# Midterm 1 SOLUTIONS

Total: 1000 points

CSCI 110 Section 1

Monday, September 12, 2016

1) Determine the result of evaluating each expression. [200 points]

```
28 / 5
```
5

```
29 % 5
```
4

```
true && !(false || true)
```
false

```
"5" == 5 || true
```
error

```
(4 * 5) % 3 > 2 ||
    (35 - 15) % 3 == 0
```
false

```
"Wonderful" > "great"
```
error

```
int a = 13;
int b = 4;
a % b > 1 || a / b > 3
```
false

```
boolean a = false;
boolean b = true;
!(b || !b) || !(a || !a)
```
false this is false no matter what the values of a and b are

```
float c = 4.7;
int d = 5;
boolean e = d < c;
String f = "hello";
(f.length() * 3) % d > c ||
    (e && (d + c) < 10)
```
false https://repl.it/D7Vq

```
String g = "damn";
String h = "daniel";
boolean i = g.length() > 4;
boolean j = h.length() > 5;
i = true;
boolean k = !i || !j;
(i && j) || (j && k)
```
true (i && j) is true https://repl.it/D7Vn

2) How many times will the body of the following while loop be executed? [50 points]

```
int i = 12;
while (i < 35) {
    i++;
    System.out.println("foo");
}
```

23 times: once for i = 12 through i = 34

3) What will be printed when the following function is called? [50 points]

```
int a = 0;
System.out.println("ready 2 roll");
int n = 0;
while (n < 6) {
    n += 2;
    a += n + ",";
}
System.out.println(a);
```

Trick question; there's an error on the line `a += n + ",";`

We can get away with adding an integer and a String (gory technical details). The integer automatically gets converted to a String. However, the problem comes when you try to assign this String back to a, which is of type integer. Remember, writing

```
a += n + ",";
```

is equivalent to writing

```
a = a + n + ",";
```

4) Briefly explain what each line of code does. [100 points]

```
01  String makeEmail(String first, String last, String domain) {
02      String result = last + "." + first + "@" + domain + ".com";
03      return result;
    }
04  System.out.println(makeEmail("jay", "leno", "cars"));
05  System.out.println(makeEmail("leslie", "knope", "parks"));
06  System.out.println(makeEmail("sandor", "clegane", "pacifism"));
07  System.out.println("that's all folks");
```

01 starts the definition for a function makeEmail that returns a String and has three String parameters

02 assembles the parameters into an email address and puts that in a String

03 returns the assembled email address

04 prints "leno.jay@cars.com"

05 prints "knope.leslie@parks.com"

06 prints "clegane.sandor@pacifism.com"

07 prints "that's all folks"

5) Please convert the following while loop to a for loop. [100 points]

```
float h = 6.0f;
boolean j = false;
float m = 243.54f;
String k = "bar";
int l = 900;
while (l != 6080) {
    k += "baz";
    h -= 2.0f;
    m = h * h + m;
    j = !j;
    l += 1;
}
```

Don't overthink this one.

```
float h = 6.0f;
boolean j = false;
float m = 243.54f;
String k = "bar";
for (int l = 900; l != 6080; l += 1) {
    k += "baz";
    h -= 2.0f;
    m = h * h + m;
    j = !j;
}
```

Say you have the following function:

```
String mystery(String a, float b, boolean c) {
    String result = "";
    while (c) {
        while (result.length() < 12) {
            result += a;
            result += Boolean.toString(c);
        }
        return result;
    }
    if (b > 10) {
        float y = b;
        while (y < 4 * b) {
            result += "foobar";
            y += a.length();
        }
    }
    return result;
}
```

6) What's the result of calling mystery("hello", 11.0f, false)? You can show your work by tracing the values of variables. [100 points]

Okay, there's a lot happening here. First, mystery("hello", 11.0f, false) is a function call. Execution drops into the first line of "mystery" and a gets assigned the value "hello", b gets assigned the value 11.0f, and c gets assigned the value false.

Next, we assign the empty string to the variable result.

Since c is false, we skip past the while (c) loop to the next statement, which is if (b > 10)... We know b is 11.0f so the if statement's condition is true.

We copy the value of b into a new variable named y. We check if y < 4 * b. It's true that 11 is less than 44, so we start looping. Each iteration appends "foobar" to the variable result. We know a.length() == 5 since a is "hello", so 5 gets added to y on every loop iteration.

4 * b is 44, so the loop runs 7 times, for y = 11, 16, 21, 26, 31, 36, 41. This means "foobar" gets appended to result seven times, so **"foobarfoobarfoobarfoobarfoobarfoobarfoobar"** is returned.

7) What's the result of calling mystery("goodbye", 25.0f, true)? [100 points]

mystery("hello", 11.0f, false) is a function call. Execution drops into the first line of "mystery" and a gets assigned the value "goodbye", b gets assigned the value 25.0f, and c gets assigned the value true. Next, we assign the empty string to the variable result.

Since c is true, we enter the while(c) loop. We encounter another while loop. The condition is result.length() < 12. This is true because result is "" which has length 0.

In this inner while loop we add "goodbye" and "true" to result on every loop iteration. After one iteration, result becomes "goodbyetrue". We go back up to check the condition. result.length() has length 11, so we iterate once more. Now result is "goodbyetruegoodbyetrue". We go up to check the condition. This time it's false. So we go to the next statement after this inner while loop.

The next statement is "return result;". We stop what we're doing in the function, and execution goes back to the site of the function call.

Result's final value is **"goodbyetruegoodbyetrue"**, so that is what gets returned the caller.

8) Assume there are already defined integers x and y that you don't know the values of. If x is positive and y is negative, or if x is negative and y is positive, print "whoa". Otherwise, print "chill". [100 points]

```
if ((x > 0 && y < 0) || (x < 0 && y > 0)) {
    System.out.println("whoa");
} else {
    System.out.println("chill");
}
```

9) Say there are two functions already defined: boolean isFuzzy(int num) takes an integer and returns true if the number is fuzzy and false if not. boolean isMagical(int num) takes an integer and returns true if the number is magical and false if not. Write a loop to count how many numbers are fuzzy but NOT magical between 867 and 10942 (inclusive). [100 points]

Think of this as "accumulating" the count of fuzzy and non-magical numbers.

```
int acc = 0;
for (int i = 867; i <= 10942; i++) {
    if (isFuzzy(i) && !isMagical(i)) {
        acc++;
    }
}
```

10) Assume there's an integer named `num` that you don't know the value of. Write code to round `num` to the nearest multiple of 10. For numbers in the middle (i.e. ending in 5) round up. Hint: how do you get the last digit of a number? [100 points]

We get the last digit by computing num modulo 10. Then we round up if the last digit is at least 5. Otherwise, we round down.

```
int lastDigit = num % 10;
if (lastDigit >= 5) {
    num = num - lastDigit + 10;
} else {
    num = num - lastDigit;
}
```

11) Say you have two ranges [a1, a2] and [b1, b2] where a1, a2, b1, and b2 are pre-defined integers. Write code to calculate a boolean value: true if the two ranges overlap, false if not. Example: Given two ranges [0, 3] and [2, 4], there is an overlap from 2 to 3 so the result would be true. Hint: draw a number line to help visualize this problem. [extra credit, 100 points]

This is the solution I came up with:

```
bool doOverlap = false;
if (a1 >= b1 && a1 <= b2) {
    doOverlap = true;
}
if (a2 >= b1 && a2 <= b2) {
    doOverlap = true;
}
```

I searched around and there's a shorter way to do this. Because you know that a1 <= a2 and b1 <= b2, the expression a1 <= b2 && b1 <= a2 is sufficient.
http://stackoverflow.com/questions/3269434/whats-the-most-efficient-way-to-test-two-integer-ranges-for-overlap