**Lab Taks-1**

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 submit within the given deadline in VUES to the section named Lab Tak-1
* Must include resources for all the section in the table

|  |
| --- |
| **Question-**  Draw the object- |
| **Graph Plot (Picture)-**    Code: |
| #include <windows.h> // for MS Windows  #include <GL/glut.h> // GLUT, include glu.h and gl.h  void rect()  {  glColor3f(0,0,0);  glBegin(GL\_LINES);  glVertex2f(-6.36,4.92);  glVertex2f(6.58,4.94);  glVertex2f(6.58,4.94);  glVertex2f(6.68,-2.94);  glVertex2f(6.68,-2.94);  glVertex2f(-6.36,-2.98);  glVertex2f(-6.36,-2.98);  glVertex2f(-6.36,4.92);  glEnd();  }  void display() {  glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  rect();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitWindowSize(1320, 320);  glutInitWindowSize(320, 320);  glutCreateWindow("OpenGL Setup Test"); // Create a window with the given title  //glutInitWindowSize(320, 320); // Set the window's initial width & height  glutDisplayFunc(display); // Register display callback handler for window re-paint  gluOrtho2D(-20,10,-30,15);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Draw the object- |
| **Graph Plot (Picture)-** |
| **Code-**  #include <windows.h>  #include <GL/glut.h>  void trapizoidal()  {  glBegin(GL\_QUADS);  glColor3f(1,0,0);  glVertex2f(-6.99,2.43);  glVertex2f(-7.99,-1.61);  glVertex2f(-3.33,-1.61);  glVertex2f(-4.67,2.43);  glEnd();  }  void display() {  glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  trapizoidal();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitWindowSize(1320, 320);  glutCreateWindow("OpenGL Setup Test"); // Create a window with the given title  glutInitWindowSize(1320, 320); // Set the window's initial width & height  glutDisplayFunc(display); // Register display callback handler for window re-paint  gluOrtho2D(-10,10,-10,10);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Draw the object-  Octagon Shape | Area & Angles - Video & Lesson Transcript | Study.com |
| **Graph Plot (Picture)-** |
| #include <windows.h>  #include <GL/glut.h>  void octagon()  {  glBegin(GL\_POLYGON);  glColor3f(1,0,0);  glVertex2f(-7.21,2.31);  glVertex2f(-4.45,4.15);  glVertex2f(-0.35,4.21);  glVertex2f(2.87,2.09);  glVertex2f(2.83,-0.85);  glVertex2f(-0.19,-2.93);  glVertex2f(-4.27,-2.91);  glVertex2f(-7.17,-0.85);  glEnd();  }  void line()  {  glBegin(GL\_LINES);  glColor3f(0,0,0);  glVertex2f(-7.21,2.31);  glVertex2f(-4.45,4.15);  glVertex2f(-4.45,4.15);  glVertex2f(-0.35,4.21);  glVertex2f(-0.35,4.21);  glVertex2f(2.87,2.09);  glVertex2f(2.87,2.09);  glVertex2f(2.83,-0.85);  glVertex2f(2.83,-0.85);  glVertex2f(-0.19,-2.93);  glVertex2f(-0.19,-2.93);  glVertex2f(-4.27,-2.91);  glVertex2f(-4.27,-2.91);  glVertex2f(-7.17,-0.85);  glVertex2f(-7.17,-0.85);  glVertex2f(-7.21,2.31);  glEnd();  }  void display() {  glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  octagon();  line();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitWindowSize(1320, 320);  glutCreateWindow("OpenGL Setup Test"); // Create a window with the given title  glutInitWindowSize(1320, 320); // Set the window's initial width & height  glutDisplayFunc(display); // Register display callback handler for window re-paint  gluOrtho2D(-10,10,-10,10);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Draw the object- |
| **Graph Plot (Picture)-** |
| #include <windows.h>  #include <GL/glut.h>  void main\_axis()  {  glBegin(GL\_LINES);  glColor3ub(4,4,4);  glVertex2f(0,6);  glVertex2f(0,-6);  glVertex2f(8,0);  glVertex2f(-8,0);  glEnd();  }  void rectangle()  {  glBegin(GL\_POLYGON);  glColor3f(1,0,0);  glVertex2f(-6,5);  glVertex2f(-3,5);  glVertex2f(-3,2);  glVertex2f(-6,2);  glEnd();  }  void triangle\_1()  {  glBegin(GL\_POLYGON);  glColor3ub(76,33,87);  glVertex2f(-6,-3);  glVertex2f(-4,-2);  glVertex2f(-4,-4);  glEnd();  }  void triangle\_2()  {  glBegin(GL\_POLYGON);  glColor3ub(255,255,0);  glVertex2f(5,-2);  glVertex2f(4,-4);  glVertex2f(6,-4);  glEnd();  }  void arrow()  {  glBegin(GL\_POLYGON);  glColor3ub(20, 249, 9);  glVertex2f(1,3);  glVertex2f(5,3);  glVertex2f(5,2);  glVertex2f(1,2);  glEnd();  glBegin(GL\_POLYGON);  glColor3ub(20, 249, 9);  glVertex2f(4,4);  glVertex2f(7.022,2.587);  glVertex2f(4,1);  glEnd();  }  void display() {  glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT);  main\_axis();  rectangle();  triangle\_1();  triangle\_2();  arrow();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  //glutInitWindowSize(1320, 320);  glutCreateWindow("OpenGL Setup Test"); // Create a window with the given title  glutInitWindowSize(1320, 320); // Set the window's initial width & height  glutDisplayFunc(display); // Register display callback handler for window re-paint  gluOrtho2D(-10,10,-10,10);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |