NATIONAL UNIVERSITY OF MODERN LANGUAGES

Human Computer Interaction lab Report – 6

NAME: Abdul Rafay

ROLL NO: FL23791

PROGRAM: BSSE (5th Semester)

COURSE: HUMAN COMPUTER INTERACTION

SUBMITTED TO: Sir Khateeb khan

DATE: 15 Oct 2025 **DAY: Tuesday**

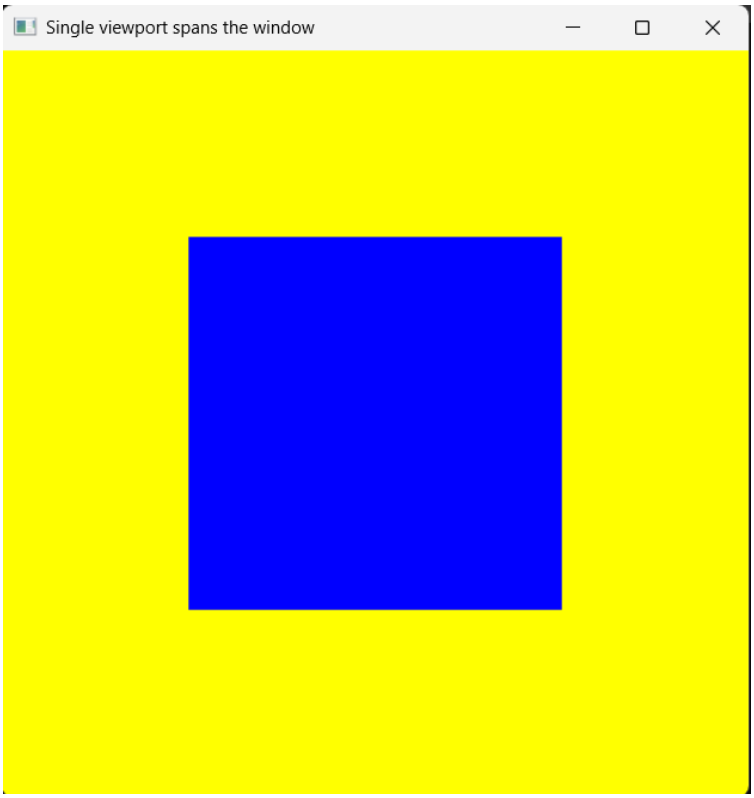
Class Task: Program to capture the create a viewport in the output window.

```
#include <GL/glut.h> // Header File For The GLut Library

void draw() {
   // Make background color yellow
   glClearColor(1.0, 1.0, 0.0, 0.0);
   glClear(GL_COLOR_BUFFER_BIT);
   // Sets up viewport spanning the entire window (500x500)
   glViewport(0, 0, 500, 500);
   // Sets up the PROJECTION matrix
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(-500.0, 500.0, -500.0, 500.0); // sets up world window (left, right, bottom, top)
   // Draw BLUE rectangle
   glColor3f(0, 0, 1);
   glRectf(-250.0, -250.0, 250.0, 250.0);
   // Display rectangles
   glutSwapBuffers();
}
// Keyboard method to allow ESC key to quit
void keyboard(unsigned char key, int x, int y) {
   if (key == 27) exit(0);
}
int main(int argc, char** argv) {
   glutInit(&argc, argv);
   // Double Buffered RGB display
   glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
   // Set window size
   glutInitWindowSize(500, 500);
```

```
    glutCreateWindow("Single viewport spans the window");
    // Declare the display and keyboard functions
    glutDisplayFunc(draw);
    glutKeyboardFunc(keyboard);
    // Start the Main Loop
    glutMainLoop();
    return 0;
}
```

Output :



Lab Tasks

1. Draw a static **blue rectangle** only in the **bottom-left** viewport using world coordinates.

```
#include <GL/glut.h> // GLUT header
#include <GL/glu.h>  // GLU header for gluOrtho2D
#include <stdlib.h>  // for exit()
```

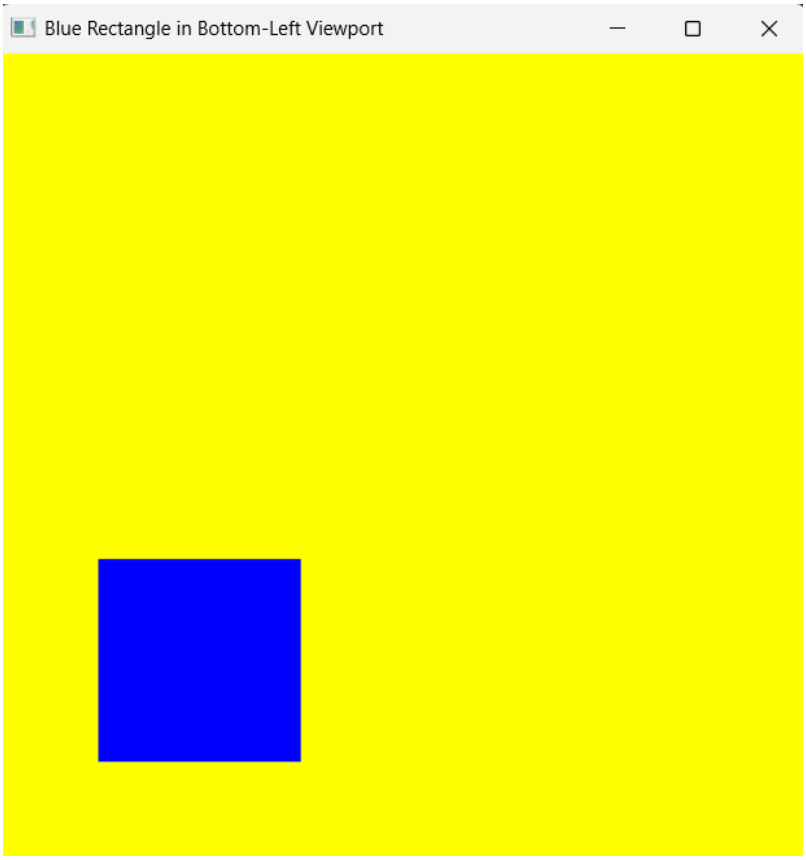
```
void draw() {
   // Clear the background with yellow color
   glClearColor(1.0, 1.0, 0.0, 1.0);
   glClear(GL_COLOR_BUFFER_BIT);
   // ----------------------------
   // Bottom-left viewport setup
   // ----------------------------
   glViewport(0, 0, 250, 250);  // bottom-left quarter of 500x500 window
   // Set up projection for this viewport
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(-500.0, 500.0, -500.0, 500.0); // world coordinates
   // Set modelview (for drawing)
   glMatrixMode(GL_MODELVIEW);
   glLoadIdentity();
   // Draw blue rectangle (in world coordinates)
   glColor3f(0.0, 0.0, 1.0); // blue color
   glRectf(-250.0, -250.0, 250.0, 250.0);
   // Swap buffers to display
   glutSwapBuffers();
}
// Keyboard: press ESC to quit
void keyboard(unsigned char key, int x, int y) {
   if (key == 27) exit(0);
}
int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
   glutInitWindowSize(500, 500);
```

```
    glutCreateWindow("Blue Rectangle in Bottom-Left Viewport");

    glutDisplayFunc(draw);
    glutKeyboardFunc(keyboard);

    glutMainLoop();
    return 0;
}
```

Output:



2. **Create** two viewports **side-by-side:**

   Left: shows the full world **(-100 to 100).**

   Right: shows a zoomed-in **view (-50 to 50).**

```
#include <GL/glut.h>
```

```
#include <GL/glu.h>
#include <stdlib.h>

// Draw a simple shape (a blue rectangle with coordinate axes)
void drawScene()
{
   // Draw coordinate axes (in gray)
   glColor3f(0.6f, 0.6f, 0.6f);
   glBegin(GL_LINES);
   glVertex2f(-100.0f, 0.0f);
   glVertex2f(100.0f, 0.0f);
   glVertex2f(0.0f, -100.0f);
   glVertex2f(0.0f, 100.0f);
   glEnd();
   // Draw a blue rectangle in world coordinates
   glColor3f(0.0f, 0.0f, 1.0f);
   glRectf(-30.0f, -20.0f, 30.0f, 20.0f);
}
void display()
{
   glClearColor(1.0, 1.0, 0.0, 1.0); // Yellow background
   glClear(GL_COLOR_BUFFER_BIT);
   // ----------------------------
   // Left Viewport (Full World)
   // ----------------------------
   glViewport(0, 0, 250, 500); // Left half of the window
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluOrtho2D(-100.0, 100.0, -100.0, 100.0); // full world view
```

```
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    drawScene(); // draw same scene
    // ---------------------------
    // Right Viewport (Zoomed-In)
    // ---------------------------
    glViewport(250, 0, 250, 500); // Right half of the window
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-50.0, 50.0, -50.0, 50.0); // zoomed-in view
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    drawScene(); // draw same scene again (but zoomed)
    glutSwapBuffers();
}
// Keyboard handler (ESC to exit)
void keyboard(unsigned char key, int x, int y)
{
    if (key == 27)
        exit(0);
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_RGB | GLUT_DOUBLE);
    glutInitWindowSize(500, 500);
    glutCreateWindow("Two Side-by-Side Viewports");
```

```
    glutDisplayFunc(display);
    glutKeyboardFunc(keyboard);

    glutMainLoop();
    return 0;
}
```

Output: