CMPT-166: Sample Final Exam Answer Key

Last name *exactly as it appears on your student card*										**First name** *exactly as it appears on your student card*
Student Number										
SFU Email			**Section** if you know it!							

This is a **3 hour** test. It is **closed book**: no calculators, computers, notes, books, etc. are allowed.

Instructor: Toby Donaldson

Short Answer

(10 marks) Write your answer for each question in the corresponding box. Unanswered questions are worth 0.

Question	Answer
(1 mark) What is the type of 3.24?	float (or double)
(1 mark) What is the type of "3.24"?	String
(1 mark) What does this print? `println(2 + 3 * 4);`	14
(1 mark) What does this print? `int a;` `int b;` `a = 4;` `b = 5;` `a -= b;` `b = a – b;` `a += b;` `println(a + b);`	-13
(1 mark) What does this print? `int x = 3;` `print("(\"" +  x + "\")");`	("3")
(1 mark) What does this print? `print(17 % 5);`	2
(1 mark) True or false: in Processing, a **global variable** is any variable defined outside a function or class.	TRUE
(1 mark) True or false: in Processing, you **cannot** define a class inside another class.	FALSE
(1 mark) Around which point does the `rotate` function rotate?	the origin: (0, 0)
(1 mark) What are the names of the two functions used to save and restore the coordinate system?	pushMatrix() and popMatrix()

Boolean Expressions

(10 marks) Write "true" or "false" in the box for each question. Every correct answer is worth **1 mark**, every incorrect answer is worth **-0.5 marks** (guessing is not recommended!), and unanswered questions are worth **0 marks**.

Answer	Question
false	`!(!(2 == 1))`
false	`!(1 >= 0 || false)`
false	`!true`
false	`(8 == 0 && 8 != 0)`
true	`(true) || (!true && !true)`
true	`(5 != 5 || true) && (!(7 == 8))`
true	`(true || 1 != 4) && (!false || false)`
true	`((!(2 <= 3)) || (!false)) || (!false)`
false	`(6 <= 5 || 6 > 1) && (false && false)`
false	`!(0 == 9 || !false)`

Writing Functions

a) (5 marks) Write a function called `pointInCircle(a, b, x, y, diam)` that returns `true` just when the point (`a`, `b`) is in the circle whose center is (`x`, `y`) and whose diameter is `diam`. If (`a`, `b`) is not in the circle, then `false` is returned. All the input parameters are of type `float`.

Solution

```
boolean pointInCircle(float a, float b, float x, float y, float diam) {
  return dist(a, b, x, y) <= diam / 2;
}
```

b) (5 marks) If (r, g, b) is an RGB color triple, then the **inverse color** of (r, g, b) is (255 – r, 255 – g, 255 – b). Write a function called `invertColor(r, g, b)` that returns the inverse color of (`r`, `g`, `b`) as a Processing `color` object. The `r`, `g`, and `b` input parameters are each of type `int`. Also assume that `r`, `g`, and `b` are a valid RGB color.

Solution

```
color inverseColor(int r, int g, int b) {
  return color(255-r, 255-g, 255-b);
}
```

Loops

a) (2 marks) Write a while-loop that never stops (i.e. an infinite loop) and prints "hello" every time its body is executed.

```
while (true) {// any always true condition can be used
 println("hello");
}
```

b) (2 marks) Write a fragment of code that uses a while-loop to print "hello" exactly 25 times.

```
int i = 0;
while (i < 25) {
   println("hello");
   ++i;
}
```

c) (3 marks) Write a fragment of code that uses a while-loop to print the numbers from 1 to 150 on the screen (one number per line). The first number printed should be 1, and the last number printed should be 150.

```
     int i = 1;
while (i <= 150) {
   println(i);
   ++i;
}
```

d) (3 marks) Write a fragment of code that uses a while-loop to print the numbers from 150 **down** to 1 on the screen (one number per line). The first number printed should be 150, and the last number printed should be 1.

```
     int i = 150;
while (i >= 1) {
   println(i);
   --i;
}
```

Classes and Objects

1. (5 marks) Write a complete class called `Particle` that has the following:
   - It stores the (x, y) position of the particle as a `float` values.
   - It stores the (dx, dy) velocity of the particle as `float` values.
   - It has a *constructor* that takes the position and velocity of the particle as input parameters, e.g.

   ```
   Particle p;
   p = new Particle(34, -2.2, 1, 1.5);
   ```

   Solution:

   ```
   class Particle {
      float x, y;
      float dx, dy;
      Particle(float init_x, float init_y,
               float init_dx, float init_dy) {
         x = init_x;
         y = init_y;
         dx = init_dx;
         dy = init_dy;
      }
   }
   ```

2. (2 marks) Write the code that creates a variable called `origin` of type `Particle`. Initialize it to be the point (0, 0), and with 0 velocity (in both directions).

   Solution:

   ```
   Particle origin = new Particle(0, 0, 0, 0);
   ```

3. (1 mark) Write a fragment of code that creates an `ArrayList` variable called `dust` that can hold `Particle` objects. After your code runs `dust` should refer to an empty `ArrayList`.

   Solution
   ```
   ArrayList<Particle> dust = new ArrayList<Particle>();
   ```

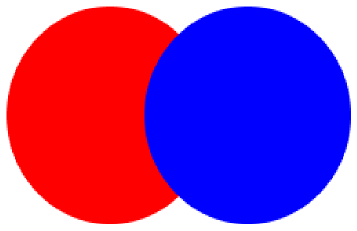
4. (5 marks) Suppose that `dust` is an empty `ArrayList` that can hold `Particle` objects (as in the previous question). Write a fragment of code that uses a while-loop to assign the particles the positions (0, 0), (1, 1), (2, 2), (3, 3), ..., (99, 99). Give `dx` and `dy` for each particle a random value in the range -1.5 to 1.5.

   Solution

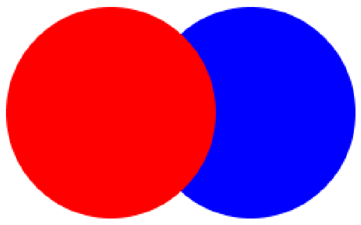
```
int i = 0;
while (i < 100) {
   Particle p = new Particle(i, i,
                             random(-1.5, 1.5),
                             random(-1.5, 1.5)
                            );
   dust.add(p);
   ++i;
}
```

Overlapping Circles

(10 marks) Write a complete program that draws two circles that are initially overlapping like this:



When you click the mouse (anywhere), the red circle is now on top, and the blue circle is on bottom:



Every time you click the mouse, the circle on the bottom is now drawn on the top.

Note that the size and location of the circles stays the same throughout the program. All that changes is the order in which they are drawn.

```
boolean blueOnTop;

void setup() {
  size(500, 500);
  blueOnTop = true;
}

// ... more on next page ....
```

```
void draw() {
  background(255);

  if (blueOnTop) {
    fill(255, 0, 0);
    ellipse(200, 250, 150, 150);
    fill(0, 0, 255);
    ellipse(300, 250, 150, 150);
  } else {
    fill(0, 0, 255);
    ellipse(300, 250, 150, 150);
    fill(255, 0, 0);
    ellipse(200, 250, 150, 150);
  } // if
}

void mousePressed() {
  blueOnTop = !blueOnTop;
}
```