**Lab Taks-6**

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.
* Must include resources for all the section in the table

|  |
| --- |
| **Question-**  Develop an animation that will change the background color of the window after 20ms. Use at least two different colors. |
| **Code-**  #include <windows.h>  #include <GL/glut.h>  #include <math.h>  #include <cstdlib>  #include <ctime>  #include <GL/glut.h>  GLfloat white[] = {1.0f, 0.0f, 0.0f}; // White color  GLfloat yellow[] = {0.0f, 1.0f, 0.0f}; // Yellow color  GLfloat\* currentColor = white; // Initial color is white  // Function to update the background color  void updateColor(int value) {  if (currentColor == white) {  currentColor = yellow;  } else {  currentColor = white;  }  glutPostRedisplay();  glutTimerFunc(20, updateColor, 0); // Call update function again after 20 milliseconds  }  // Function to draw the scene  void drawScene() {  // Set background color  glClearColor(currentColor[0], currentColor[1], currentColor[2], 1.0f);  // Clear the color buffer  glClear(GL\_COLOR\_BUFFER\_BIT);  // Swap buffers to display the scene  glutSwapBuffers();  }  int main(int argc, char \*\*argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(800, 600);  glutCreateWindow("Color Changing Background");  glutDisplayFunc(drawScene);  glutTimerFunc(20, updateColor, 0);  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-**  A red square on a black background  Description automatically generated |

|  |
| --- |
| **Question-**  Develop an animation that will call four objects separately, each after 20 ms. |
| **Code-**  #include<windows.h>  #include <GL/glut.h>  #include <unistd.h>  #include <stdlib.h>  #include <math.h>  // Global variables  int currentObject = 0; // Index of the currently displayed object  int windowWidth = 300; // Width of the window  int windowHeight = 300; // Height of the window  // Function to draw a square  void drawSquare(float x, float y, float size) {  glBegin(GL\_QUADS);  glVertex2f(x, y);  glVertex2f(x + size, y);  glVertex2f(x + size, y + size);  glVertex2f(x, y + size);  glEnd();  }  // Function to draw a triangle  void drawTriangle(float x, float y, float size) {  glBegin(GL\_TRIANGLES);  glVertex2f(x + size / 2.0, y + size);  glVertex2f(x, y);  glVertex2f(x + size, y);  glEnd();  }  // Function to draw a polygon  void drawPolygon(float x, float y, float size, int sides) {  glBegin(GL\_POLYGON);  for (int i = 0; i < sides; i++) {  float angle = 2.0 \* M\_PI \* i / sides;  glVertex2f(x + size \* cos(angle), y + size \* sin(angle));  }  glEnd();  }  // Function to draw a circle  void drawCircle(float x, float y, float radius) {  glBegin(GL\_TRIANGLE\_FAN);  glVertex2f(x, y);  for(int i = 0; i <= 360; i++) {  glVertex2f(x + (radius \* cos(i \* M\_PI / 180.0)), y + (radius \* sin(i \* M\_PI / 180.0)));  }  glEnd();  }  // Function to draw objects  void drawObjects() {  float size = 50.0; // Size of the shapes  float offset = 25.0; // Offset for drawing shapes  switch (currentObject) {  case 0:  glColor3f(1.0, 0.0, 0.0); // Red color  drawSquare(100, 100, size);  break;  case 1:  glColor3f(0.0, 1.0, 0.0); // Green color  drawTriangle(175, 75, size);  break;  case 2:  glColor3f(0.0, 0.0, 1.0); // Blue color  drawPolygon(125, 175, size, 5);  break;  case 3:  glColor3f(1.0, 1.0, 0.0); // Yellow color  drawCircle(50, 150, size/2);  break;  }  }  // Timer callback function  void timer(int value) {  currentObject = (currentObject + 1) % 4; // Move to the next object  glutPostRedisplay(); // Request display refresh  glutTimerFunc(20, timer, 0); // Set the timer to call itself after 1 second  }  // Display callback function  void display() {  glClear(GL\_COLOR\_BUFFER\_BIT);  drawObjects(); // Draw the current object  glFlush();  }  // Initialize OpenGL  void init() {  glClearColor(0.0, 0.0, 0.0, 1.0); // Set clear color to black  gluOrtho2D(0, windowWidth, 0, windowHeight); // Set orthographic projection  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);  glutInitWindowSize(windowWidth, windowHeight);  glutCreateWindow("Object Animation");  init(); // Initialize OpenGL  glutDisplayFunc(display); // Set display callback function  glutTimerFunc(20, timer, 0); // Start the timer to display objects  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-**  A green triangle on a black background  Description automatically generated |

