CSC 240: Computer Graphics
Midterm: Spring 2023
Friday, March 24 – Sunday, March 26

- This is a self-scheduled, closed book exam.
- While completing the exam you are allowed to use one page of notes, 8.5x11, both sides.
- Testing hours are Friday 3:00-9:00 pm, Saturday and Sunday 12:00-6:00 pm.
- You have two hours to complete the exam from the time you sign it out.
- Students with accommodations allowing extra time can compute their time accordingly.
- Exams must be turned in by the end of the testing window, so please plan ahead.
- Students with accommodations for individual space can reserve a room in advance.
- If you are unable to make progress on any part of the exam, tell me what you tried: describe your thought process. I may be able to grant partial credit.
- The problems on this exam are worth a total of 54 points.
- There are two pages of scratch paper at the end of this packet.
- When your exam is complete, before submitting it, please copy, sign, and date the statement below:

  "I certify that my work on this exam adheres to the Smith Honor Code and the instructions given above."

____

____

____

____

____

Signed: ______________________________ Date: ______________

Curves (10 points)

1.1. Fill in the missing spots in the table below.

Order	Description	Maximum bends	Control Points
Third			
		4	
			2
	Quadratic		

1.2. Compute the point at $t = 0.6$ for the curve with control points (0,2), (4,0), and (2,2), specified in sequential order.

Animation (10 points)

2. As written, the page below displays a ball that does not move. Now consider each of the possible replacements for the `draw()` function. Predict the motion that will be observed in each case, selecting from options (a)-(f) below.

```
<!DOCTYPE html>
<html>
<head>
<title>Transformations</title>
<script>
    var canvas;    // DOM object corresponding to the canvas
    var graphics;  // 2D graphics context for drawing on the canvas
    var count = 0; // counter

    function drawBall() {
        for (let i = 0; i < 3; i++) {
            graphics.fillStyle = "blue";
            graphics.beginPath();
            graphics.moveTo(0,0);
            graphics.arc(0,0, 25, 2*i*Math.PI/3, (2*i+1)*Math.PI/3);
            graphics.lineTo(0,0);
            graphics.fill();
            graphics.fillStyle = "yellow";
            graphics.beginPath();
            graphics.moveTo(0,0);
            graphics.arc(0,0, 25, (2*i+1)*Math.PI/3, (2*i+2)*Math.PI/3);
            graphics.lineTo(0,0);
            graphics.fill();
            graphics.strokeStyle = "black";
            graphics.beginPath();
```

```
                graphics.arc(0,0, 25, 0, 2*Math.PI);
                graphics.stroke();
            }
        }

        function clearCanvas() {
            graphics.save();
            graphics.setTransform(1,0,0,1,0,0);
            graphics.clearRect(0, 0, canvas.width, canvas.height);
            graphics.restore();
        }

        function draw() {
            clearCanvas();
            drawBall();
            count++;
        }

        function init() {
            canvas = document.getElementById("theCanvas");
            graphics = canvas.getContext("2d");
            graphics.translate(100,100);
            setInterval(draw,40);
        }
    </script>
    </head>
    <body onload="init()">
        <canvas id="theCanvas" width="200" height="200"></canvas>
    </body>
    </html>
```

Possible behaviors:

(a) Slides to the right without rolling
(b) Moves to the right while rolling
(c) Moves in a fixed circle
(d) Spirals outward
(e) Doesn't move
(f) Does something else (none of the options above) – describe the behavior

2.1. First replacement for `draw()`. Predicted behavior: __________

```
function draw() {
    clearCanvas();
    graphics.save();
    graphics.translate(2*count,0);
    graphics.rotate(0.1*count,0);
    drawBall();
    graphics.restore();
    count++;
}
```

2.2. Second replacement for `draw()`. Predicted behavior: __________

```
function draw() {
    clearCanvas();
    drawBall();
    graphics.translate(2,0);
}
```

2.3. Third replacement for `draw()`. Predicted behavior: __________

```
function draw() {
    clearCanvas();
    graphics.save();
    drawBall();
    graphics.translate(2*count,0);
    graphics.rotate(0.1*count,0);
    graphics.restore();
    count++;
}
```

2.4. Fourth replacement for `draw()`. Predicted behavior: __________

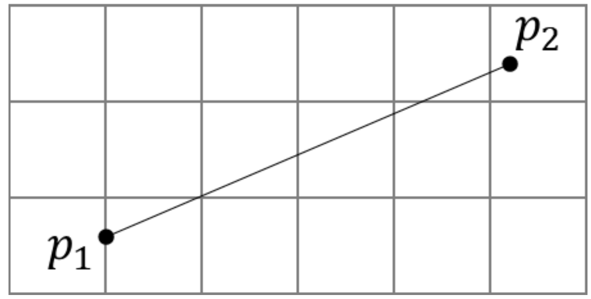
```
function draw() {
    clearCanvas();
    graphics.save();
    graphics.translate(2*count,0);
    graphics.rotate(0.1*count,0);
    graphics.restore();
    drawBall();
    count++;
}
```

2.5. Fifth replacement for `draw()`. Predicted behavior: __________

```
function draw() {
    clearCanvas();
    drawBall();
    graphics.translate(2,0);
    graphics.rotate(0.1,0);
}
```

Lines (12 points)

3. Consider the line endpoints labeled p_1 and p_2 at right. Note that each square in the figure is one pixel, and the y axis points upwards.



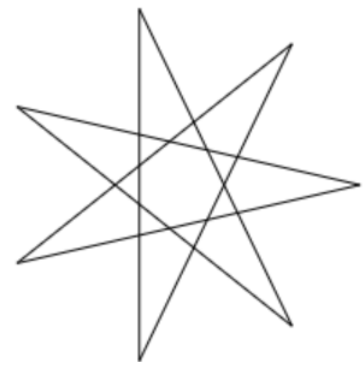
   3.1. In the figure, identify all the pixels that would be filled by the simple line algorithm by drawing an X in each one.

   3.2. In the figure, identify all the points (accurate to the nearest half pixel) where the midpoint algorithm must test the value of F in order to decide which pixel to fill. Mark each point with a letter A.

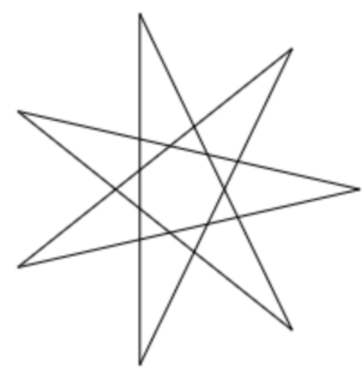
   3.3. Suppose that the difference in the *x* coordinates between p_1 and p_2 is 4.2, and the difference in the *y* coordinates is 1.8. Suppose that the value of F at the bottom left decision point is 0.78. Compute the values of F at the other decision points, as they would be computed by the incremental midpoint algorithm.

Polygons (6 points)

4. Consider the code below, which is intended to draw one of the 7-pointed stars shown.



```
function star() {
    let cx = 150;
    let cy = 150;
    let r = 100;
    graphics.strokeStyle = 'black';
    graphics.beginPath();
    graphics.moveTo(cx+r,cy);
    for (i = 1; i <= 7; i++) {
      graphics.lineTo(cx+r*Math.cos(i*theta), cy+r*Math.sin(i*theta));
    }
    graphics.stroke();
}
```



4.1. What value is needed for `theta`, in radians? (Hint: since we are skipping two points, it will be three times the value for a regular polygon with the same number of points.)

4.2. Suppose that our floodfill algorithm is called, with starting point at the center of the figure.
Draw what would be filled, using the first star on top.

4.3. Suppose that a scanline fill is called on the polygon. Draw what would be filled, using the second star on the bottom.

Transformation Matrices (10 points)

5. Give the 3x3 homogeneous transformation matrix that would achieve the following effects when applied before drawing something on the canvas.

5.1. Translate by (100,20) and then by (30,10).

5.2. Translate by (50,25) and then rotate by 90 degrees.

5.3. Rotate by 180 degrees and then translate by (40,70)

5.4. Scale by 3 along both axes and then translate by (10,10)

Clipping (6 points)

6. Suppose that we have a viewport that is 240 pixels tall by 400 pixels wide, with a standard *y* axis pointing down. We want to draw a line segment between the endpoints (-64, -25) and (416, 275). Compute the intersection points of this line segment with all four boundary lines of the viewport (top, bottom, left, and right).

   6.1. What is the intersection point with the top of the viewport?

   6.2. What is the intersection point with the bottom of the viewport?

   6.3. What is the intersection point with the left of the viewport?

   6.4. What is the intersection point with the right of the viewport?

   6.5. What is the Cohen-Sutherland 4-bit code for the first endpoint?

   6.6. What is the Cohen-Sutherland 4-bit code for the second endpoint?

____ / 54

SCRATCH PAPER

SCRATCH PAPER