**Lab Taks-2**

Submission Guidelines-

* Rename the file with your serial number only.
* Must submit within time that will be discussed in class VUES to the section named Lab Tak-2
* Must include resources for all the section in the table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Question- 1**  Draw a Rainbow Flag   |  | | --- | |  | |  | |  | |  | |  | |  | |  | |
| **Graph Plot (Picture)-** |
| **Code-**  **#include <windows.h>**  **#include <GL/glut.h>**  **#include <cmath>**  **box1(){**  **glBegin(GL\_QUADS);**  **glColor3ub(205, 169, 245);**  **glVertex2d(-4,4);**  **glVertex2d(4, 4);**  **glVertex2d(4, 3);**  **glVertex2d(-4, 3);**  **glEnd();**  **}**  **box2(){**  **glBegin(GL\_QUADS);**  **glColor3ub(156, 173, 250);**  **glVertex2d(-4,3);**  **glVertex2d(4, 3);**  **glVertex2d(4, 2);**  **glVertex2d(-4, 2);**  **glEnd();**  **}**  **box3(){**  **glBegin(GL\_QUADS);**  **glColor3ub(53, 176, 182);**  **glVertex2d(-4,2);**  **glVertex2d(4, 2);**  **glVertex2d(4, 1);**  **glVertex2d(-4, 1);**  **glEnd();**  **}**  **box4(){**  **glBegin(GL\_QUADS);**  **glColor3ub(10, 159, 12);**  **glVertex2d(-4,1);**  **glVertex2d(4, 1);**  **glVertex2d(4, 0);**  **glVertex2d(-4, 0);**  **glEnd();**  **}**  **box5(){**  **glBegin(GL\_QUADS);**  **glColor3ub(170, 90, 7);**  **glVertex2d(-4,0);**  **glVertex2d(4, 0);**  **glVertex2d(4, -1);**  **glVertex2d(-4, -1);**  **glEnd();**  **}**  **box6(){**  **glBegin(GL\_QUADS);**  **glColor3ub(30, 53, 3);**  **glVertex2d(-4,-1);**  **glVertex2d(4, -1);**  **glVertex2d(4, -2);**  **glVertex2d(-4, -2);**  **glEnd();**  **}**  **box7(){**  **glBegin(GL\_QUADS);**  **glColor3ub( 218, 59, 34);**  **glVertex2d(-4,-2);**  **glVertex2d(4, -2);**  **glVertex2d(4, -3);**  **glVertex2d(-4, -3);**  **glEnd();**  **}**  **box11(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,4);**  **glVertex2d(4, 4);**  **glVertex2d(4, 3);**  **glVertex2d(-4, 3);**  **glEnd();**  **}**  **box22(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,3);**  **glVertex2d(4, 3);**  **glVertex2d(4, 2);**  **glVertex2d(-4, 2);**  **glEnd();**  **}**  **box33(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,2);**  **glVertex2d(4, 2);**  **glVertex2d(4, 1);**  **glVertex2d(-4, 1);**  **glEnd();**  **}**  **box44(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,1);**  **glVertex2d(4, 1);**  **glVertex2d(4, 0);**  **glVertex2d(-4, 0);**  **glEnd();**  **}**  **box55(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,0);**  **glVertex2d(4, 0);**  **glVertex2d(4, -1);**  **glVertex2d(-4, -1);**  **glEnd();**  **}**  **box66(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,-1);**  **glVertex2d(4, -1);**  **glVertex2d(4, -2);**  **glVertex2d(-4, -2);**  **glEnd();**  **}**  **box77(){**  **glBegin(GL\_LINE\_LOOP);**  **glColor3ub(255, 255, 255);**  **glVertex2d(-4,-2);**  **glVertex2d(4, -2);**  **glVertex2d(4, -3);**  **glVertex2d(-4, -3);**  **glEnd();**  **}**  **Rainbow(){**  **box1();**  **box2();**  **box3(); box4();box5();box6();box7();**  **}**  **Line(){**  **box11();**  **box22();**  **box33(); box44();box55();box66();box77();**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **Rainbow();**  **Line();**  **glFlush();**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutCreateWindow("Rainbow");**  **glutInitWindowSize(720, 720);**  **glutDisplayFunc(display);**  **gluOrtho2D(-6, 6, -6, 6);**  **glutMainLoop();**  **}** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question- 2**  Draw 4X4 Chess Board |
| **Graph Plot (Picture)-** |
| **Code-#include <windows.h> // for MS Windows**  **#include <GL/glut.h>**  **#include <cmath> // GLUT, include glu.h and gl.h**  **void Board1(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(-2, 4);**  **glVertex2d(-1, 4);**  **glVertex2d(-1, 3);**  **glVertex2d(-2, 3);**  **glEnd();**  **}**  **void Board2(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(-1,4);**  **glVertex2d(0, 4);**  **glVertex2d(0, 3);**  **glVertex2d(-1, 3);**  **glEnd();**  **}**  **void Board3(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(0,4);**  **glVertex2d(1, 4);**  **glVertex2d(1, 3);**  **glVertex2d(0, 3);**  **glEnd();**  **}**  **void Board4(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(1,4);**  **glVertex2d(2, 4);**  **glVertex2d(2, 3);**  **glVertex2d(1, 3);**  **glEnd();**  **}**  **void Board5(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(1,3);**  **glVertex2d(2, 3);**  **glVertex2d(2, 2);**  **glVertex2d(1, 2);**  **glEnd();**  **}**  **void Board6(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(0,3);**  **glVertex2d(1, 3);**  **glVertex2d(1, 2);**  **glVertex2d(0, 2);**  **glEnd();**  **}**  **void Board7(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(-1,3);**  **glVertex2d(0, 3);**  **glVertex2d(0, 2);**  **glVertex2d(-1, 2);**  **glEnd();**  **}**  **void Board8(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(-2,3);**  **glVertex2d(-1, 3);**  **glVertex2d(-1, 2);**  **glVertex2d(-2, 2);**  **glEnd();**  **}**  **void Board9(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(-2,2);**  **glVertex2d(-1, 2);**  **glVertex2d(-1, 1);**  **glVertex2d(-2, 1);**  **glEnd();**  **}**  **void Board10(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(-1,2);**  **glVertex2d(0, 2);**  **glVertex2d(-0, 1);**  **glVertex2d(-1, 1);**  **glEnd();**  **}**  **void Board11(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(0,2);**  **glVertex2d(1, 2);**  **glVertex2d(1, 1);**  **glVertex2d(0, 1);**  **glEnd();**  **}**  **void Board12(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(1,2);**  **glVertex2d(2, 2);**  **glVertex2d(2, 1);**  **glVertex2d(1, 1);**  **glEnd();**  **}**  **void Board13(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(1,1);**  **glVertex2d(2, 1);**  **glVertex2d(2, 0);**  **glVertex2d(1, 0);**  **glEnd();**  **}**  **void Board14(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(0,1);**  **glVertex2d(1, 1);**  **glVertex2d(1, 0);**  **glVertex2d(0, 0);**  **glEnd();**  **}**  **void Board15(){**  **glBegin(GL\_QUADS);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(-1,1);**  **glVertex2d(0, 1);**  **glVertex2d(0, 0);**  **glVertex2d(-1, 0);**  **glEnd();**  **}**  **void Board16(){**  **glBegin(GL\_QUADS);**  **glColor3f(1.0f, 1.0f, 1.0f);**  **glVertex2d(-2,1);**  **glVertex2d(-1, 1);**  **glVertex2d(-1, 0);**  **glVertex2d(-2, 0);**  **glEnd();**  **}**  **void Lines(){**  **glBegin(GL\_LINE\_LOOP); // Each set of 4 vertices form a quad**  **glColor3f(0.0f, 0.0f, 0.0f); // Red**  **glVertex2d(-2,4);**  **glVertex2d(2, 4);**  **glVertex2d(2, 0);**  **glVertex2d(-2, 0);**  **glEnd();**  **}**  **void display()**  **{**  **glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to white and opaque**  **glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)**  **glLineWidth(1.5);**  **Board1();**  **Board2();**  **Board3();**  **Board5();**  **Board6();**  **Board7();**  **Board8();**  **Board9();**  **Board10();**  **Board11();**  **Board12();**  **Board13();Board14();Board15();Board16();**  **Lines();**  **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(720, 720);**  **glutCreateWindow("Cheeseboard(manual)"); // Create a window with the given title**  **// Set the window's initial width & height**  **glutDisplayFunc(display);**  **gluOrtho2D(-6,6,-6,6);// Register display callback handler for window re-paint**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question- 3**  Create the batman logo given below-  Batman Logo Clip Art N14 free image download |
| **Graph Plot (Picture)-** |
| **Code-#include <windows.h> // for MS Windows**  **#include <GL/glut.h>**  **#include <cmath> // GLUT, include glu.h and gl.h**  **void vat(){**  **glBegin(GL\_POLYGON);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(-3.22,1.58);**  **glVertex2d(-1.02,1.56);**  **glVertex2d(0,1);**  **glVertex2d(1.18,1.62);**  **glVertex2d(3.4,1.6);**  **glVertex2d(2.42,0.28);**  **glVertex2d(0.08,-0.62);**  **glVertex2d(-2.26,0.28);**  **glEnd();**  **}**  **void circle(float radius, float xc, float yc, float r, float g, float b)**  **{**  **glBegin(GL\_POLYGON);// Draw a Red 1x1 Square centered at origin**  **for(int i=0;i<200;i++)**  **{**  **glColor3f(r,g,b);**  **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+xc,y+yc);**  **}**  **glEnd();**  **}**  **void circles(){**  **circle(1.5, -3.32, 0.58, 255,255,255);**  **circle(1.6, -1.1, -1.4, 255,255,255);**  **circle(1.3, 1.66, -0.9, 255,255,255);**  **circle(1.19, 3.58, 0.66, 255,255,255);**  **circle(1 ,0.3, 2, 255,255,255);**  **circle(0.3,0.27, 1, 0,0,0);**  **}**  **void Line(){**  **glBegin(GL\_LINES);**  **glColor3f(0.0f, 0.0f, 0.0f);**  **glVertex2d(0.16,1.2);**  **glVertex2d(0.16,1.43324);**  **glVertex2d(0.36,1.2);**  **glVertex2d(0.36,1.43324);**  **glEnd();**  **}**  **void display()**  **{**  **glClearColor(1.0f, 1.0f, 1.0f, 0.0f); // Set background color to white and opaque**  **glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)**  **glLineWidth(1.5);**  **vat();**  **circles();**  **Line();**  **glFlush(); // Render now**  **}**  **int main(int argc, char\*\* argv)**  **{**  **glutInit(&argc, argv); // Initialize GLUT**  **glutInitWindowSize(720, 720);**  **glutCreateWindow("Bat"); // Create a window with the given title**  **// Set the window's initial width & height**  **glutDisplayFunc(display);**  **gluOrtho2D(-6,6,-6,6);// Register display callback handler for window re-paint**  **glutMainLoop(); // Enter the event-processing loop**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |