**Kathmandu University**

**Department of Computer Science and Engineering**

**Dhulikhel, Kavre**



A Report on ‘**Lab Work 1’** [COMP 342]

**Submitted by:**

Chandan Kumar Mahato (31)

III-year, II semester

**Submitted to**:

Mr. Dhiraj Shrestha

Department of Computer Science and Engineering

**Submission Date**: Mar 12, 2023

***Qlab1***

1. ***Get Familiar with the coordinate system and draw a flag of Nepal using the chosen graphics library.***

***Solution:***

<https://flagwebgl.netlify.app>

I have used JavaScript as the programming language and WebGL as the graphics library. **JavaScript** is a popular programming language that is used for web development and has the capability of creating interactive graphics and animations on the web. **WebGL** is a graphics library that enables high-performance 3D graphics library that enables high-performance 3D graphics rendering in web browser using JavaScript. These technologies provide a powerful platform for creating engaging and interactive graphics, which is essential for my lab work.

The code snippets for setting graphics environment in my chosen graphics library and programming language and display system resolution are as follow:

**Code snippets for setting graphics environment:**

**.html**

  <canvas class="display\_area" id="canvas"></canvas>

**.js**

*var* canvas = document.querySelector("canvas");

*var* gl = canvas.getContext("webgl");

if (!gl) {

  throw new *Error*("WebGL not supported");

}

// supported

**Code snippets to get display system resolution:**

*function* getResolution() {

  alert("Your screen resolution is: " +

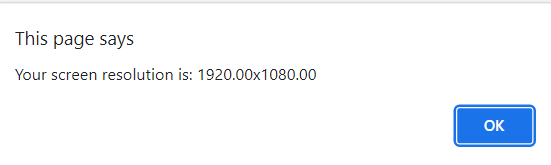
      (window.screen.width \* window.devicePixelRatio).toFixed(2) +

      "x" +(window.screen.height \* window.devicePixelRatio).toFixed(2)

  );

}

getResolution();



**Source Code:**

<https://github.com/ChandankMahato/Graphics_Lab_6th_Sem>

**flag.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

    <meta name="language" content="English" />

    <meta name="author" content="Chandan Kumar Mahato" />

    <title>Nepal Flag WebGL</title>

</head>

<body>

    <canvas class="display\_area" id="canvas"></canvas>

  <script src="./javaScript/sunData.js"></script>

    <script src="./javaScript/moonData.js"></script>

    <script src="./javaScript/data.js"></script>

    <script src="./javaScript/flag.js"></script>

</body>

</html>

**<Script/>**

**sunData.js**

*var* sunVertexData = [];

*function* createSunVertexList(*radius*, *xo*, *yo*) {

  for (i = 0; i < 360; i += 15) {

    sunVertexData.push(...[*xo*, *yo*, 0]);

    x = *radius* \* Math.cos((Math.PI / 180) \* i) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* i) + *yo*;

    sunVertexData.push(...[x, y, 0]);

    x = *radius* \* Math.cos((Math.PI / 180) \* (i + 15)) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* (i + 15)) + *yo*;

    sunVertexData.push(...[x, y, 0]);

  }

}

createSunVertexList(0.75, 1.25, 1.25);

*var* sunSpikeData = [];

*function* createSunSpikeList(*radius*, *xo*, *yo*) {

  midWidth = *radius* / 8;

  for (i = -15; i < 375; i += 30) {

    x = *radius* \* Math.cos((Math.PI / 180) \* i) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* i) + *yo*;

    sunSpikeData.push(...[x, y, 0]);

    x = *radius* \* Math.cos((Math.PI / 180) \* (i + 30)) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* (i + 30)) + *yo*;

    sunSpikeData.push(...[x, y, 0]);

    if (i <= 90) {

      x = *radius* \* Math.cos((Math.PI / 180) \* (i + 15)) + *xo* + midWidth;

      y = *radius* \* Math.sin((Math.PI / 180) \* (i + 15)) + *yo* + midWidth;

      sunSpikeData.push(...[x, y, 0]);

