Family Name: . . . . . **Other Names:** . . . . . .

ID Number: . . . . . **Signature** . . . . . .

Solution Notes
CGRA 151: Terms Test

22nd August 2017

Instructions

- Time allowed: **45 minutes** .
- Answer **all** the questions. There are four questions and 45 marks in total.
- Write your answers in the boxes in this test paper and hand in pages 1–12.
- There is brief Processing documentation on page 13, which you may detach from the test paper.
- If you think some question is unclear, ask for clarification.
- This test contributes 10% of your final grade
- You may use paper translation dictionaries and approved calculators.
- You may write notes and working on this paper, but make sure your answers are clear.

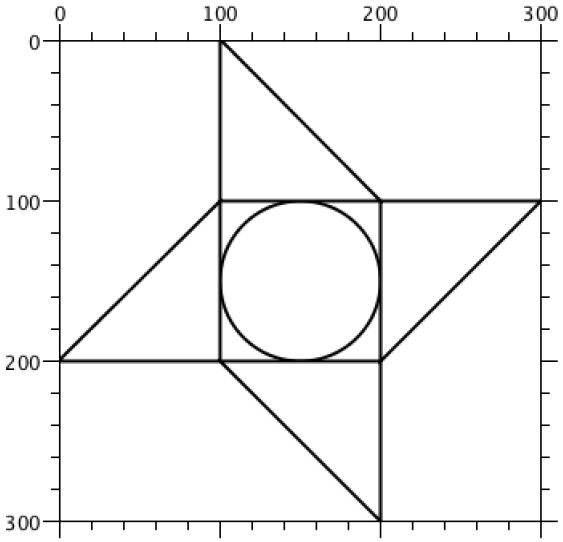
Questions	**Marks**	
1. Graphical Output in Processing	[10]	
2. Straight Line Algorithm	[10]	
3. Vectors, Matrices and Transformations	[15]	
4. Bézier Cubic Curve	[10]	
	TOTAL	

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

1. **Graphical output in Processing** **(10 marks)**

(a) **(5 marks)** Generating graphical output. Write Processing code to draw the following things in the Processing window.



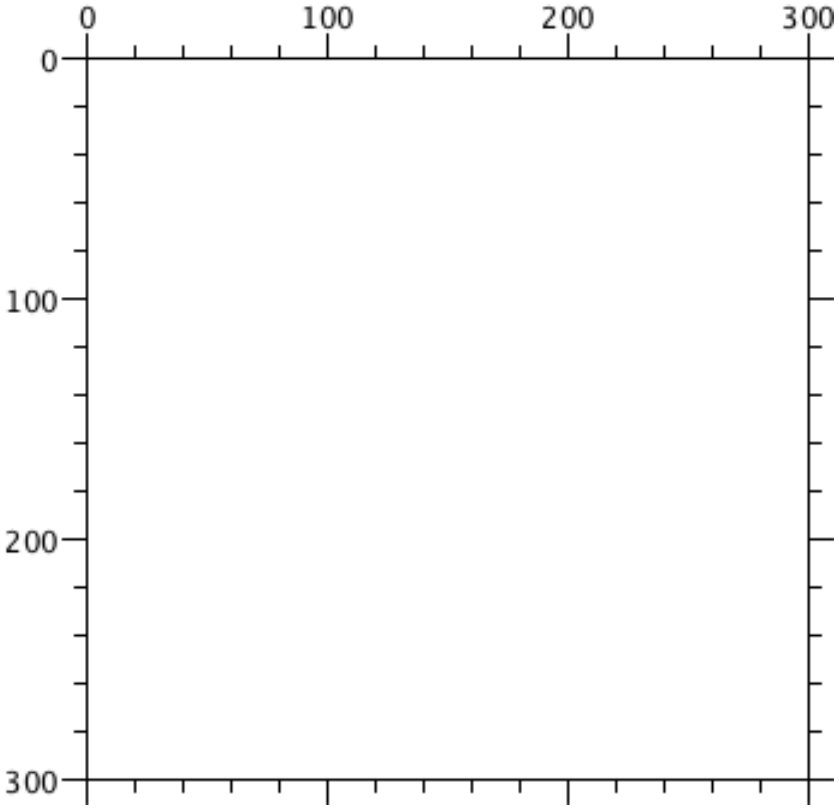
(the external grid and numbers are not part of the sketch, they are there for guidance)

```
size (300, 300);

// setup
background(255);
fill(255);
stroke(0);

// draw the shapes directly
ellipse(150,150, 100,100);
triangle(100,0, 100,100, 200,100);
triangle(100,200, 200,200, 200,300);
triangle(0,200, 100,200, 100,100);
triangle(200,100, 200,200, 300,100);

// alternatively you could use transforms
translate(150,150);
ellipse(0,0, 100,100);
for( int i=0 ; i<4 ; i++ ){
  triangle(50,50, -50,50, 50,150);
  rotate(PI/2);
}



//
```

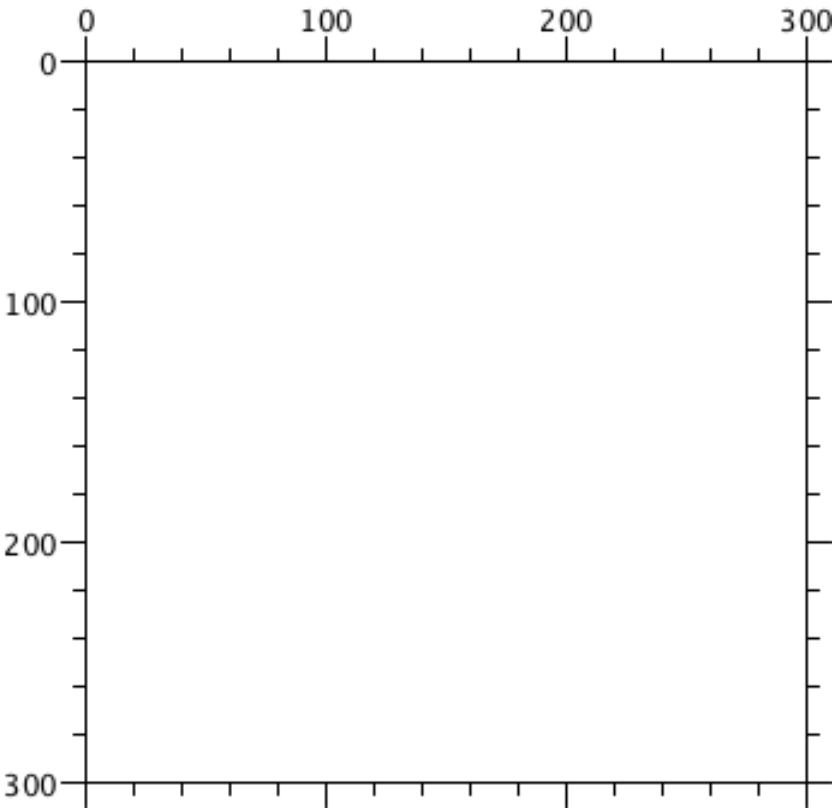
(Question 1 continued on next page)

(Question 1 continued)

Extra copies, provided in case you make a mistake in Question 1(b)



(spare copy of 300 × 300 Processing window)

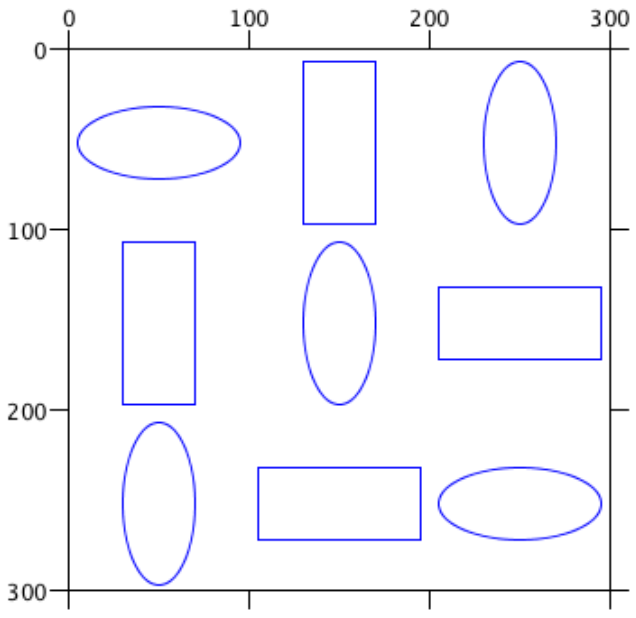


(spare copy of 300 × 300 Processing window)

(b) **(5 marks)** Transformations

Sketch what the following code will draw in the Processing window.

```
size( 300, 300 ) ;
background( 255 ) ;
noFill();
stroke(0);
int i = 0;
while (i < 3 ){
  for( int j = 0 ; j < 3 ; j++ ) {
      if (i + j == 1) {
        rectMode(CENTER);
        rect( i*100 + 50, j * 100 + 50, 40, 90 ) ;
      }
      else if( i + j == 2 ) {
        ellipse (i*100 + 50, j * 100 + 50, 40, 90);
      }
      else if (i + j == 3){
        rectMode(CENTER);
        rect( i*100 + 50, j * 100 + 50, 90, 40 ) ;
      }
      else{
        ellipse (i*100 + 50, j * 100 + 50, 90, 40);
      }
  }
  i++;
}
```



(300 × 300 Processing window)

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

2. **Straight line algorithm** **(10 marks)**

Write a method to draw a straight line by drawing the correct individual pixels.

- Assume that the slope of the line is between 45° and 90°.
- Assume that the line starts at (x_0, y_0) and ends at (x_1, y_1).
- Assume that $y_0 < y_1$.
- Assume that x_0, y_0, x_1, y_1 are all integers.
- Assume that function `drawPixel(x,y)` draws the pixel at integer location (x, y).
- You may write in Java code, Processing code, or pseudo-code. Minor syntax errors will not be penalised.

Hint: note that this line requires one pixel to be drawn in each *row* of pixels, unlike the examples in the lecture notes which require one pixel in each *column*.

```
void drawLine( int x0, int y0, int x1, int y1 ) {

      // get the inverse slope of the line
      // need 1.0 in there to prevent integer division
      float m = (1.0*(x1-x0))/(y1-y0);

      // set up variables for the integer and fractional parts of x
      // true x = x+xi
      float xi = 0.0 ;
      int x = x0 ;

      int y = y0 ;

      // draw the first pixel
      drawPixel(x,y);

      // ensure that the loop terminates after drawing the last pixel
      // not after the last-but-one or the one-after-the-last
      while (y < y1) {
            y++ ;

            // the fractional part of x must be between -0.5 and 0.5
            xi += m ;
            if( xi > 0.5 ) {
                xi -= 1.0 ;
                x += 1 ;
            }
            drawPixel(x,y)
      }


}
```

3. **Vectors, Matrix and Transformation** **(15 marks)**

(a) **(5 marks)** Vector and matrix algebra

You are given three vectors, $\mathbf{p} = \begin{bmatrix} -2 \\ 1 \end{bmatrix}$, $\mathbf{q} = \begin{bmatrix} 5 \\ 3 \end{bmatrix}$ and $\mathbf{r} = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}$

and two matrices, $\mathbf{M_1} = \begin{bmatrix} 2 & 1 & 3 \\ 0 & 2 & 3 \\ 0 & 0 & 1 \end{bmatrix}$ and $\mathbf{M_2} = \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$,

Evaluate the following expressions.

Vector addition $\mathbf{p} + \mathbf{q} =$ $\begin{bmatrix} 3 \\ 4 \end{bmatrix}$

Dot product $\mathbf{p} \cdot \mathbf{q} =$ $(-2) \times 5 + 1 \times 3 = -7$

Magnitude $|\mathbf{p} + \mathbf{q}| =$ $\sqrt{3^2 + 4^2} = \sqrt{25} = 5$

Matrix addition $\mathbf{M_1} + \mathbf{M_2} =$ $\begin{bmatrix} 2 & 0 & 4 \\ 1 & 2 & 4 \\ 0 & 0 & 2 \end{bmatrix}$

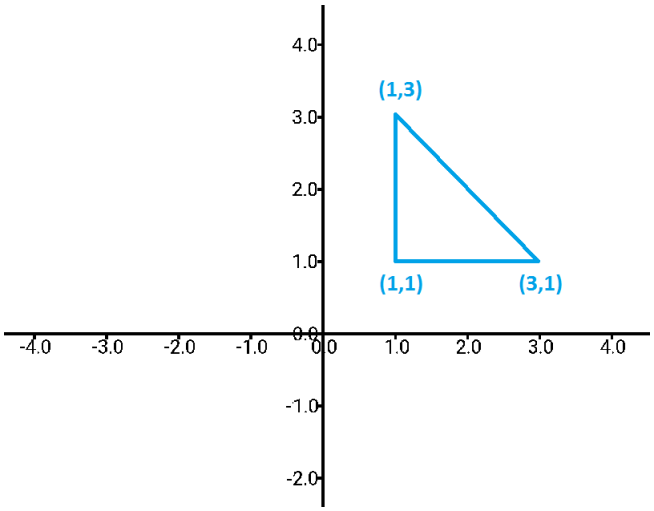
Matrix multiplication $\mathbf{M_2 r} =$ $\begin{bmatrix} -1 \\ 3 \\ 1 \end{bmatrix}$

(Question 3 continued on next page)

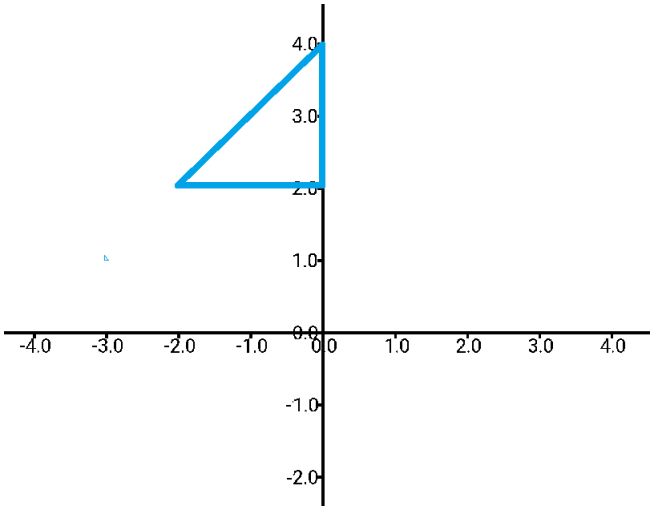
(Question 3 continued)

(b) **(10 marks)** Transformation

The diagram shows a triangle with a right angle. We now want to apply the transformation represented by M_2 in Question 3(a) to this triangle.



(i) (6 marks) Draw the transformed triangle.



(ii) (4 marks) Give a sequence of transformation commands in Processing that would produce the same result as the transformation represented by the matrix M_2.

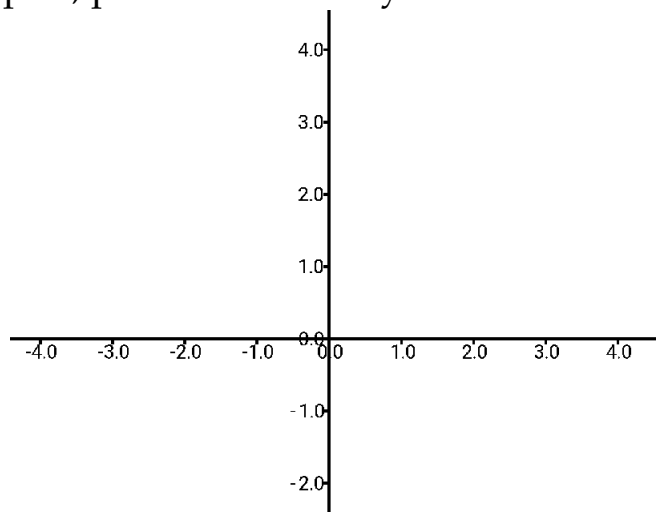
1 translate(1,1)

2 rotate(PI/2)

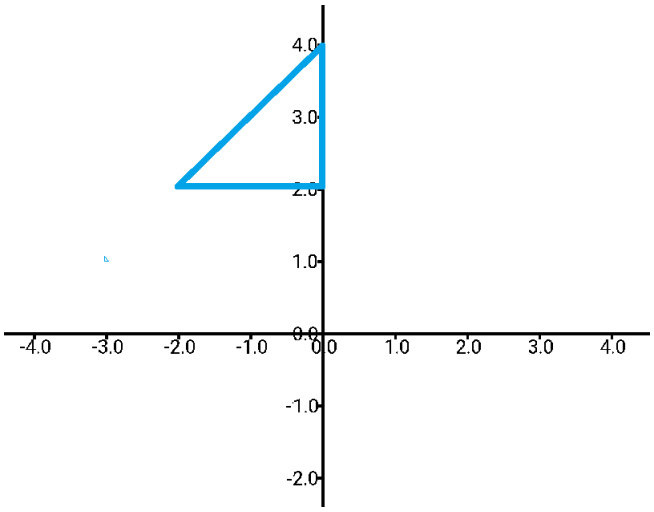
(Question 3 continued on next page)

(Question 3 continued)

Extra copies, provided in case you make a mistake



(spare copy of coordinates for 3(b)(i))



(spare copy of coordinates for 3(b)(i))

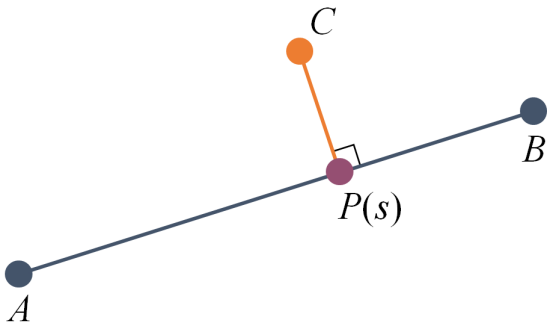
$$\text{point}\,(1,1) \rightarrow \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} \rightarrow \text{point}\,(0,2)$$

$$\text{point}\,(1,3) \rightarrow \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \\ 1 \end{bmatrix} \rightarrow \text{point}\,(-2,2)$$

$$\text{point}\,(3,1) \rightarrow \begin{bmatrix} 0 & -1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \\ 1 \end{bmatrix} \rightarrow \text{point}\,(0,4)$$

4. **Bézier Cubic Curve** **(10 marks)**

(a) **(6 marks)** Given two endpoints of a Bézier cubic curve, A and B, and one control point C, we draw a perpendicular from C to line segment AB. The interception point is $P(s)$. Given the representation that $P(s) = (1-s)A + sB$, write a formula for the value of s given the positions of the points A, B, and C as inputs.



$s = \frac{\overline{AB} \cdot \overline{AC}}{|\overline{AB}|^2}$

where:

$\overline{AB} = (x_B - x_A, y_B - y_A)$
$\overline{AC} = (x_C - x_A, y_C - y_A)$
$|\overline{AB}|^2 = (\sqrt{(x_B - x_A)^2 + (y_B - y_A)^2})^2 = (x_B - x_A)^2 + (y_B - y_A)^2$

alternatively:
$s = \frac{(x_B - x_A)(x_C - x_A) + (y_B - y_A)(y_C - y_A)}{(x_B - x_A)^2 + (y_B - y_A)^2}$

(b) **(4 marks)** Explain how this distance is used in the algorithm for drawing a Bézier cubic curve as a sequence of straight lines.

Bézier curves are drawn by checking whether you can approximate the curve by a straight line.
To do this you use the distance $|\overline{CP(s)}|$ as a flatness check.
You do this flatness check for both control points.
If both distances are within a tolerance then you draw a straight line from A to B.
Otherwise you subdivide the curve into two Bézier curves and recurse.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Brief Processing documentation

```
void point( float x, float y )
// draw a point at location (x,y)

void line( float x1, float y1, float x2, float y2)
// draw a line segment from location (x1,y1) to location (x2,y2)

void rect( float left, float top, float width, float height )
// draw a rectangle with its top left corner at the indicated position
// of size width times height

void ellipse( float centerX, float centerY, float width, float height )
// draw an ellipse centred at the indicated position
// of size width times height

void triangle( float x1, float y1, float x2, float y2, float x3, float y3 )
// draw a triangle with vertices at the three points
// (x1,y1), (x2,y2), (x3, y3)

void background( float grey )
void stroke( float grey )
void fill( float grey )
// set the colour of the background, stroke or fill to be
// a grey value between 0 (black) and 255 (white)

void noStroke()
void noFill()
// turn off drawing strokes around objects and filling objects, respectively

void rotate( float angle )
// rotate the universe by angle radians clockwise

void scale( float s )
// scale the universe by a factor of s
void scale( float sx, float sy )
// scale the universe by sx in the x-direction and sy in the y-direction

translate( float tx, float ty )
// translate the universe a distance (tx, ty)
```

****