

15-104 *Practice* Written Examination, Fall 2016

Roger B. Dannenberg, instructor

Possibly useful function signatures (*italics* mean “an expression goes here”):

<code>createCanvas(w, h);</code>	<i>create a canvas with the given dimensions</i>
<code>width</code>	<i>the width of the canvas, once it has been created</i>
<code>height</code>	<i>the height of the canvas, once it has been created</i>
<code>key</code>	<i>the last key typed</i>
<code>background(r, g, b);</code>	<i>fill/erase the canvas with an RGB color</i>
<code>background(grayLevel);</code>	<i>fill/erase the canvas with a grayscale value</i>
<code>rect(x, y, w, h);</code>	<i>draw a rectangle</i>
<code>ellipse(x, y, w, h);</code>	<i>draw an ellipse or circle</i>
<code>line(x1, y1, x2, y2);</code>	<i>draw a line</i>
<code>point(x1, y1);</code>	<i>draw a point (size will be determined by <code>strokeWeight</code>)</i>
<code>rectMode(mode);</code>	<i>mode can be CENTER, RADIUS, CORNER, or CORNERS</i>
<code>ellipseMode(mode);</code>	<i>mode can be CENTER, RADIUS, CORNER, or CORNERS</i>
<code>fill(r, g, b);</code>	<i>set the fill color for subsequent shapes to the color (r,g,b)</i>
<code>fill(grayLevel);</code>	<i>set the fill color to the specified graylevel</i>
<code>stroke(r, g, b);</code>	<i>set the color for lines or borders, to the color (r,g,b)</i>
<code>stroke(grayLevel);</code>	<i>set the color for subsequent lines or borders, to the graylevel</i>
<code>noFill();</code>	<i>choose that subsequent shapes will be drawn with no fill</i>
<code>noStroke();</code>	<i>choose that subsequent shapes will be drawn with no border</i>
<code>strokeWeight(pixels);</code>	<i>set the thickness for subsequent lines or borders</i>
<code>push();</code>	<i>save the current graphics transformation settings</i>
<code>pop();</code>	<i>restore the most-recently saved transformation settings</i>
<code>translate(x, y);</code>	<i>translate subsequent drawing by offsets x and y</i>
<code>scale(x, y);</code>	<i>scale by factors x and y (from the origin)</i>
<code>rotate(radians(degrees));</code>	<i>rotate by degrees (around the origin)</i>
<code>random(x);</code>	<i>get a random number between 0 and x</i>
<code>random(low, high);</code>	<i>get a random number between low and high values</i>
<code>min(a, b);</code>	<i>get the lesser of the two numbers, a and b</i>
<code>max(a, b);</code>	<i>get the greater of the two numbers, a and b</i>
<code>map(val, inLo, inHi, outLo, outHi);</code>	<i>linearly re-maps a number, whose current range extends from inLo to inHi, to a new ‘target’ range, which extends from outLo to outHi.</i>
<code>function setup(){...}</code>	<i>a handler for what happens when the sketch first starts</i>
<code>function draw(){...}</code>	<i>a handler for what happens when the screen is refreshed</i>
<code>function keyPressed(){...}</code>	<i>a handler for what happens when the user presses a key</i>
<code>function mousePressed(){...}</code>	<i>a handler for what happens when the user clicks the mouse</i>
<code>noLoop();</code>	<i>disables continuous updating; draw() is executed only once.</i>
<code>[a, b, ...]</code>	<i>expression to construct an array</i>

<code>a.push(x)</code>	insert value of <code>x</code> at the end of array <code>a</code>
<code>a.pop()</code>	remove and return value from end of array <code>a</code>
<code>a.unshift(x)</code>	insert value of <code>x</code> at the beginning of array <code>a</code> , shifting each existing element to the next index to make room
<code>a.shift()</code>	remove and return value from beginning of array <code>a</code> , shifting each other element to the next lower index to fill the vacancy
<code>a.length</code>	get the length of array <code>a</code>
<code>a.splice(start, count)</code>	remove <code>count</code> elements from the array, starting at <code>start</code>
<code>a.toString()</code>	convert array <code>a</code> to a (printable) string
<code>s.split(c)</code>	split string <code>s</code> at each occurrence of character <code>c</code> , returning an array of strings

operators:**and:** `&&` **or:** `||`**equality test:** `===` or `==` (*not recommended*)**inequality test:** `!==` or `!=` (*not recommended*)**comparisons:** `>` `<` `>=` `<=`

