**Lab Practice-7**

Submission Guidelines-

* Rename the file to your id only. If your id is 18-XXXXX-1, then the file name must be 18-XXXXX-1.docx.

|  |
| --- |
| **Question-**  Create a simple day and night scenario that will automatically change from day to night |
| **Graph** |
| **Code-**  #include <windows.h>  #include <GL/glut.h>  #include <math.h>  bool isDay = true; // Flag to track day or night  void tree() {  // Land  glBegin(GL\_POLYGON);  glColor3ub(81, 164, 64);  glVertex2f(0, 2);  glVertex2f(7, 2);  glVertex2f(7, 0);  glVertex2f(0, 0);  glEnd();  // Sky  glBegin(GL\_POLYGON);  if (isDay)  glColor3ub(115, 215, 255); // Daytime color  else  glColor3ub(0, 0, 51); // Nighttime color  glVertex2f(0, 2);  glVertex2f(7, 2);  glVertex2f(7, 7);  glVertex2f(0, 7);  glEnd();  // Sun or Moon  glBegin(GL\_POLYGON);  if (isDay)  glColor3ub(253, 184, 19); // Sun color  else  glColor3ub(192, 192, 192); // Moon color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.6;  float x = r \* cos(A) + 5.4;  float y = r \* sin(A) + 6;  glVertex2f(x, y);  }  glEnd();  // Trees  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(2.6, 3);  glVertex2f(2.9, 3);  glVertex2f(3.2, 2.6);  glVertex2f(3, 2.5);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3, 2.5);  glVertex2f(3.2, 2.6);  glVertex2f(3.2, 0.6);  glVertex2f(3, 0.6);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3.8, 3);  glVertex2f(3.5, 3);  glVertex2f(3.2, 2.6);  glVertex2f(3.4, 2.5);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3.2, 2.6);  glVertex2f(3.4, 2.5);  glVertex2f(3.4, 0.6);  glVertex2f(3.2, 0.6);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.65;  float x = r \* cos(A) + 2.1;  float y = r \* sin(A) + 3.1;  glVertex2f(x, y);  }  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.65;  float x = r \* cos(A) + 4.1;  float y = r \* sin(A) + 3.25;  glVertex2f(x, y);  }  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 1.3;  float x = r \* cos(A) + 3.2;  float y = r \* sin(A) + 4;  glVertex2f(x, y);  }  glEnd();  }  void display() {  glClearColor(1, 1, 1, 1.0f);  glClear(GL\_COLOR\_BUFFER\_BIT);  tree();  glFlush();  }  void updateSky(int value) {  isDay = !isDay; // Toggle day/night  glutPostRedisplay(); // Call display function  glutTimerFunc(5000, updateSky, 0); // Change every 5 seconds  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitWindowSize(500, 320);  glutInitWindowPosition(200, 50);  glutCreateWindow("Day and Night Scenario");  gluOrtho2D(0, 7, 0, 7);  glutDisplayFunc(display);  updateSky(0); // Start the timer to update the sky  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Create a simple day and night scenario using keyboard interaction. The key ‘D’ or ‘d’ will initiate the day mode and the key ‘N’ or ‘n’ will initiate the night mode. |
| **Graph** |
| **Code-**  #include <windows.h>  #include <GL/glut.h>  #include <math.h>  bool isDay = true; // Flag to track day or night  void tree() {  // Land  glBegin(GL\_POLYGON);  glColor3ub(81, 164, 64);  glVertex2f(0, 2);  glVertex2f(7, 2);  glVertex2f(7, 0);  glVertex2f(0, 0);  glEnd();  // Sky  glBegin(GL\_POLYGON);  if (isDay)  glColor3ub(115, 215, 255); // Daytime color  else  glColor3ub(0, 0, 51); // Nighttime color  glVertex2f(0, 2);  glVertex2f(7, 2);  glVertex2f(7, 7);  glVertex2f(0, 7);  glEnd();  // Sun or Moon  glBegin(GL\_POLYGON);  if (isDay)  glColor3ub(253, 184, 19); // Sun color  else  glColor3ub(192, 192, 192); // Moon color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.6;  float x = r \* cos(A) + 5.4;  float y = r \* sin(A) + 6;  glVertex2f(x, y);  }  glEnd();  // Trees  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(2.6, 3);  glVertex2f(2.9, 3);  glVertex2f(3.2, 2.6);  glVertex2f(3, 2.5);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3, 2.5);  glVertex2f(3.2, 2.6);  glVertex2f(3.2, 0.6);  glVertex2f(3, 0.6);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3.8, 3);  glVertex2f(3.5, 3);  glVertex2f(3.2, 2.6);  glVertex2f(3.4, 2.5);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(138, 115, 98);  glVertex2f(3.2, 2.6);  glVertex2f(3.4, 2.5);  glVertex2f(3.4, 0.6);  glVertex2f(3.2, 0.6);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.65;  float x = r \* cos(A) + 2.1;  float y = r \* sin(A) + 3.1;  glVertex2f(x, y);  }  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 0.65;  float x = r \* cos(A) + 4.1;  float y = r \* sin(A) + 3.25;  glVertex2f(x, y);  }  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(0, 255, 0); // Tree leaves color  for (int i = 0; i < 200; i++) {  float pi = 3.1416;  float A = (i \* 2 \* pi) / 200;  float r = 1.3;  float x = r \* cos(A) + 3.2;  float y = r \* sin(A) + 4;  glVertex2f(x, y);  }  glEnd();  }  void display() {  glClearColor(1, 1, 1, 1.0f);  glClear(GL\_COLOR\_BUFFER\_BIT);  tree();  glFlush();  }  void toggleDayNight(unsigned char key, int x, int y) {  if (key == 'D' || key == 'd') {  isDay = true;  } else if (key == 'N' || key == 'n') {  isDay = false;  }  glutPostRedisplay(); // Redraw the scene  }  void updateSky(int value) {  isDay = !isDay; // Toggle day/night  glutPostRedisplay(); // Call display function  glutTimerFunc(5000, updateSky, 0); // Change every 5 seconds  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitWindowSize(500, 320);  glutInitWindowPosition(200, 50);  glutCreateWindow("Day and Night Scenario");  gluOrtho2D(0, 7, 0, 7);  glutDisplayFunc(display);  glutKeyboardFunc(toggleDayNight); // Register keyboard callback function  updateSky(0); // Start the timer to update the sky  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-** |