Name: _____

CS 141: Introduction to (Java) Programming: Exam 1

Jenny Orr • Willamette University • Fall 2017

1.	(max 18)	5.	(max 12)
2.	(max 9)	6.	(max 16)
3.	(max 16)	7.	(max 20)
4.	(max 9)		
Total:			(max 100)

1. (2 pts each, 18 pts total) Given the variable declarations:

```
char c1 = '3';
char c2 = '3';
int x = 3;
int y = 5;
int z = 6;
String s1 = new String("3");
String s2 = new String("3");
```

Indicate whether the following Boolean expressions evaluate to true or false by circling the correct answer or indicating that the expression has an error

a) `(c1 == c2)`	true	false	error
b) `(s1 == s2)`	true	false	error
c) `(s1.equals(s2))`	true	false	error
d) `( !(x == y) && (y !=z ) )`	true	false	error
e) `( (c1 != '4') || (c1 != '3') )`	true	false	error
f) `( y/2.0 == 2.5)`	true	false	error
g) `( x < y < z )`	true	false	error
h) `( (x % 4) == 1 )`	true	false	error
i) `( x && y )`	true	false	error

2. (3 pts each, 9 pts total) Suppose x and y are 2 integers. Write a Boolean expression that tests whether or not:

a) Both of them are zero: (x == 0) && (y == 0)

b) At least one of them is zero: (x == 0) || (y == 0)

c) Exactly one of them is zero: ((x != 0) && (y == 0)) || ((x == 0) && (y != 0))

3. (1 pts each, 16 pts total) **True and False:** Please circle T or F
(Credit is only given if the instructor can clearly tell which answer is circled)

a) **T or F:** A relational operator compares two boolean values.

b) **T or F:** In general, member variables should be public.

c) **T or F:** The name of a constructor must be the name of the class.

d) **T or F:** A constructor should always have a void return value.

e) **T or F:** Object parameters are passed by value.

f) **T or F:** An object's getter method is called when the keyword `new` is used.

g) **T or F:** The scope of a *member* variable is the entire class.

h) **T or F:** The declaration below creates a Player object.

```
Player p;
```

i) **T or F:** Variables declared inside a block of code have scope which extends beyond the block of code.

j) **T or F:** Addition has a higher precedence than multiplication.

k) **T or F:** It is never ok for two variables within a program to have the same name.

l) **T or F:** If you want to store an integer value in a variable of type double, you must cast it

m) **T or F:** A number that appears in the code without explanation is called a Magic Number.

n) **T or F:** A char is a primitive type and a String is a class type.

o) **T or F:** Java will garbage collect an object if the object has no references pointing to it.

p) **T or F:** In an assignment statement, the value of the expression on the left of the "=" sign is copied into the variable on the right.

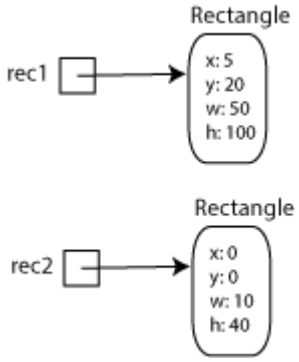
4. (9 pts) Write a multi-way if-else statement that prints “cold” when the temperature is 20 or below, “comfortable” when the temperature is between 20 and 80, and “hot” when the temperature is 80 or above.
   Assume the temperature value is stored in a double called `temp`.

```
if (temp <= 20) {
   System.out.println(“cold”) ;
} else if ( temp < 80 ) {
    System.out.println(“comfortable”) ;
} else {
    System.out.println(“hot”) ;
}
```

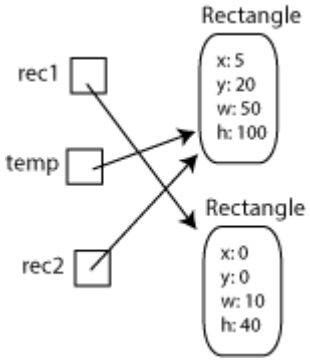
5. (4 pts each, 12 pts total) Recall the Rectangle class. The constructor creates a new Rectangle(x,y,w,h) where (x,y) is the location, w is the width and h is the height. Consider the following code below.

```
Line 1: Rectangle rec1 = new Rectangle(5, 20, 50, 100); // width = 50
Line 2: Rectangle rec2 = new Rectangle(0, 0, 10, 40);   // width = 10

Line 3: Rectangle temp = rec1;
Line 4: rec1 = rec2;
Line 5: rec2 = temp;

Line 6: System.out.println("rec1: " + rec1.getWidth());
Line 7: System.out.println("rec2: " + rec2.getWidth());
Line 8: System.out.println("temp: " + temp.getWidth());
```

a) Draw the memory (references and objects) immediately after Lines 1-2 have been executed.



b) Draw the memory (references and objects) after Lines 1-5 have been executed.



c) What is the output at lines 6-8?

rec1: 10
rec2 : 50
temp: 50

6. (16 pts total) Scope: For each variable listed at the top:
   a. On the row below the variable name, indicate whether the variable is a
      - parameter (**P**)
      - local variable (**L**), or
      - member variable (**M**).
   b. Place a **D** on the line where the variable is declared.
   c. Mark an "**x**" to indicate the variable's scope.

Note: The variable `args` is already completed as an example.

		args	b	eggs	time	x	tot	a	b	x
	Enter P, L or M →	**P**	**L**	**M**	**M**	**L**	**L**	**P**	**L**	**L**
1	`public class HungryProject{`									
2	`    public static void main(String[] args) {`	**D**								
3	**`        Breakfast b = new Breakfast();`**	**x**	**D**							
4	`    }`	**x**	**x**							
5	`}`									
6	`public class Breakfast`			**x**	**x**					
7	`{`			**x**	**x**					
8	`    private int eggs = 12;`			**D**	**x**					
9	`    private double time = 3.5;`			**x**	**D**					
10	`    public Breakfast() {`			**x**	**x**					
11	`        int x = 2;`			**x**	**x**	**D**				
12	`        double tot = time * calc(x);`			**x**	**x**	**x**	**D**			
13	`        System.out.println("cook "+tot+" min.");`			**x**	**x**	**x**	**x**			
14	`    }`			**x**	**x**	**x**	**x**			
15	`    public double calc(int a) {`			**x**	**x**			**D**		
16	`        System.out.println("a="+a);`			**x**	**x**			**x**		
17	`        int b = 1;`			**x**	**x**			**x**	**D**	
18	`        if (eggs > 6) {`			**x**	**x**			**x**	**x**	
19	`            int x = 3;`			**x**	**x**			**x**	**x**	**D**
20	`            b = a*x;`			**x**	**x**			**x**	**x**	**x**
21	`        }`			**x**	**x**			**x**	**x**	**x**
22	`        return b;`			**x**	**x**			**x**	**x**	
23	`    }`			**x**	**x**			**x**	**x**	
24	`}`			**x**	**x**					

7. (20 pts) Create a `City` class consisting of the following:
   - Two member variables: one for the city's name (e.g. Salem) and its population (e.g. 167419) (*make sure you use the appropriate data type such as* `String, char, int, double`, etc).
   - A constructor which has 2 parameters used for setting the value of each member variable.
   - A setter & getter for the name variable.
   - A `toString` method.

```
public class City {

    // Member variables:

      private String name;
      private int population;


    // Constructor:

     public City(String n, int p) {
       name = n;
       population = p;
     }


   // Getter for name:

      public String getName() {
       return name;
     }


   // Setter for name:

     public void setName(String n) {
       name = n;
     }


   // toString:


   public String toString() {
    Return “City “ + name + “ has population “ + population;
   }



}
```