    } else if (i <= 180) {

      x = *radius* \* Math.cos((Math.PI / 180) \* (i + 15)) + *xo* - midWidth;

      y = *radius* \* Math.sin((Math.PI / 180) \* (i + 15)) + *yo* + midWidth;

      sunSpikeData.push(...[x, y, 0]);

    } else if (i <= 270) {

      x = *radius* \* Math.cos((Math.PI / 180) \* (i + 15)) + *xo* - midWidth;

      y = *radius* \* Math.sin((Math.PI / 180) \* (i + 15)) + *yo* - midWidth;

      sunSpikeData.push(...[x, y, 0]);

    } else if (i <= 360) {

      x = *radius* \* Math.cos((Math.PI / 180) \* (i + 15)) + *xo* + midWidth;

      y = *radius* \* Math.sin((Math.PI / 180) \* (i + 15)) + *yo* - midWidth;

      sunSpikeData.push(...[x, y, 0]);

    }}}

createSunSpikeList(0.75, 1.25, 1.25);

**moonData.js**

//moon

*var* moonVertexData = [];

*function* createMoonVertexList(*radius*, *xo*, *yo*) {

  for (i = 0; i > -180; i -= 15) {

    moonVertexData.push(...[*xo*, *yo*, 0]);

    x = *radius* \* Math.cos((Math.PI / 180) \* i) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* i) + *yo*;

    moonVertexData.push(...[x, y, 0]);

    x = *radius* \* Math.cos((Math.PI / 180) \* (i - 15)) + *xo*;

    y = *radius* \* Math.sin((Math.PI / 180) \* (i - 15)) + *yo*;

    moonVertexData.push(...[x, y, 0]);

  }

}

createMoonVertexList(0.75, 1.25, 4.25);

*var* moonSpikeData = [];

*function* createMoonSpikeList(*radius*, *xo*, *yo*) {

  gap = *radius* / 4;

  midWidth = gap / 2;

  leftX = *xo* - *radius*;

  leftY = *yo*;

  rightX = *xo* + gap - *radius*;

  rightY = *yo*;

  topX = leftX + midWidth;

  topY = *yo* + gap;

  for (i = 0; i < 8; i++) {

    moonSpikeData.push(

      ...[leftX, leftY, 0],

      ...[rightX, rightY, 0],

      ...[topX, topY, 0]

    );

    leftX = rightX;

    rightX = rightX + gap;

    topX = topX + gap;

  }

}

createMoonSpikeList(0.75, 1.25, 4.25);

**data.js**

//RED PART

t1 = [...[0, 0, 0], ...[0, 3.75, 0], ...[5, 0, 0]];

t2 = [...[0, 3, 0], ...[0, 6, 0], ...[5, 3, 0]];

//Blue PART

t3 = [...[5.75, 2.75, 0], ...[-0.25, 6.5, 0], ...[-0.25, 2.75, 0]];

t4 = [...[-0.25, -0.25, 0], ...[5.75, -0.25, 0], ...[-0.25, 4.25, 0]];

//processing data

*const* vertexData = [];

*const* dataArray = [

  ...t1,

  ...t2,

  ...t3,

  ...t4,

  ...sunVertexData,

  ...moonVertexData,

  ...moonSpikeData,

  ...sunSpikeData,

];

for (i = 0; i < dataArray.length; i++) {

  y = dataArray[i] / 6.5;

  vertexData.push(y);

}

**flag.js**

document.getElementById("resolution").innerHTML =

  "Your screen resolution is: " +

  (window.screen.width \* window.devicePixelRatio).toFixed(2) +

  "x" +

  (window.screen.height \* window.devicePixelRatio).toFixed(2);

*var* canvas = document.querySelector("canvas");

*var* gl = canvas.getContext("webgl");

if (!gl) {

  throw new *Error*("WebGL not supported");

}

*const* buffer = gl.createBuffer();

gl.bindBuffer(gl.ARRAY\_BUFFER, buffer);

gl.bufferData(gl.ARRAY\_BUFFER, new *Float32Array*(vertexData), gl.STATIC\_DRAW);

*const* vertexShader = gl.createShader(gl.VERTEX\_SHADER);

gl.shaderSource(

  vertexShader,

  `attribute vec3 position;void main() {gl\_Position = vec4(position, 1);}`

);

gl.compileShader(vertexShader);

*function* Triangles(*color*, *start*, *end*) {

*const* fragmentShader = gl.createShader(gl.FRAGMENT\_SHADER);

  gl.shaderSource(fragmentShader, `${*color*}`);

  gl.compileShader(fragmentShader);

*const* program = gl.createProgram();

  gl.attachShader(program, vertexShader);

  gl.attachShader(program, fragmentShader);

  gl.linkProgram(program);

*const* positionLocation = gl.getAttribLocation(program, `position`);

  gl.enableVertexAttribArray(positionLocation);

  gl.vertexAttribPointer(positionLocation, 3, gl.FLOAT, false, 0, 0);

  gl.useProgram(program);

  for (*let* i = *start*; i <= *end*; i += 3) {

    gl.drawArrays(gl.TRIANGLES, i, 3);

  }

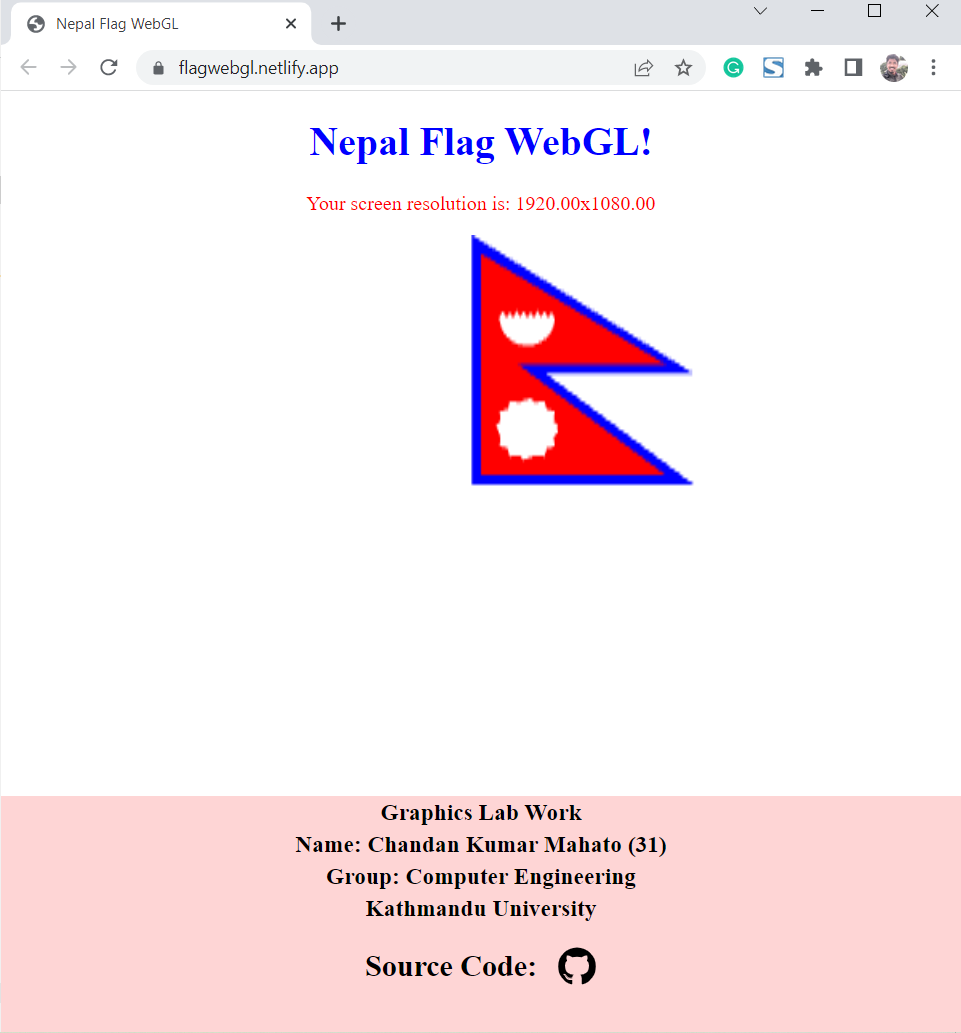
}

Triangles(`void main() {gl\_FragColor = vec4(0, 0, 1, 1);}`, 6, 9); //Blue

Triangles(`void main() {gl\_FragColor = vec4(1, 0, 0, 1);}`, 0, 3); //Red

Triangles(`void main() {gl\_FragColor = vec4(1, 1, 1, 1);}`, 12, 230); //White

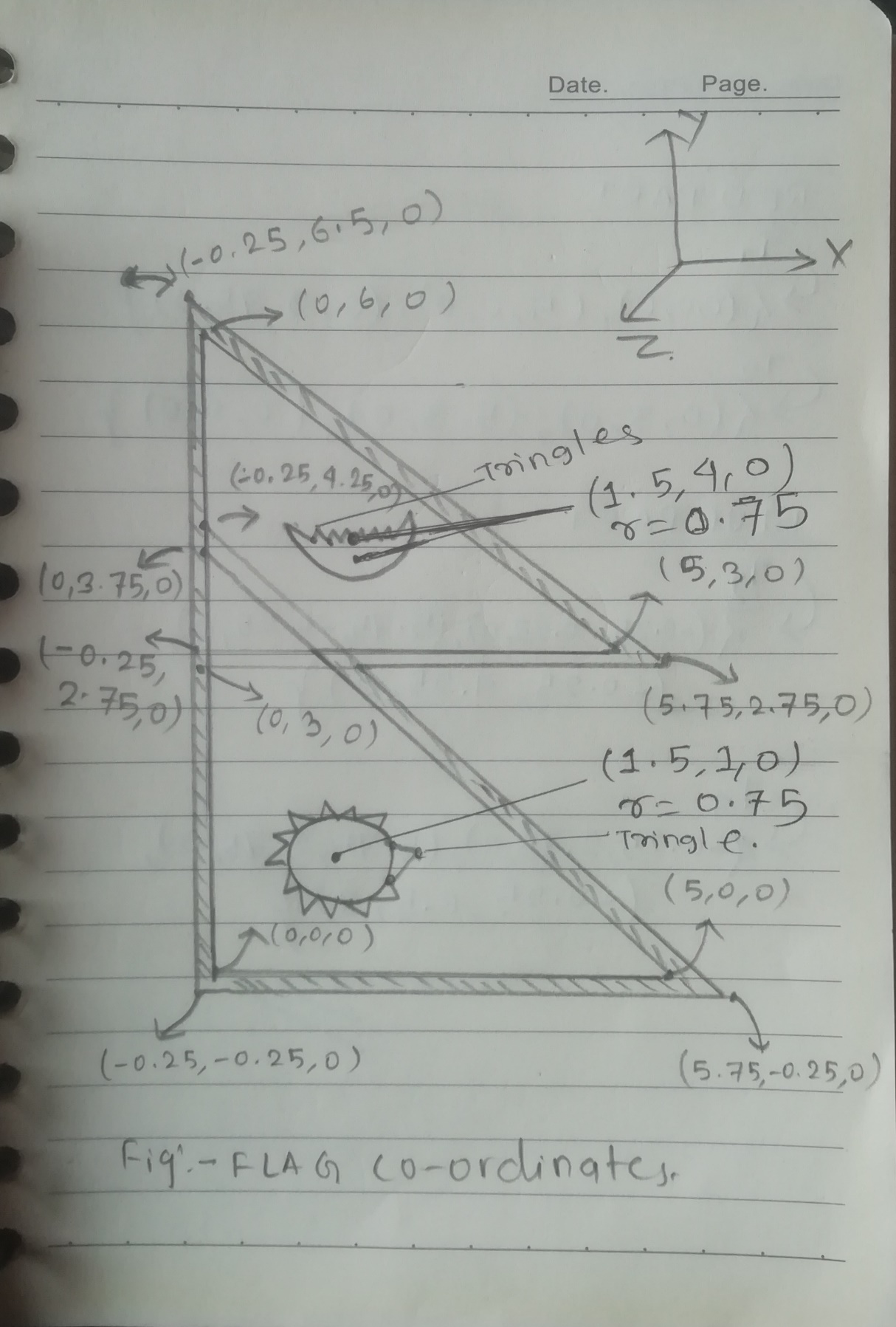
**Output:**

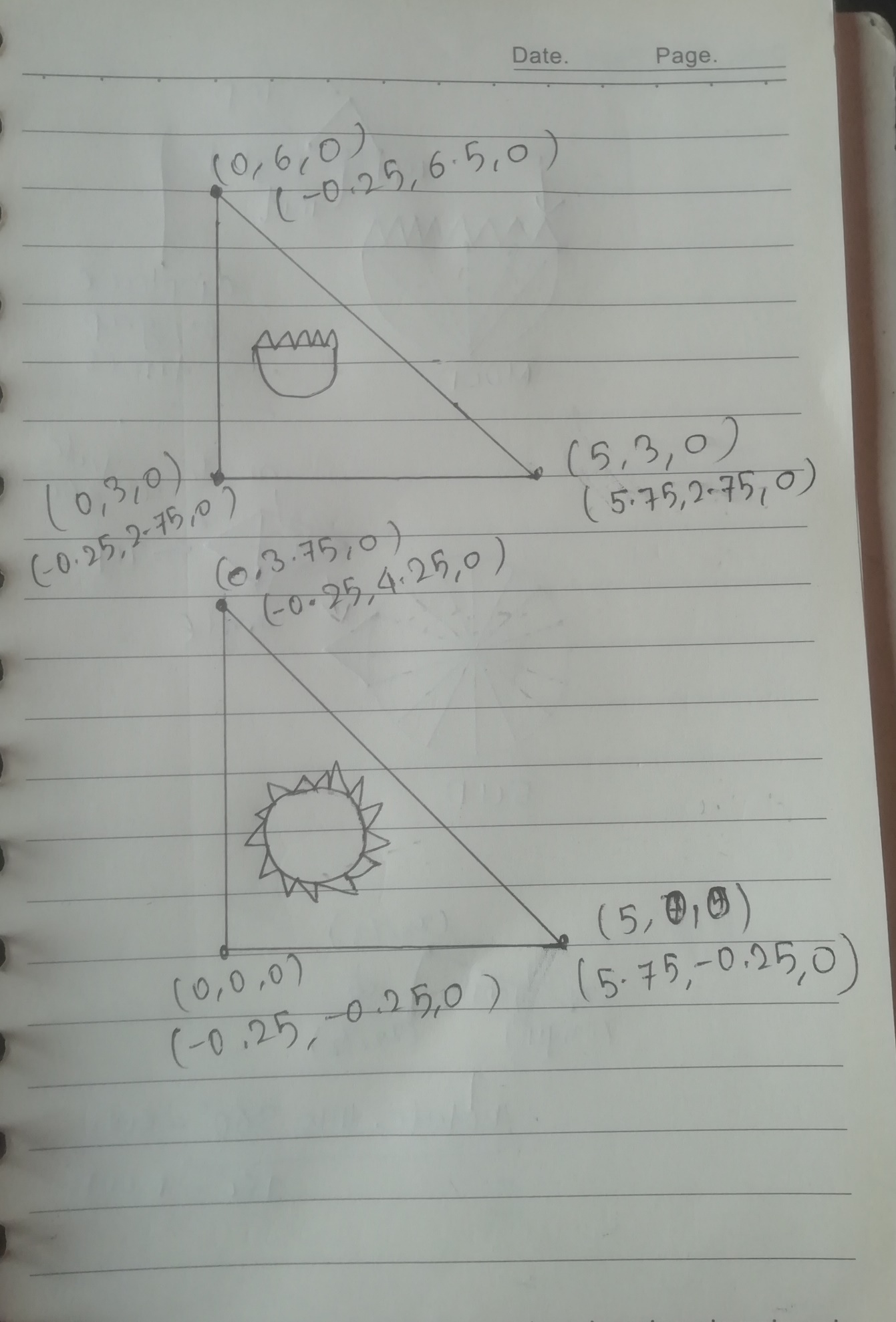
****

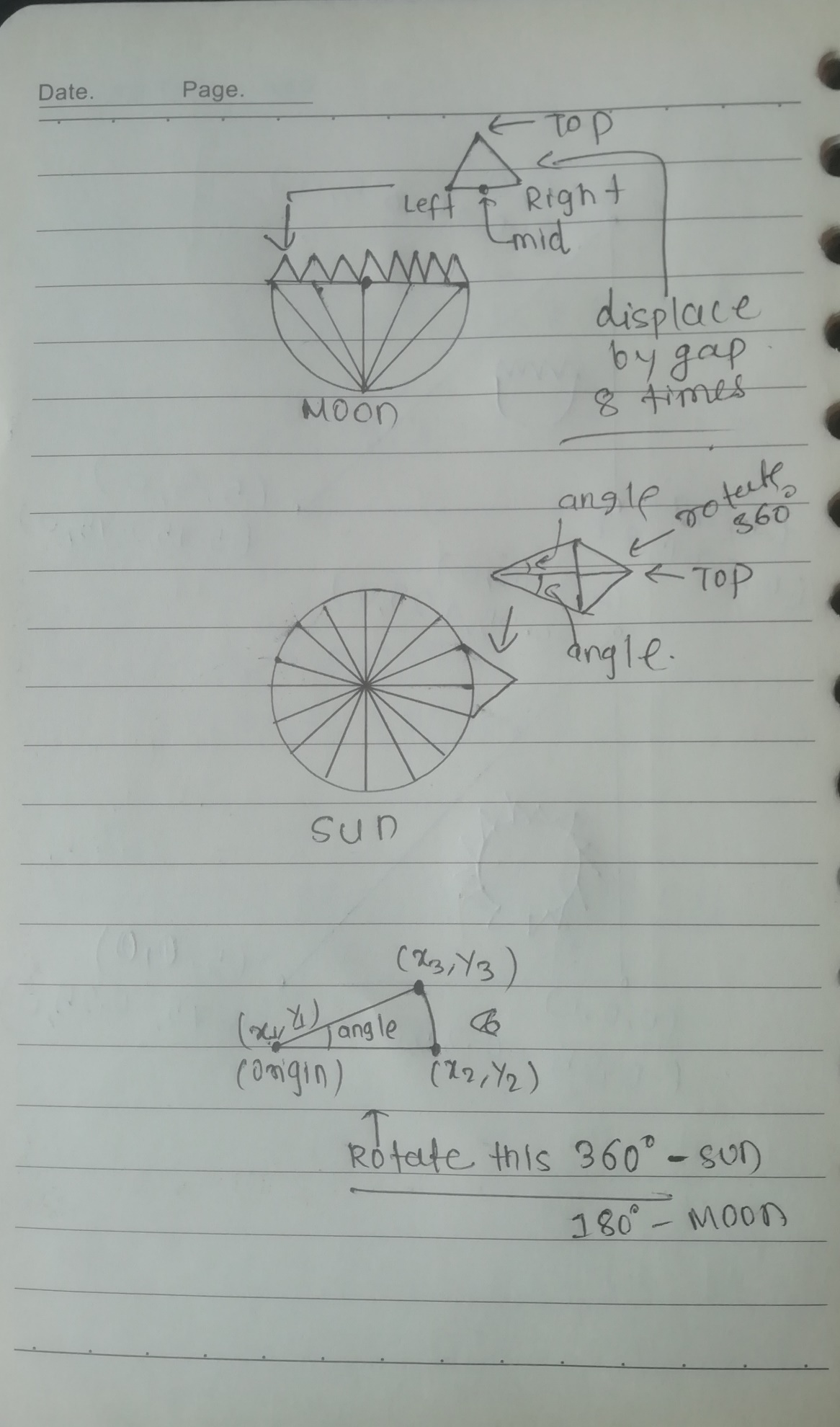
**Conclusion**

By getting familiar with the coordinate system and utilizing the graphics geometrical functions and classes provided by our chosen graphics library, I was able to successfully draw the flag of Nepal. This exercise helped me to develop a better understanding of the coordinate system and the use of graphics geometrical functions and classes to create visually appealing graphics.

**Rough Sketch**

****

****

****