|  |
| --- |
| **Question-**  Develop a code that will have four different objects (keep it simple). The objects will move to the left, right, up and down in a loop. |
| **Code-**  #include <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  using namespace std;  float \_move = 0.0f;  float \_moveY=0.0f;  void Box(int x,int y,int z){  glColor3d(x, y, z);  glBegin(GL\_QUADS);  glVertex2f(-0.2f, 0.2f);  glVertex2f(0.2f, 0.20f);  glVertex2f(0.2f, 0.4f);  glVertex2f(-0.2f, 0.4f);  glEnd();  }  void Box2(int x,int y,int z){  glColor3d(x, y, z);  glBegin(GL\_QUADS);  glVertex2f(-0.1f, -0.2f);  glVertex2f(0.1f, -0.2f);  glVertex2f(0.1f, 0.2f);  glVertex2f(-0.1f, 0.2f);  glEnd();  }  void drawScene() {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glMatrixMode(GL\_MODELVIEW);  glPushMatrix();  glTranslatef(\_move, 0.0f, 0.0f);  Box(1,1,0);  glPopMatrix();  glPushMatrix();  glTranslatef(-\_move, 0.0f, 0.0f);  glTranslatef(0.0f, -0.40f, 0.0f);  Box(0,1,1);  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,\_moveY, 0.0f);  glTranslatef(0.0f, 0.0f, 0.0f);  Box2(0,1,0);  glPopMatrix();  glPushMatrix();  glTranslatef( 0.0f,-\_moveY, 0.0f);  glTranslatef(-0.4f, 0.0f, 0.0f);  Box2(1,1,1);  glPopMatrix();  glutSwapBuffers();  }  void update(int value) {  \_move += .02;  if(\_move > 1.3)  {  \_move = -1.3;  }  \_moveY += .02;  if(\_moveY > 1.3)  {  \_moveY = -1.3;  }  glutPostRedisplay();  glutTimerFunc(30, update, 0);  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(800, 800);  glutCreateWindow("Transformation");  glutDisplayFunc(drawScene);  glutTimerFunc(30, update, 0); //Add a timer  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-**  A screenshot of a computer  Description automatically generated |

|  |
| --- |
| **Question-**  Develop a code that will have four different objects (keep it simple). Four different keys will be dedicated each objects. The objects will move to the left, right, up and down in a loop as the keys are pressed individually. |
| **Code-**  #include<cstdio>  #include <GL/gl.h>  #include <GL/glut.h>  // Define initial positions for the objects  float object1\_posX = -0.5f;  float object2\_posX = -0.3f;  float object3\_posX = 0.3f;  float object4\_posX = 0.5f;  float object1\_posY = 0.5f;  float object2\_posY = 0.3f;  float object3\_posY = 0.3f;  float object4\_posY = 0.5f;  // Define the speed of movement  float speed = 0.01f;  void display() {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  // Draw objects  glColor3f(1.0f, 0.0f, 0.0f); // Red  glBegin(GL\_QUADS);  glVertex2f(object1\_posX, object1\_posY);  glVertex2f(object1\_posX + 0.1f, object1\_posY);  glVertex2f(object1\_posX + 0.1f, object1\_posY - 0.1f);  glVertex2f(object1\_posX, object1\_posY - 0.1f);  glEnd();  glColor3f(0.0f, 1.0f, 0.0f); // Green  glBegin(GL\_QUADS);  glVertex2f(object2\_posX, object2\_posY);  glVertex2f(object2\_posX + 0.1f, object2\_posY);  glVertex2f(object2\_posX + 0.1f, object2\_posY - 0.1f);  glVertex2f(object2\_posX, object2\_posY - 0.1f);  glEnd();  glColor3f(0.0f, 0.0f, 1.0f); // Blue  glBegin(GL\_QUADS);  glVertex2f(object3\_posX, object3\_posY);  glVertex2f(object3\_posX + 0.1f, object3\_posY);  glVertex2f(object3\_posX + 0.1f, object3\_posY - 0.1f);  glVertex2f(object3\_posX, object3\_posY - 0.1f);  glEnd();  glColor3f(1.0f, 1.0f, 0.0f); // Yellow  glBegin(GL\_QUADS);  glVertex2f(object4\_posX, object4\_posY);  glVertex2f(object4\_posX + 0.1f, object4\_posY);  glVertex2f(object4\_posX + 0.1f, object4\_posY - 0.1f);  glVertex2f(object4\_posX, object4\_posY - 0.1f);  glEnd();  glutSwapBuffers();  }  // Callback function for handling keyboard presses  void handleKeypress(unsigned char key, int x, int y) {  switch (key) {  case 'j':  // Move object 1 to the left  object1\_posX -= speed;  if (object1\_posX < -1.0f)  object1\_posX = 1.0f;  break;  case 'l':  // Move object 2 to the right  object2\_posX += speed;  if (object2\_posX > 1.0f)  object2\_posX = -1.0f;  break;  case 'i':  // Move object 3 up  object3\_posY += speed;  if (object3\_posY > 1.0f)  object3\_posY = -1.0f;  break;  case 'k':  // Move object 4 down  object4\_posY -= speed;  if (object4\_posY < -1.0f)  object4\_posY = 1.0f;  break;  }  glutPostRedisplay();  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(800, 600);  glutCreateWindow("Moving Objects");  glutDisplayFunc(display);  glutKeyboardFunc(handleKeypress);  // Set up the viewport  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 1.0, -1.0, 1.0);  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-**  A screenshot of a computer  Description automatically generated |

|  |
| --- |
| **Question-**  Develop a code that will have four different objects (keep it simple). Two of the objects will move to the right as the right click is made on the mouse and two of the objects will move to the left as the left key is pressed on the mouse. |
| **Code-**  #include<cstdio>  #include <GL/gl.h>  #include <GL/glut.h>  float object1\_posX = -0.5f;  float object2\_posX = -0.3f;  float object3\_posX = 0.3f;  float object4\_posX = 0.5f;  float speed = 0.01f;  void display() {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glColor3f(1.0f, 0.0f, 0.0f);  glBegin(GL\_QUADS);  glVertex2f(object1\_posX, 0.0f);  glVertex2f(object1\_posX + 0.1f, 0.0f);  glVertex2f(object1\_posX + 0.1f, 0.1f);  glVertex2f(object1\_posX, 0.1f);  glEnd();  glColor3f(0.0f, 1.0f, 0.0f);  glBegin(GL\_QUADS);  glVertex2f(object2\_posX, 0.0f);  glVertex2f(object2\_posX + 0.1f, 0.0f);  glVertex2f(object2\_posX + 0.1f, -0.1f);  glVertex2f(object2\_posX, -0.1f);  glEnd();  glColor3f(0.0f, 0.0f, 1.0f);  glBegin(GL\_QUADS);  glVertex2f(object3\_posX, 0.0f);  glVertex2f(object3\_posX + 0.1f, 0.0f);  glVertex2f(object3\_posX + 0.1f, -0.1f);  glVertex2f(object3\_posX, -0.1f);  glEnd();  glColor3f(1.0f, 1.0f, 0.0f);  glBegin(GL\_QUADS);  glVertex2f(object4\_posX, 0.0f);  glVertex2f(object4\_posX + 0.1f, 0.0f);  glVertex2f(object4\_posX + 0.1f, 0.1f);  glVertex2f(object4\_posX, 0.1f);  glEnd();  glutSwapBuffers();  }  void handleMouse(int button, int state, int x, int y) {  if (button == GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN) {  object1\_posX += speed;  object2\_posX += speed;  } else if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN) {  object3\_posX -= speed;  object4\_posX -= speed;  }  glutPostRedisplay();  }  int main(int argc, char\*\* argv) {  glutInit(&argc, argv);  glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(800, 600);  glutCreateWindow("Moving Objects");  glutDisplayFunc(display);  glutMouseFunc(handleMouse);  glMatrixMode(GL\_PROJECTION);  glLoadIdentity();  gluOrtho2D(-1.0, 1.0, -1.0, 1.0);  glutMainLoop();  return 0;  } |
| **Output Screenshot (Full Screen)-** |