**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>**  **float timeOfDay = 0.0; // Variable to represent time of day (0.0 for day, 1.0 for night)**  **float \_move=0.0;**  **/\* Initialize OpenGL Graphics \*/**  **void initGL() {**  **// Set "clearing" or background color**  **glClearColor(0.784f, 0.98f, 0.976f, 0.0f); // Daytime color**  **}**  **void setDayNightColors() {**  **if (timeOfDay < 0.5) {**  **// Interpolate between day and night colors for the sky**  **float black = 0.784 + timeOfDay \* 0.392; // Interpolate blue component**  **glClearColor(0.784f, 0.98f, black, 0.0f);**  **} else {**  **// Interpolate between day and night colors for the sky**  **float black = 1.176 - timeOfDay \* 0.392; // Interpolate blue component**  **glClearColor(0.0f, 0.0f, black, 0.0f);**  **}**  **// Adjust lighting conditions based on time of day**  **if (timeOfDay < 0.5) {**  **// Daytime colors and lighting**  **glColor3f(1.0f, 1.0f, 1.0f); // White for objects**  **} else {**  **// Nighttime colors and lighting**  **glColor3f(0.6f, 0.3f, 0.5f); // Dim gray for objects**  **}**  **}**  **void circle(float radius, float cX, float cY) {**  **glBegin(GL\_POLYGON);**  **for (int i = 0; i < 200; i++) {**  **if (timeOfDay < 0.5) {**  **glColor3f(1.0, 0.0, 0.0); // Red**  **}else{**  **glColor3f(1.0, 1.0, 1.0);**  **}**  **float pi = 3.1416;**  **float A = (i \* 2 \* pi) / 200;**  **float r = radius;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x + cX, y + cY);**  **}**  **glEnd();**  **}**  **void circle1(float radius, float cX, float cY) {**  **glBegin(GL\_POLYGON);**  **for (int i = 0; i < 200; i++) {**  **if (timeOfDay < 0.5) {**  **glColor3f(0.298, 0.522, 0.227); // Green during day**  **} else {**  **glColor3f(0.2, 0.2, 0.2); // Dark gray during night**  **}**  **float pi = 3.1416;**  **float A = (i \* 2 \* pi) / 200;**  **float r = radius;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x + cX, y + cY);**  **}**  **glEnd();**  **}**  **void sun()**  **{**  **glMatrixMode(GL\_MODELVIEW);**  **glPushMatrix();**  **glTranslatef(0.0f, \_move, 0.0f);**  **circle(10,70,80);**  **glPopMatrix();**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **setDayNightColors(); // Clear the color buffer with current clearing color**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.43, 0.53, 0.97); // Yellow**  **glVertex2f(90.0f, -15.0f);**  **glVertex2f(90.0f, 30.0f);**  **glVertex2f(-90.0f,30.0f);**  **glVertex2f(-90.0f,-15.0f);**  **glEnd();**  **////Road**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.787,0.78,0.725); // Yellow**  **glVertex2f(20.0f, 0.0f);**  **glVertex2f(12.0f, 15.0f);**  **glVertex2f(-12.0f, 15.0f);**  **glVertex2f(-20.0f,0.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.601,0.214,0.420); // Yellow**  **glVertex2f(18.0f, -15.0f);**  **glVertex2f(18.0f, 0.0f);**  **glVertex2f(-18.0f, 0.0f);**  **glVertex2f(-18.0f,-15.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(15.0f, -10.0f);**  **glVertex2f(15.0f, -5.0f);**  **glVertex2f(10.0f, -5.0f);**  **glVertex2f(10.0f, -10.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(-15.0f, -10.0f);**  **glVertex2f(-15.0f, -5.0f);**  **glVertex2f(-10.0f, -5.0f);**  **glVertex2f(-10.0f, -10.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(5.0f, -15.0f);**  **glVertex2f(5.0f, -5.0f);**  **glVertex2f(-5.0f, -5.0f);**  **glVertex2f(-5.0f, -15.0f);**  **glEnd();**  **/////tree**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.98,0.64,0.39); // Yellow**  **glVertex2f(-25.0f, -15.0f);**  **glVertex2f(-25.0f, 15.0f);**  **glVertex2f(-30.0f, 15.0f);**  **glVertex2f(-30.0f, -15.0f);**  **glEnd();**  **circle1(10,-32,15);**  **circle1(10,-23,15);**  **circle1(10,-27.5,17);**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.98,0.64,0.39); // Yellow**  **glVertex2f(25.0f, -15.0f);**  **glVertex2f(25.0f, 15.0f);**  **glVertex2f(30.0f, 15.0f);**  **glVertex2f(30.0f, -15.0f);**  **glEnd();**  **circle1(10, 32,15);**  **circle1(10, 23,15);**  **circle1(10, 27.5,17);**  **sun();**  **/////mountain**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(60.0f, 30.0f);**  **glVertex2f(20.0f, 30.0f);**  **glVertex2f(40.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(40.0f, 30.0f);**  **glVertex2f(-40.0f, 30.0f);**  **glVertex2f(10.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(-30.0f, 30.0f);**  **glVertex2f(-90.0f, 30.0f);**  **glVertex2f(-60.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(90.0f, 30.0f);**  **glVertex2f(20.0f, 30.0f);**  **glVertex2f(40.0f, 60.0f);**  **glEnd();**  **////sun**  **////**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(90.0f, -90.0f);**  **glVertex2f(90.0f, -15.0f);**  **glVertex2f(-90.0f, -15.0f);**  **glVertex2f(-90.0f, -90.0f);**  **glEnd();**  **glFlush(); // Render now**  **}**  **void update(int value) {**  **timeOfDay += 0.006; // Increment time of day**  **if (timeOfDay > 1.0) {**  **timeOfDay = 0.0; // Reset time of day to simulate day-night cycle**  **}**  **glutPostRedisplay();**  **glutTimerFunc(25, update, 0);**  **}**  **void update1(int value) {**  **\_move -= 0.5; // Increment time of day**  **if (\_move < -41.0) {**  **\_move = 0.0;// Reset time of day to simulate day-night cycle**  **}**  **glutPostRedisplay();**  **glutTimerFunc(25, update1, 0);**  **}**  **/\* Main function: GLUT runs as a console application starting at main() \*/**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv); // Initialize GLUT**  **glutCreateWindow("Vertex, Primitive & Color"); // Create window with the given title**  **glutInitWindowSize(320, 320); // Set the window's initial width & height**  **glutInitWindowPosition(50, 50);**  **// Position the window's initial top-left corner**  **glutDisplayFunc(display); // Register callback handler for window re-paint event**  **initGL();**  **gluOrtho2D(-90,90,-90,90);**  **glutTimerFunc(4, update, 0);**  **glutTimerFunc(4, update1, 0); // Our own OpenGL initialization**  **glutMainLoop(); // Enter the event-processing loop**  **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>**  **float timeOfDay = 0.0; // Variable to represent time of day (0.0 for day, 1.0 for night)**  **/\* Initialize OpenGL Graphics \*/**  **void initGL() {**  **// Set "clearing" or background color**  **glClearColor(0.784f, 0.98f, 0.976f, 0.0f); // Daytime color**  **}**  **void setDayNightColors() {**  **if (timeOfDay < 0.5) {**  **// Interpolate between day and night colors for the sky**  **float green = 0.298 + timeOfDay \* 0.224; // Interpolate green component**  **glClearColor(0.784f, green, 0.227, 0.0f);**  **} else {**  **// Interpolate between day and night colors for the sky**  **float black = timeOfDay \* 0.5; // Interpolate blue component**  **glClearColor(black, black, black, 0.0f);**  **}**  **// Adjust lighting conditions based on time of day**  **if (timeOfDay < 0.5) {**  **// Daytime colors and lighting**  **glColor3f(0.0f, 0.6f, 0.0f); // Green for objects**  **} else {**  **// Nighttime colors and lighting**  **glColor3f(0.0f, 0.0f, 0.0f); // Black for objects**  **}**  **}**  **void circle(float radius, float cX, float cY) {**  **glBegin(GL\_POLYGON);**  **for (int i = 0; i < 200; i++) {**  **glColor3f(1.0, 0.0, 0.0); // Red**  **float pi = 3.1416;**  **float A = (i \* 2 \* pi) / 200;**  **float r = radius;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x + cX, y + cY);**  **}**  **glEnd();**  **}**  **void circle1(float radius, float cX, float cY) {**  **glBegin(GL\_POLYGON);**  **for (int i = 0; i < 200; i++) {**  **if (timeOfDay < 0.5) {**  **glColor3f(0.0, 0.6, 0.0); // Green during day**  **} else {**  **glColor3f(0.298,0.522,0.227); // Black during night**  **}**  **float pi = 3.1416;**  **float A = (i \* 2 \* pi) / 200;**  **float r = radius;**  **float x = r \* cos(A);**  **float y = r \* sin(A);**  **glVertex2f(x + cX, y + cY);**  **}**  **glEnd();**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **setDayNightColors(); // Clear the color buffer with the current clearing color**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.43, 0.53, 0.97); // Yellow**  **glVertex2f(90.0f, -15.0f);**  **glVertex2f(90.0f, 30.0f);**  **glVertex2f(-90.0f, 30.0f);**  **glVertex2f(-90.0f, -15.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.787,0.78,0.725); // Yellow**  **glVertex2f(20.0f, 0.0f);**  **glVertex2f(12.0f, 15.0f);**  **glVertex2f(-12.0f, 15.0f);**  **glVertex2f(-20.0f,0.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.601,0.214,0.420); // Yellow**  **glVertex2f(18.0f, -15.0f);**  **glVertex2f(18.0f, 0.0f);**  **glVertex2f(-18.0f, 0.0f);**  **glVertex2f(-18.0f,-15.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(15.0f, -10.0f);**  **glVertex2f(15.0f, -5.0f);**  **glVertex2f(10.0f, -5.0f);**  **glVertex2f(10.0f, -10.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(-15.0f, -10.0f);**  **glVertex2f(-15.0f, -5.0f);**  **glVertex2f(-10.0f, -5.0f);**  **glVertex2f(-10.0f, -10.0f);**  **glEnd();**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.0,0.0,0.0); // Yellow**  **glVertex2f(5.0f, -15.0f);**  **glVertex2f(5.0f, -5.0f);**  **glVertex2f(-5.0f, -5.0f);**  **glVertex2f(-5.0f, -15.0f);**  **glEnd();**  **/////tree**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.98,0.64,0.39); // Yellow**  **glVertex2f(-25.0f, -15.0f);**  **glVertex2f(-25.0f, 15.0f);**  **glVertex2f(-30.0f, 15.0f);**  **glVertex2f(-30.0f, -15.0f);**  **glEnd();**  **circle1(10,-32,15);**  **circle1(10,-23,15);**  **circle1(10,-27.5,17);**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.98,0.64,0.39); // Yellow**  **glVertex2f(25.0f, -15.0f);**  **glVertex2f(25.0f, 15.0f);**  **glVertex2f(30.0f, 15.0f);**  **glVertex2f(30.0f, -15.0f);**  **glEnd();**  **circle1(10, 32,15);**  **circle1(10, 23,15);**  **circle1(10, 27.5,17);**  **//mountain**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(60.0f, 30.0f);**  **glVertex2f(20.0f, 30.0f);**  **glVertex2f(40.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(40.0f, 30.0f);**  **glVertex2f(-40.0f, 30.0f);**  **glVertex2f(10.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(-30.0f, 30.0f);**  **glVertex2f(-90.0f, 30.0f);**  **glVertex2f(-60.0f, 60.0f);**  **glEnd();**  **glBegin(GL\_POLYGON); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Yellow**  **glVertex2f(90.0f, 30.0f);**  **glVertex2f(20.0f, 30.0f);**  **glVertex2f(40.0f, 60.0f);**  **glEnd();**  **//...**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.298, 0.522, 0.227); // Yellow**  **glVertex2f(90.0f, -90.0f);**  **glVertex2f(90.0f, -15.0f);**  **glVertex2f(-90.0f, -15.0f);**  **glVertex2f(-90.0f, -90.0f);**  **glEnd();**  **glFlush(); // Render now/ ... (Remaining drawing code remains unchanged)**  **//...**  **glBegin(GL\_QUADS); // These vertices form a closed polygon**  **glColor3f(0.298,0.522,0.227); // Black**  **glVertex2f(90.0f, -90.0f);**  **glVertex2f(90.0f, -15.0f);**  **glVertex2f(-90.0f, -15.0f);**  **glVertex2f(-90.0f, -90.0f);**  **glEnd();**  **glFlush(); // Render now**  **}**  **void update(int value) {**  **timeOfDay += 0.005; // Increment time of day**  **if (timeOfDay > 1.0) {**  **timeOfDay = 0.0; // Reset time of day to simulate day-night cycle**  **}**  **glutPostRedisplay();**  **glutTimerFunc(25, update, 0);**  **}**  **// Keyboard callback function**  **void keyboard(unsigned char key, int x, int y) {**  **switch (key) {**  **case 'D':**  **case 'd':**  **timeOfDay = 0.0; // Set to day mode**  **break;**  **case 'N':**  **case 'n':**  **timeOfDay = 0.5; // Set to night mode**  **break;**  **}**  **glutPostRedisplay();**  **}**  **/\* Main function: GLUT runs as a console application starting at main() \*/**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv); // Initialize GLUT**  **glutCreateWindow("Vertex, Primitive & Color"); // Create window with the given title**  **glutInitWindowSize(320, 320); // Set the window's initial width & height**  **glutInitWindowPosition(50, 50);**  **// Position the window's initial top-left corner**  **glutDisplayFunc(display); // Register callback handler for window re-paint event**  **glutKeyboardFunc(keyboard); // Register keyboard callback function**  **initGL();**  **gluOrtho2D(-90, 90, -90, 90); // Our own OpenGL initialization**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |