**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 <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  #include <windows.h>  #include <math.h>  using namespace std;  float r = 0;  float g = 0;  float b = 0;  void window()  {  glColor3f(0.9, 0.6, 0);  glBegin(GL\_POLYGON);  //  //upper black portion  //  glBegin(GL\_POLYGON);  glVertex2f(-95, 60);  glVertex2f(-80, 52);  glVertex2f(70, 52);  glVertex2f(82, 60);  glEnd();  //  //1st window  //  glBegin(GL\_POLYGON);  glVertex2f(-80, 52);  glVertex2f(-120, 75);  glVertex2f(-125,70);  glVertex2f(-125,-75);  glVertex2f(-120,-80);  glVertex2f(-80,-68);  glVertex2f(-80,-68);  glEnd();  //  //under black portion  //  glBegin(GL\_POLYGON);  glVertex2f(-80,-68);  glVertex2f(-80,-73);  glVertex2f(70,-73);  glVertex2f(70,-68);  glEnd();  //  //2nd window  //  glBegin(GL\_POLYGON);  glVertex2f(70,-68);  glVertex2f(110,-75);  glVertex2f(118,-72);  glVertex2f(118,74);  glVertex2f(110,80);  glVertex2f(70,52);  glEnd();  //  //1st window sqr  //  glColor3f(1, 1, 1);  glBegin(GL\_POLYGON);  glVertex2f(-85, 45);  glVertex2f(-115, 60);  glVertex2f(-115, 5);  glVertex2f(-85, 1);  glEnd();  glBegin(GL\_POLYGON);  glVertex2f(-85, -10);  glVertex2f(-115, -5);  glVertex2f(-115, -70);  glVertex2f(-85, -60);  glEnd();  //  //2nd window 2 sqr  //  glBegin(GL\_POLYGON);  glVertex2f(76, 48);  glVertex2f(102, 66);  glVertex2f(101.5, 5.5);  glVertex2f(76, 0);  glEnd();  glBegin(GL\_POLYGON);  glVertex2f(76, -10);  glVertex2f(76, -60);  glVertex2f(102, -64);  glVertex2f(102, -6);  glEnd();  //  //plus  //  glColor3f(0,0,0);  glBegin(GL\_POLYGON);  glVertex2f(-80,0);  glVertex2f(-80,-5);  glVertex2f(70,-5);  glVertex2f(70,0);  glEnd();  glColor3f(0,0,0);  glBegin(GL\_POLYGON);  glVertex2f(-10,52);  glVertex2f(-10,-68);  glVertex2f(-5,-68);  glVertex2f(-5,52);  glEnd();  //  //down layer  //  glColor3f(0.9, 0.6, 0);  glBegin(GL\_POLYGON);  glVertex2f(-80,-73);  glVertex2f(-60,-90);  glVertex2f(50,-90);  glVertex2f(70,-73);  glEnd();  }  void window\_background()  {  //  //window background  //  glMatrixMode(GL\_MODELVIEW);  glPushMatrix();  glColor3f(r, g, b);  glBegin(GL\_POLYGON);  glVertex2f(-80,52);  glVertex2f(-80,-68);  glVertex2f(70,-68);  glVertex2f(70,52);  glEnd();  glPopMatrix();  }  void update\_color(int value)  {  r += 0.01;  g += 0.01;  if (r >= 1)  {  r = 0;  }  if (g >= 1)  {  g = 0.5;  }  glutPostRedisplay();  glutTimerFunc(20, update\_color, 0);  }  void display()  {  glClearColor(1.0f, 1.0f, 1.0f, 1.0f);  glClear(GL\_COLOR\_BUFFER\_BIT);  window\_background();  window();  glFlush();  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(720, 720); // Set the window's initial width & height  glutCreateWindow("Window Animation");  glutDisplayFunc(display); // Register callback handler for window re-paint event  gluOrtho2D(-150,150,-150,150);  glutTimerFunc(20, update\_color, 0); //Add a timer  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Develop an animation that will call four objects separately, each after 20 ms. |
| **Code-**  #include <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  #include <windows.h>  #include <math.h>  using namespace std;  float position = 0.0f;  float position1 = 0.0f;  float speed = 0.1f;  void obj();  void display();  void update(int value)  {  if(position <-1.5)  position = 1.0f;  position -= speed;  glutPostRedisplay();  glutTimerFunc(20,update,0);  }  void update1(int value)  {  if(position1 >1.0)  position1 = -1.0f;  position1 += speed;  glutPostRedisplay();  glutTimerFunc(20,update1,0);  }  void init()  {  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  }  void disback(int val)  {  glutDisplayFunc(display);  }  void display7()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  glPushMatrix();  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glutTimerFunc(1500,disback,0);  glFlush();  }  void display6(int val)  {  glutDisplayFunc(display7);  }  void display5()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  glPushMatrix();  glBegin(GL\_QUADS);  glColor3f(0.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glutTimerFunc(1500,display6,0);  glFlush();  }  void display4(int val)  {  glutDisplayFunc(display5);  }  void display3()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  glPushMatrix();  glBegin(GL\_QUADS);  glColor3f(0.0f, 1.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glutTimerFunc(1500,display4,0);  glFlush();  }  void display2(int val)  {  glutDisplayFunc(display3);  }  void display()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glPushMatrix();  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glutTimerFunc(1500,display2,0);  glFlush();  }  void obj()  {  glutDisplayFunc(display);  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(720, 720); // Set the window's initial width & height  glutCreateWindow("Transformation");  glutDisplayFunc(obj); // Register callback handler for window re-paint event  init();  glutTimerFunc(20, update, 0); // Add a timer  glutTimerFunc(20, update1, 0);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **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>  #include <windows.h>  #include <math.h>  using namespace std;  float position = 0.0f;  float position1 = 0.0f;  float speed = 0.1f;  void display();  void update(int value)  {  if(position <-1.5)  position = 1.0f;  position -= speed;  glutPostRedisplay();  glutTimerFunc(200,update,0);  }  void update1(int value)  {  if(position1 >1.0)  position1 = -1.0f;  position1 += speed;  glutPostRedisplay();  glutTimerFunc(200,update1,0);  }  void init()  {  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  }  void display()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glPushMatrix();  glTranslatef(position,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(position1,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 1.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,position, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,position1, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glFlush();  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(720, 720); // Set the window's initial width & height  glutCreateWindow("Transformation");  glutDisplayFunc(display); // Register callback handler for window re-paint event  init();  glutTimerFunc(200, update, 0); // Add a timer  glutTimerFunc(200, update1, 0);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **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 <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  #include <windows.h>  #include <math.h>  using namespace std;  float position = 0.0f;  float position1 = 0.0f;  float position2 = 0.0f;  float position3 = 0.0f;  float speed = 0.0f;  float speed1 = 0.0f;  float speed2 = 0.0f;  float speed3 = 0.0f;  void display();  void update(int value)  {  if(position <-1.5)  position = 1.0f;  position -= speed;  glutPostRedisplay();  glutTimerFunc(100,update,0);  }  void update1(int value)  {  if(position1 >1.0)  position1 = -1.0f;  position1 += speed1;  glutPostRedisplay();  glutTimerFunc(100,update1,0);  }  void update2(int value)  {  if(position2 <-1.5)  position2 = 1.0f;  position2 -= speed2;  glutPostRedisplay();  glutTimerFunc(100,update2,0);  }  void update3(int value)  {  if(position3 >1.0)  position3 = -1.0f;  position3 += speed3;  glutPostRedisplay();  glutTimerFunc(100,update3,0);  }  void init()  {  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  }  void display()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glPushMatrix();  glTranslatef(position,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(position1,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 1.0f, 0.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,position2, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(0.0f,position3, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glFlush();  }  void handleKeypress(unsigned char key, int x, int y)  {  switch (key)  {  case 'a':  speed = 0.1f;  break;  case 'b':  speed1 = 0.1f;  break;  case 'c':  speed2 = 0.1f;  break;  case 'd':  speed3 = 0.1f;  break;  case 'e':  speed = 0.0f;  break;  case 'f':  speed1 = 0.0f;  break;  case 'g':  speed2 = 0.0f;  break;  case 'h':  speed3 = 0.0f;  break;  glutPostRedisplay();  }  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv)  {  glutInit(&argc, argv); // Initialize GLUT  //glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(720, 720); // Set the window's initial width & height  glutCreateWindow("Transformation");  glutDisplayFunc(display); // Register callback handler for window re-paint event  init();  glutTimerFunc(20, update, 0); //Add a timer  glutTimerFunc(20, update1, 0);  glutTimerFunc(20, update2, 0);  glutTimerFunc(20, update3, 0);  glutKeyboardFunc(handleKeypress);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **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 <iostream>  #include<GL/gl.h>  #include <GL/glut.h>  #include <windows.h>  #include <math.h>  using namespace std;  float position = 0.0f;  float position1 = 0.0f;  float speed = 0.0f;  float speed1 = 0.0f;  void display();  void update(int value)  {  if(position <-1.5)  position = 1.0f;  position -= speed;  glutPostRedisplay();  glutTimerFunc(100,update,0);  }  void update1(int value)  {  if(position1 >1.0)  position1 = -1.0f;  position1 += speed1;  glutPostRedisplay();  glutTimerFunc(100,update1,0);  }  void init()  {  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);  }  void display()  {  glClear(GL\_COLOR\_BUFFER\_BIT);  glLoadIdentity();  glPushMatrix();  glTranslatef(position,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 0.0f);  glVertex2f(-0.2f, 0.3f);  glVertex2f( 0.2f, 0.3f);  glVertex2f( 0.2f, 0.7f);  glVertex2f(-0.2f, 0.7f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(position1,0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 1.0f, 0.0f);  glVertex2f(-0.2f, 0.3f);  glVertex2f( 0.2f, 0.3f);  glVertex2f( 0.2f, 0.7f);  glVertex2f(-0.2f, 0.7f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(position, 0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(0.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glPushMatrix();  glTranslatef(position1, 0.0f, 0.0f);  glBegin(GL\_QUADS);  glColor3f(1.0f, 0.0f, 1.0f);  glVertex2f(-0.2f, -0.2f);  glVertex2f( 0.2f, -0.2f);  glVertex2f( 0.2f, 0.2f);  glVertex2f(-0.2f, 0.2f);  glEnd();  glPopMatrix();  glFlush();  }  void handleMouse(int button, int state, int x, int y)  {  if (button == GLUT\_LEFT\_BUTTON)  {  speed = 0.1f;  }  if (button == GLUT\_RIGHT\_BUTTON)  {  speed1 = 0.1f;  }  glutPostRedisplay();  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv)  {  glutInit(&argc, argv); // Initialize GLUT  //glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);  glutInitWindowSize(720, 720); // Set the window's initial width & height  glutCreateWindow("Transformation");  glutDisplayFunc(display); // Register callback handler for window re-paint event  init();  glutTimerFunc(20, update, 0); //Add a timer  glutTimerFunc(20, update1, 0);  glutMouseFunc(handleMouse);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |