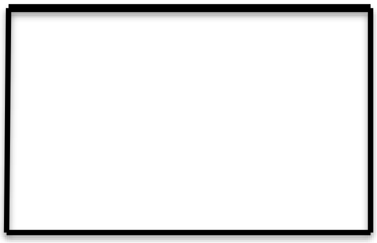
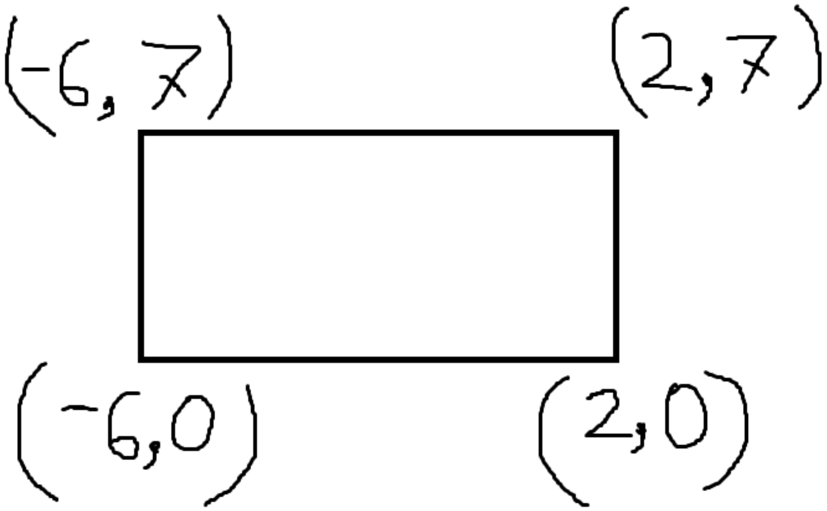
Lab Taks-1

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)-

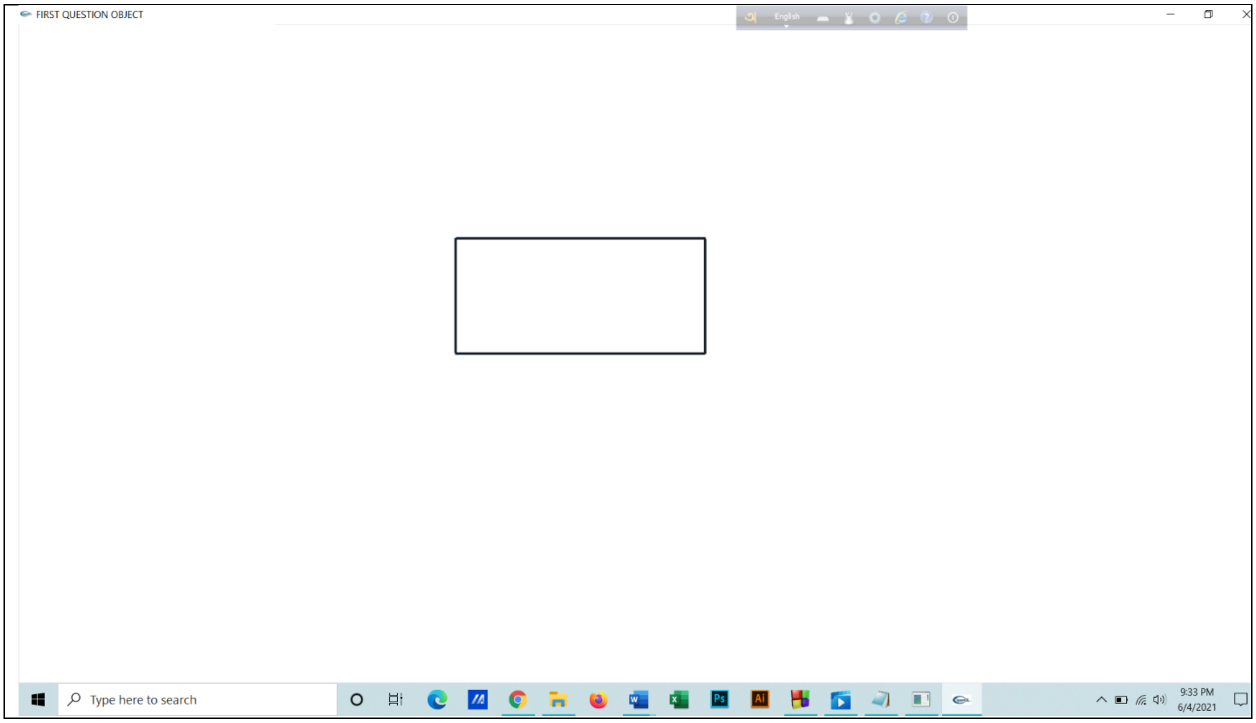
First Question Object



Code-

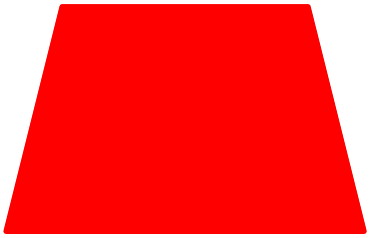
```
#include <windows.h> // for MS Windows
#include <GL/glut.h> // GLUT, include glu.h and gl.h

/* Handler for window-repaint event. Call back when the window first appears and
whenever the window needs to be re-painted. */
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)
glLineWidth(3);
// Draw a Red 1x1 Square centered at origin
glBegin(GL_LINES);
glColor3ub(23, 32, 42 ); // Each set of 4 vertices form a quad// Red // x, y
glVertex2f(-6.0f,0.0f);
glVertex2f(2.0f,0.0f);
glEnd();

glBegin(GL_LINES);
glColor3ub(23, 32, 42 ); // Each set of 4 vertices form a quad// Red // x, y
glVertex2f(2.0f,0.0f);
glVertex2f(2.0f,7.0f);
glEnd();
glBegin(GL_LINES);
glColor3ub(23, 32, 42 ); // Each set of 4 vertices form a quad// Red // x, y
glVertex2f(2.0f,7.0f);
glVertex2f(-6.0f,7.0f);
glEnd();

glBegin(GL_LINES);
glColor3ub(23, 32, 42 ); // Each set of 4 vertices form a quad// Red // x, y
glVertex2f(-6.0f,7.0f);
glVertex2f(-6.0f,0.0f);
glEnd();
 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
glutCreateWindow("FIRST QUESTION OBJECT"); // Create a window with the given title
glutInitWindowSize(320,320);
gluOrtho2D(-20,20,-20,20); // Set the window's initial width & height
glutDisplayFunc(display); // Register display callback handler for window re-paint
glutMainLoop(); // Enter the event-processing loop
return 0;
}
```

Output Screenshot (Full Screen)-



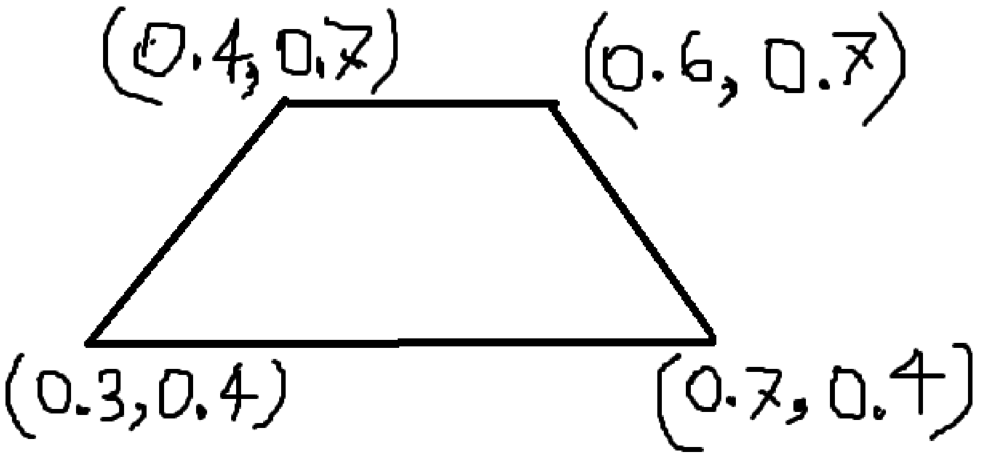
Question-

Draw the object-



Graph Plot (Picture)-

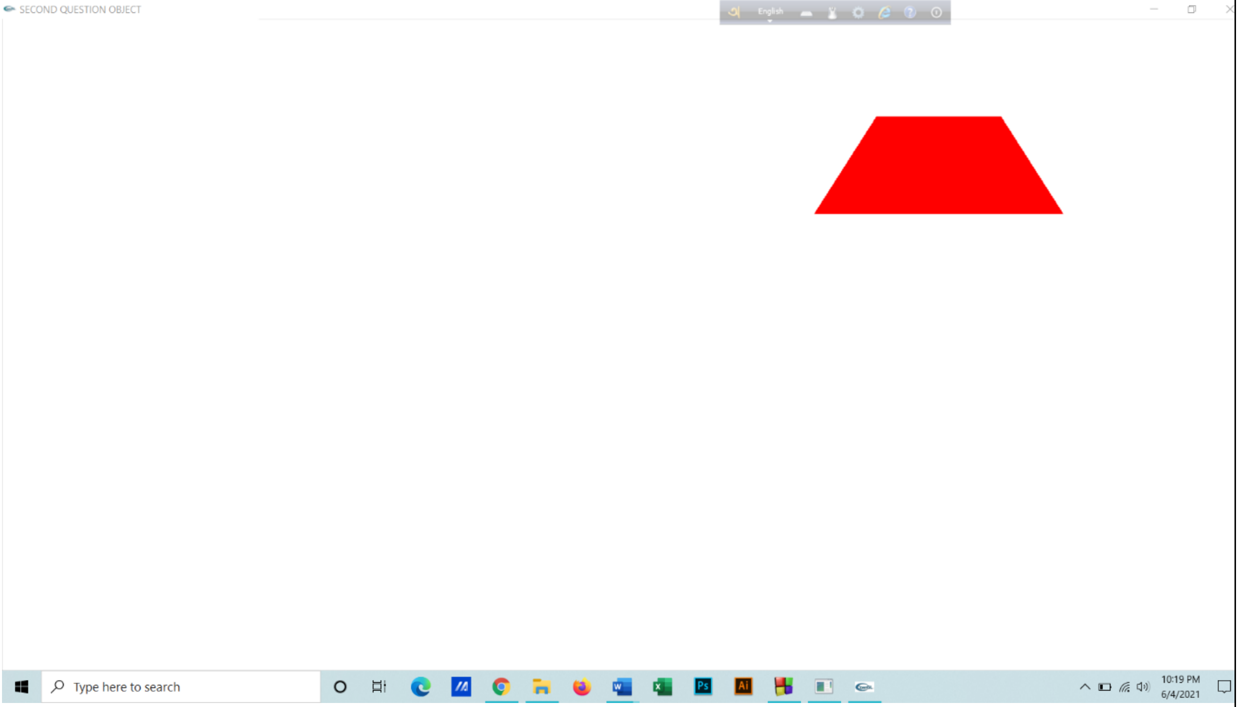
Second Question Object



Code-

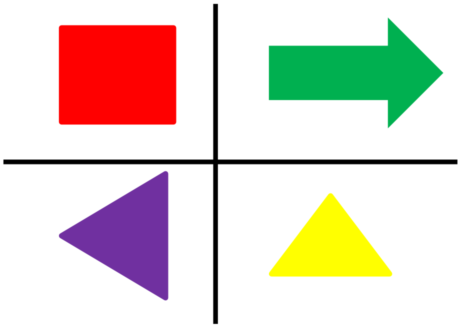
```
#include <windows.h>  // for MS Windows
#include <GL/glut.h>  // GLUT, include glu.h and gl.h

/* Initialize OpenGL Graphics */
void initGL() {
        // Set "clearing" or background color
        glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Black and opaque
}

/* Handler for window-repaint event. Call back when the window first appears and
whenever the window needs to be re-painted. */
void display() {
        glClear(GL_COLOR_BUFFER_BIT);   // Clear the color buffer with current clearing
color


        glBegin(GL_POLYGON);           // These vertices form a closed polygon
        glColor3f(1.0f, 0.0f, 0.0f); // Yellow


        glVertex2f(0.7f, 0.4f);
        glVertex2f(0.6f, 0.7f);
        glVertex2f(0.4f, 0.7f);
        glVertex2f(0.3f, 0.4f);
        glEnd();



        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
        glutCreateWindow("SECOND QUESTION OBJECT");  // Create window with the given
title
        glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutDisplayFunc(display);       // Register callback handler for window re-paint event
        initGL();                       // Our own OpenGL initialization
        glutMainLoop();                 // Enter the event-processing loop
        return 0;
}
```

Output Screenshot (Full Screen)-

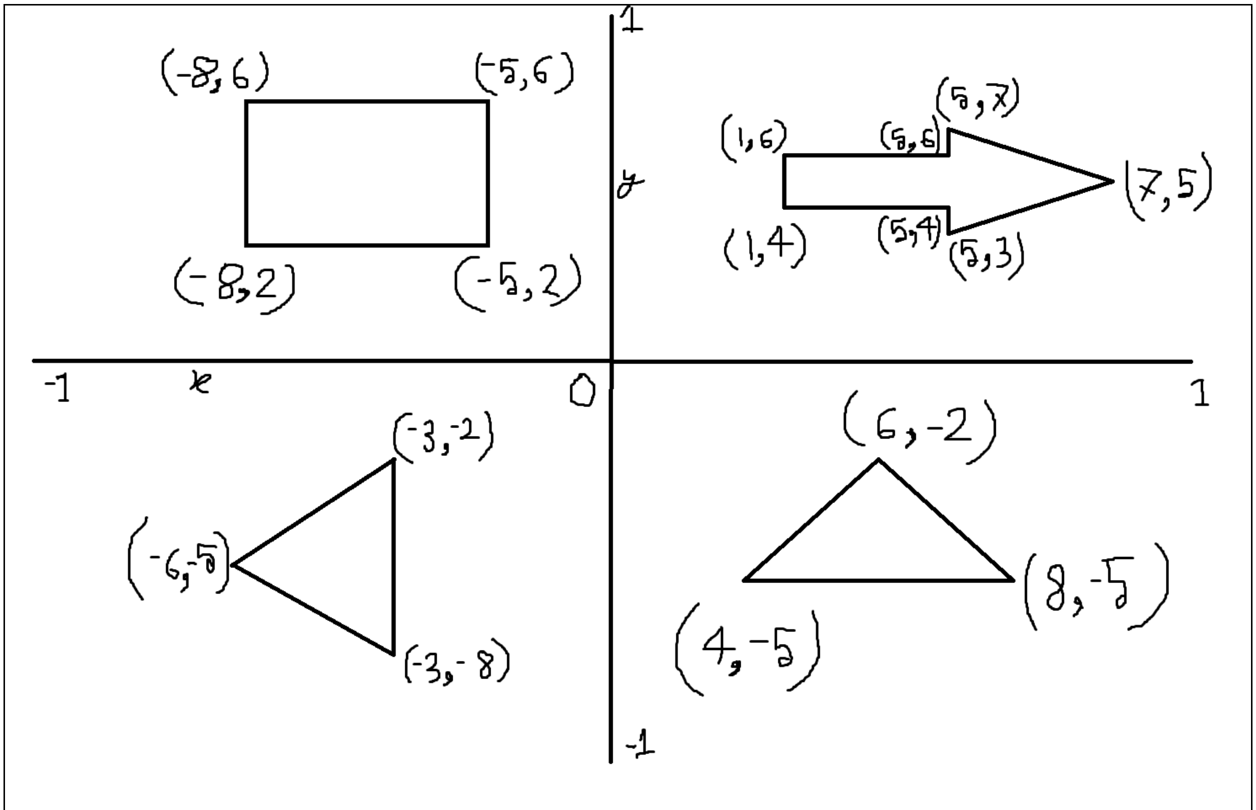


Question-

Draw the object-



Graph Plot (Picture)-



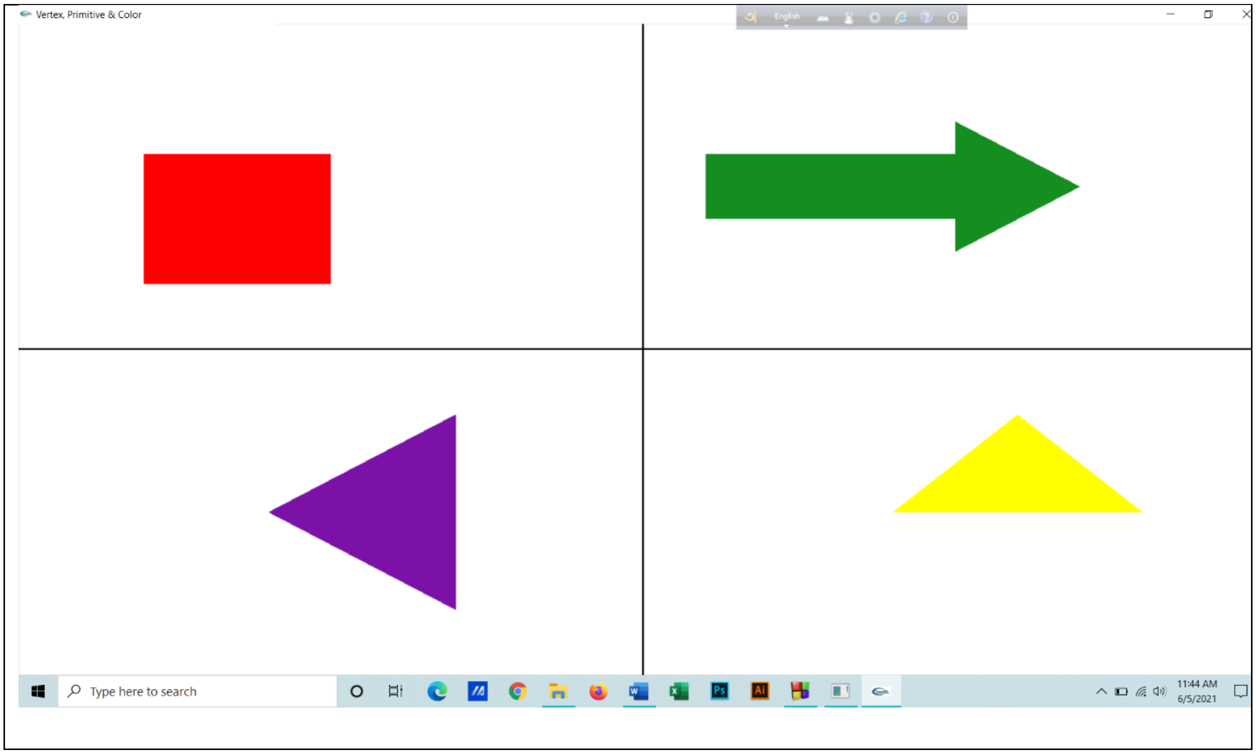
Code-

```
#include <windows.h>  // for MS Windows
#include <GL/glut.h>  // GLUT, include glu.h and gl.h

/* Initialize OpenGL Graphics */
void initGL() {
        // Set "clearing" or background color
        glClearColor(1.0f, 1.0f, 1.0f, 1.0f); // Black and opaque
}

/* Handler for window-repaint event. Call back when the window first appears and
whenever the window needs to be re-painted. */
void display() {
        glClear(GL_COLOR_BUFFER_BIT);   // Clear the color buffer with current clearing
color
glLineWidth(2);
        // Draw a Red 1x1 Square centered at origin
        glBegin(GL_LINES);              // Each set of 4 vertices form a quad
        glColor3f(0.0f, 0.0f, 0.0f); // Red

        glVertex2f(0.0f, 0.0f);    // x, y
        glVertex2f(1.0f, 0.0f);    // x, y
```

```
          glVertex2f(0.0f, 0.0f);    // x, y
          glVertex2f(0.0f, 1.0f);    // x, y

   glVertex2f(0.0f, 0.0f);    // x, y
          glVertex2f(-1.0f, 0.0f);    // x, y

          glVertex2f(0.0f, 0.0f);    // x, y
          glVertex2f(0.0f, -1.0f);

          glEnd();

          glBegin(GL_POLYGON);
glColor3ub(20, 142, 30);
glVertex2f(0.1f,0.6f);
glVertex2f(0.1f,0.4f);
glVertex2f(0.5f,0.4f);
glVertex2f(0.5f,0.6f);
glEnd();
glBegin(GL_POLYGON);
glColor3ub(20, 142, 30);
glVertex2f(0.5f,0.6f);
glVertex2f(0.5f,0.4f);
glVertex2f(0.5f,0.3f);
glVertex2f(0.7f,0.5f);
glVertex2f(0.5f,0.7f);

          glEnd();

           glBegin(GL_QUADS);              // Each set of 4 vertices form a quad

           glColor3f(1.0f, 0.0f, 0.0f); // Red

          glVertex2f(-0.5f, +0.2f);    // x, y
          glVertex2f(-0.5f, +0.6f);
          glVertex2f(-0.8f, +0.6f);    // x, y
          glVertex2f(-0.8f, +0.2f);

          glEnd();

                    // Draw a Red 1x1 Square centered at origin
          glBegin(GL_TRIANGLES);              // Each set of 4 vertices form a quad
          glColor3f(1.0f, 1.0f, 0.0f); // Red

          glVertex2f(0.6f, -0.2f);// x, y
```

```
        glVertex2f(0.4f,  -0.5f);
        glVertex2f(0.8f, -0.5f);

        glEnd();


   glBegin(GL_TRIANGLES);//
   glColor3ub(123, 17, 166);//rgb color picker

   glVertex2f(-0.3f, -0.2f);    // x, y
   glVertex2f(-0.6f, -0.5f);
        glVertex2f(-0.3f,-0.8f);

        glEnd();



        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
        glutCreateWindow("Vertex, Primitive & Color");  // Create window with the given
title
        glutInitWindowSize(320, 320);   // Set the window's initial width & height
        glutDisplayFunc(display);       // Register callback handler for window re-paint event
        initGL();                       // Our own OpenGL initialization
        glutMainLoop();                 // Enter the event-processing loop
        return 0;
}
```

Output Screenshot (Full Screen)-


Vertex, Primitive & Color
English
Type here to search
11:44 AM
6/5/2021