(please leave this to the graders to fill in)

Grade: _____ / 70 points

Question 0. 0 points (but please read)

As usual with p5js, the origin of the coordinate system is in the upper left and larger values of y are toward the bottom of the page.

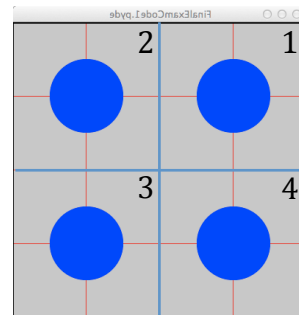
Where shown, the *grid lines* in some of the following output figures are for positional reference and to aid you in the answering the questions. The code for drawing the lines is not shown in the code.

Question 1. 4 points

The following code draws the figure to the right. Mark each line with the quadrant number in which that line draws a circle.

```
createCanvas(400, 400);
background(200); // dark gray
fill(0, 0, 255); // blue

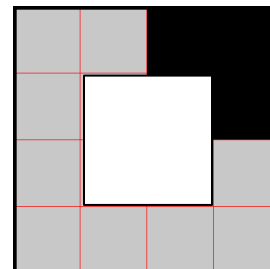
__ellipse(width*.25, height*.25, 100, 100);
__ellipse(width*.75, height*.25, 100, 100);
__ellipse(width*.25, height*.75, 100, 100);
__ellipse(width*.75, height*.75, 100, 100);
```



Question 2. 4 points

Write code in the box below to create the black and white squares in the figure to the right. Remember that the grid lines are only for reference; do not write code to create them.

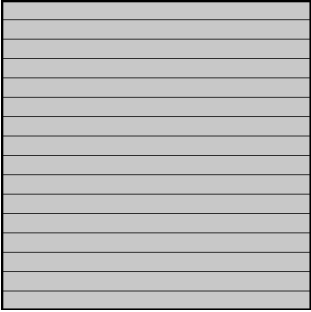
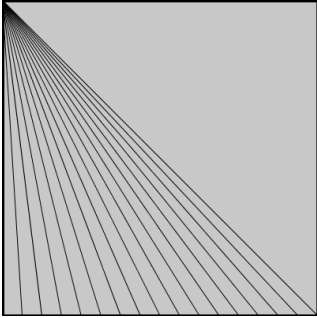
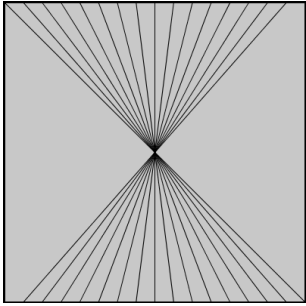
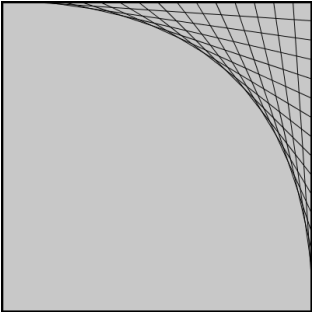
```
createCanvas(400, 400);
background(220); // gray
// add your code here:
```



Question 3. 8 points (2 point each)

Each code sample below generates one of the outputs. Label the output with the letter of the code sample that generated it.

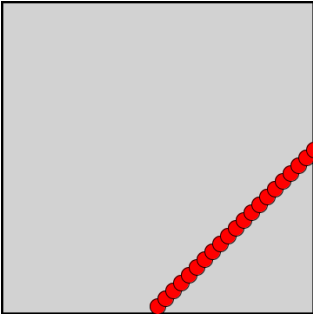
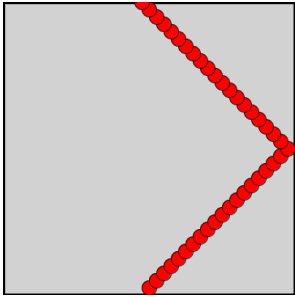
<p>Sample A: <code>createCanvas(400, 400);</code> <code>background(200);</code></p> <p><code>for (var i = 0; i < 400; i += 25) {</code> <code> line(0, i, width, i);</code> <code>}</code></p>	<p>Sample B: <code>createCanvas(400, 400);</code> <code>background(200);</code></p> <p><code>for (var i = 0; i < 400; i += 25) {</code> <code> line(i, 0, width, i);</code> <code>}</code></p>
<p>Sample C: <code>createCanvas(400, 400);</code> <code>background(200);</code></p> <p><code>for (var i = 0; i < 400; i += 25) {</code> <code> line(i, 0, width - i, height);</code> <code>}</code></p>	<p>Sample D: <code>createCanvas(400, 400);</code> <code>background(200);</code></p> <p><code>for (var i = 0; i < 400; i += 25) {</code> <code> line(0, 0, width - i, height);</code> <code>}</code></p>

 <p>Code Sample: _____</p>	 <p>Code Sample: _____</p>
 <p>Code Sample: _____</p>	 <p>Code Sample: _____</p>

Question 4. 10 points

The following code sample attempts to move the circle up and right and bounce the circle off the right edge of the window while keeping the entire circle visible in every frame. The code moves the circle but it does not bounce off the right edge. Instead, it disappears as shown in the first output below.

Add the needed code in the `draw()` function in the **ANSWER BOX** to make the circle bounce off the right edge. Your code must not let *any* portion of the circle move beyond the right edge of the window. Do not code anything that deals with the other three edges.

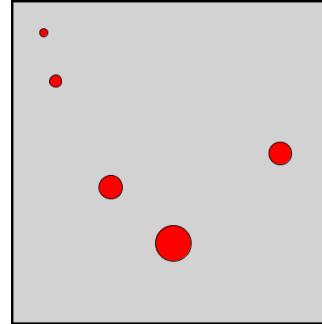
<p>Code Sample</p> <pre>var x, y, diameter, xVelocity, yVelocity; function setup() { createCanvas(400, 400); background(210); fill(255, 0, 0); // red diameter = 20; x = width/2; y = height-diameter/2; xVelocity = 10; yVelocity = -10; } function draw() { ellipse(x, y, diameter, diameter); x += xVelocity; y += yVelocity; }</pre>	<p>Actual Output</p> 
<p>ANSWER BOX</p> <pre>function draw() { ellipse(x, y, diameter, diameter); x = x + xVelocity; y = y + yVelocity; }</pre>	<p>Desired Output</p> 

Question 5. 6 points

The first code sample below draws a number of circles. Roger and Golan are tired of typing four arguments for every call of `ellipse` when, for perfect circles, the last two arguments are the same. Define a function named `circle` so the second code sample will compile and draw the same output.

```
function setup() {
  size(400, 400);
  background(210); // gray
}
function draw() {
  fill(100); // darker gray
  ellipse(40, 40, 10, 10);
  ellipse(55, 100, 15, 15);
  ellipse(200, 300, 44, 44);
  ellipse(332, 189, 28, 28);
  ellipse(123, 231, 29, 29);
}
```

Desired Output:



```
function setup() {
  createCanvas(400, 400);
  background(210); // gray
}

function draw() {
  fill(100); // darker gray
  circle(40, 40, 10);
  circle(55, 100, 15);
  circle(200, 300, 44);
  circle(332, 189, 28);
  circle(123, 231, 29);
}

// Define your circle function here:
```

Question 6. 4 points

Here is how one might represent a Queue in JavaScript:

Arrays as Queues:

```
// an empty queue:  
var queue = [];  
// a queue with 3 elements, 0 at the front, 2 at the back:  
queue = [0, 1, 2];
```

A - (1 point) The variable `qData` is a queue with the following values:

front-> 0 , 1 , 2 , 3 , 4 <- back

Thus, if we print `qData`, we will see:

[0, 1, 2, 3, 4]

Write a JavaScript statement to *enqueue* the value 7 to `qData` (add 7 at the back of the queue):

Answer : _____

B - (1 point) After this statement to enqueue 7, if we execute `print(qData)`, what will we see?

Answer : _____

C - (1 point) Write a JavaScript statement to *dequeue* an element from `qData` after enqueueing 7 in part A (remove the value at the front of the queue):

Answer: _____

After this statement to dequeue an element, if we execute `print(qData)`, what will we see?

Answer: _____

Question 7. 4 points

Here is how one might represent a Stack in JavaScript:

Arrays as Stacks:

```
// an empty stack:  
stack = []  
// a stack with 3 elements, 0 at the bottom, 2 at the top:  
queue = [0, 1, 2];
```

A - (2 points) The variable `sData` is a **Stack** object. It contains the following values:

bottom-> 0 , 1 , 2 , 3 , 4 <-**top**

Thus, if we print `qData`, we will see:

[0, 1, 2, 3, 4]

What values will `sData` contain after the following code is executed?

```
num1 = sData.pop();
```

```
num2 = sData.pop();
```

```
sData.push(8);
```

```
sData.push(13);
```

```
sData.push(0);
```

answer: _____

B - (2 points) What values will `sData` contain if the above code is executed on the original `sData` in this new order?

```
sData.push(8);
```

```
sData.push(13);
```

```
sData.push(0);
```

```
num1 = sData.pop();
```

```
num2 = sData.pop();
```

answer: _____

Question 8. 14 points

Use this declaration and initialization of an array named **elevator** in the following question:

```
elevator = [ {name: "Yokozuna", weight: 505},  
              {name: "Big Show", weight: 441},  
              {name: "Akebono", weight: 514},  
              {name: "Andre the Giant", weight: 520} ];
```

Elvis and four of the largest ever wrestlers get onto an elevator that can hold 2000 pounds. Your task is to write code that computes the total weight of everyone on the elevator as described by the array of objects in **elevator**. After your code completes, the total weight should be stored in the variable named **total**. For full credit, your code should not depend on the names or the number of objects in **elevator**.

Question 9. Global and Local Variables – 4 points**A. What does the following program print? (2 points)**

```
var x = "foo";

function setup() {
  fn();
  print(x);
}

function fn() {
  var x = "bar";
}
```

Answer: _____

B. What does the following program print? (2 points)

```
var x = "foo";

function setup() {
  fn();
  print(x);
}

function fn() {
  x = "bar";
}
```

Answer: _____

Question 10. Boolean Expressions – 4 points

What does the following program print?

Hint:

```
print(true + " " + false);
```

prints the string:
true false

```
a = true;
b = false;
print((a || a) + " " + (b || b) + " " + (a || b));
print((a && a) + " " + (b && b) + " " + (a && b));
```

Answer: _____

Question 11. String Functions - 8 points

A. What does this code print? (2 points)?

```
x = "do you know this?".split(" ")
print(x[1] + " " + x[0])
```

Answer: _____

B. What does

```
s = "upbeat";
print(s.substring(2) + " " + s.substring(0, 2));
```

print? (2 points)

answer: _____

C. Write an expression for the length of the string in variable **aString** (2 points)

answer: _____

D. Assume **fred** == "Margaret" and **jane** == "Morrison". Write an assignment statement using **fred** and **jane** that assigns the string "Margaret Morrison" to the variable **both**. Be careful to include the space between the two words. (2 points)

answer: _____

Question 12. Style - 8 points

Consider the following code. Find 4 stylistic problems for full credit, and more for extra credit:

```
1   var items = [10, 12, 15, 17, 21]
2
3   function setup() {
4     for (var i =0; i<10; i++) { // for (i=0;i<
5       if (items[i]==15) {
6         println ("found 20 at "+ i);
7       }
8     }
9
10
11
12 }
```

answers: _____
