

1. Mention the name of Programming language and Graphics Library you are using this semester for performing your Computer Graphics Lab and Project.

Ans: To perform my Computer Graphics Lab and Project, I will be using “**JavaScript**” as my Programming Language alongside “**p5.js**” as Graphic Library.

2. Write the code snippets for setting graphics environment in your chosen graphics library and display the resolution of your display system through functions/classes provided by your graphics library.

Ans: Here’s the code snippet that sets up the graphics environment in *p5.js*:

```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6
7     <title>Sketch</title>
8
9     <link rel="stylesheet" type="text/css" href="style.css">
10
11    <script src="libraries/p5.min.js"></script>
12    <script src="libraries/p5.sound.min.js"></script>
13  </head>
14
15  <body>
16    <script src="sketch.js"></script>
17  </body>
18 </html>
```

*index.html*

```
1 function setup() {
2   createCanvas(windowWidth, windowHeight);
3 }
4
5 function draw() {
6   background(220);
7 }
```

*sketch.js*

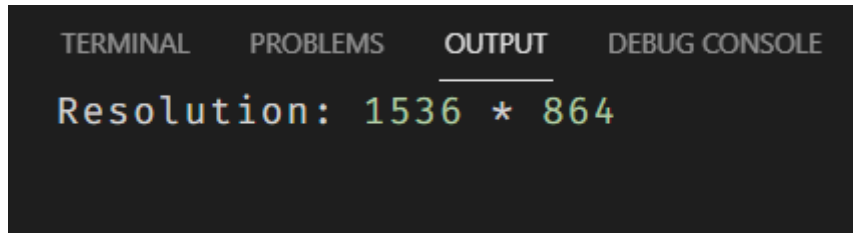
Below function prints the resolution of my display system:

```

12 class MyResolution {
13     /** Prints the resolution of the display. */
14     myScreenResolution() {
15         print("Resolution: " + displayWidth + " * " + displayHeight);
16     }
17 }

```

The output of the function is shown below:



The screenshot shows a dark-themed IDE window with tabs for 'TERMINAL', 'PROBLEMS', 'OUTPUT', and 'DEBUG CONSOLE'. The 'OUTPUT' tab is active, displaying the text 'Resolution: 1536 \* 864' in a monospaced font.

3. Get Familiar with the coordinate system and draw a flag of Nepal using the chosen Graphics geometrical functions/classes provided by your chosen graphics library and also colour the flag accordingly.

Ans: I have divided my code into multiple modules for easy understanding of the program code. Firstly, I modified my *draw()* function to call a *drawFlag()* that calls the functions to draw each part of the flag.

```

1  ✓ function setup() {
2      createCanvas(windowWidth, windowHeight);
3      var r1 = new MyResolution();
4      myScreenResolution();
5  }
6
7  ✓ function draw() {
8      background(220);
9      drawFlag();
10 }

```

```

19 function drawFlag() {
20     drawLayout();
21     drawMoon();
22     drawSun();
23 }

```

I first drew the triangles in the flag and filled it with a Crimson (#DC143C) color and provided it with a blue (#003893) stroke of weight 8. Hence, the first part of the flag was completed.

```

25  /** Draws the overall color-filled layout of the flag with border stroke.*/
26  function drawLayout() {
27      fill(220, 20, 60);
28      strokeWeight(8);
29      stroke(0, 56, 147);
30      strokeJoin(MITER);
31      beginShape();
32      vertex(30, 30);
33      vertex(350, 250);
34      vertex(140, 250);
35      vertex(330, 460);
36      vertex(30, 460);
37      endShape(CLOSE);
38  }

```

Afterwards, I created a function to draw the pointy edges in the sun and moon. Here the 'x' and 'y' parameter determine the center position of the star. Similarly, 'innerRadius' and 'outerRadius' are the radius of the inside edges and outside edges respectively. 'count' represents the number of edges to create and 'startAngle' and 'endAngle' are the angles at which the star should start and end respectively.

```

73  function drawSharpEdges(
74      x,
75      y,
76      innerRadius,
77      outerRadius,
78      count,
79      startAngle,
80      endAngle
81  ) {
82      if (!(startAngle || endAngle)) {
83          startAngle = 0;
84          endAngle = radians(360);
85      }
86
87      var angle = (endAngle - startAngle) / count;
88      var halved = angle / 2;
89
90      for (var i = endAngle; i ≥ startAngle; i -= angle) {
91          var x1 = x - cos(i + halved) * innerRadius;
92          var y1 = y - sin(i + halved) * innerRadius;
93          vertex(x1, y1);
94          var x2 = x - cos(i) * outerRadius;
95          var y2 = y - sin(i) * outerRadius;
96          vertex(x2, y2);
97      }
98  }

```

As the star was completed, I started creating the moon. For the crescent bottom of the moon, I used Bezier curve. Then I added the '*drawSharpEdges()*' function to add other parts of the moon in the flag. I also filled the entire moon with a white (#FFFFFF) color.

```

40  /** Draws a moon at the upper half triangle of the flag. */
41  function drawMoon() {
42      fill(255, 255, 255);
43      noStroke();
44      beginShape();
45      vertex(175, 175);
46      bezierVertex(150, 230, 60, 200, 55, 175);
47      bezierVertex(70, 250, 150, 250, 170, 193);
48      vertex(175, 175);
49      endShape(CLOSE);
50      beginShape();
51      drawSharpEdges(115, 189, 20, 30, 9, radians(-45), radians(270 - 45));
52      endShape(CLOSE);
53  }

```

Creating the sun wasn't a difficult task after I created the '*drawSharpEdges()*' function. I only had to pass the required parameter values and fill it with a white color. Therefore, I completed the sun as well in my flag.

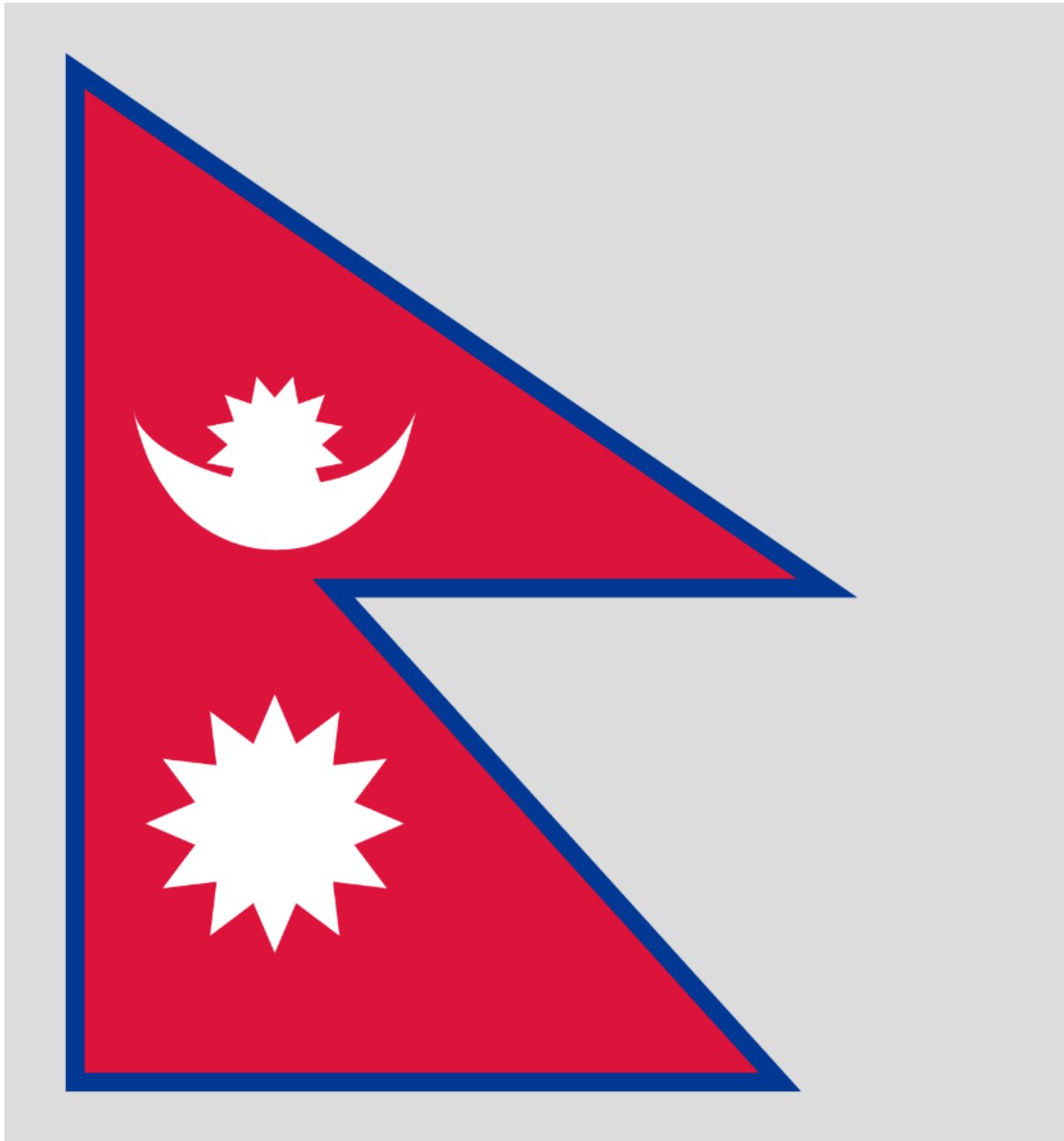
```

55  /** Draws a sun at the lower half triangle of the flag. */
56  function drawSun() {
57      fill(255, 255, 255);
58      beginShape();
59      drawSharpEdges(115, 350, 35, 55, 12);
60      endShape(CLOSE);
61  }

```

Hence, I finished the coding part for drawing the flag of Nepal.

Here's the final output of my program:



Here I have attached a link to my code files on GitHub:

[Source Code](#)