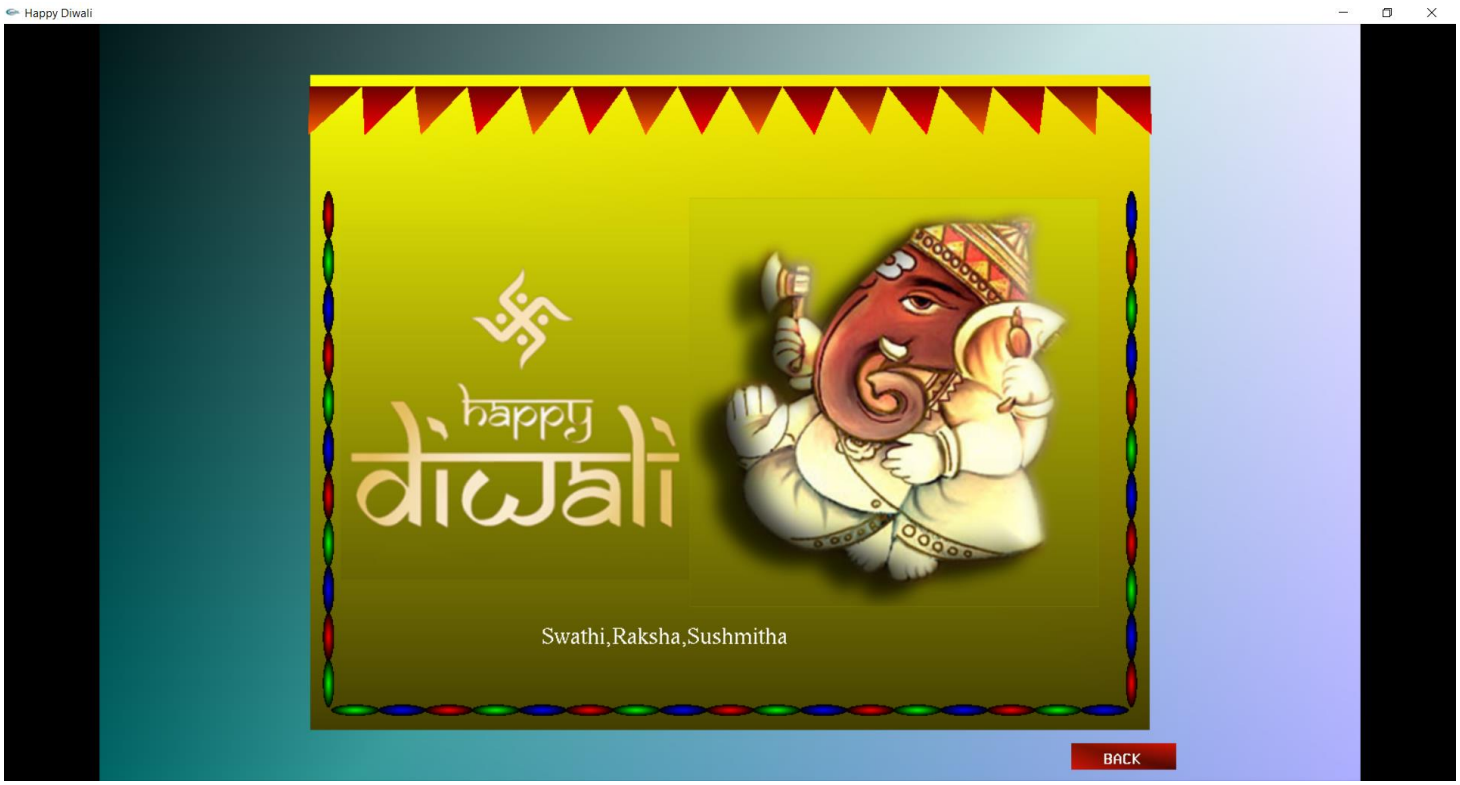

Fig.24. Diwali Greetings 2