CS113 Midterm 1

Answers are in Green

Question **1 (12 POINTS).**

Basic Java Knowledge. Write Java command(s) to do the following:

(1) Define a floating point variable named, **x**

```
double x;
```

(2) Set the variable **x** (from above) to 42.3

```
x=42.3;
```

(3) Increment the value in **x** by 5.9

```
x = x + 5.9;    or x += 5.9;
```

(4) Compute the $\mathbf{x^{1.61}}$ — that is x raised to the power of the golden mean — and place it in a new variable, **y**

```
double y = Math.pow(x, 1.61);
```

(5) Convert the value of **x** to an integer and save the result in an new integer variable **z**:

```
int z = (int)x;
```

(6) Convert **theta** (type **double**) from degrees to radians

```
double theta = 12.0;    // not required by question, I chose 12.0 arbitrarily
theta = Math.toRadians(theta);
```

Question 2: **(7 POINTS)** Unix Matching

The table below consists of three vertical sections. The leftmost section, labelled “match” is where your answers will go. The center section has Unix commands. The right section has possible descriptors for the Unix commands along with single letter identifiers. In the match column, put letters corresponding to any descriptors that accurately describe the Unix command. If there is no good descriptor, leave the cell in the match column blank. Similarly, if you think more than one descriptor is correct for a Unix command, then put multiple letters in the match column. Some descriptors may not be used. Some may be used more than once.

Match	UNIX Command		Description
	ls	**A**	copy a file from one machine to another
G	mkdir	**B**	open a terminal that is connected to another machine
J,D	mv	**C**	Open a new terminal window
B	ssh	**D**	rename a file
H	cat	**E**	move a file from one machine to another
F	cp	**F**	copy a file
I	javac	**G**	make a new directory
		H	display the contents of a text file
		I	compile a Java program
		J	change the location of a file

Question 3 (**21 Points**, total over 3 parts): Understanding and Modifying Code
For all questions in this section, you do not need to write an entire Java program; you only need to write the code as described.

Question 3.1 (7 points)
The following code uses a for-loop to compute the smallest power of 2 that is greater than some given number m. This code will compile successfully and print the correct answer (assuming you define and instantiate m)

```
1       for (int i=0; m > 0; i++) {
2           m = m / 2;
3           if (m==0) {
4              System.out.println(i);
5           }
6       }
```

Rewrite this code so that it has the same functionality (including printing the answer) but uses a while-loop instead of a for-loop. Your program may NOT use any if statements.

```
int m=10; // does not appear above, but for completeness
int i=0;
while (m>0) {
      m=m/2;
      I++;
}
System.out.println(i);
```

Question 3.2 (7 points)
The following code is attempting to compute the Nth value in a series. :

```
1         int base = 10;
2         base = base + base * 2 + 2;
3         base = base + base * 3 + 4;
4         base = base + base * 4 + 6;
5         base = base + base * 5 + 8;
6         System.out.println(base);
```

The code is correct but could be written more succinctly using a loop.

Rewrite the code above so that it has the same functionality; that is computes the Nth value of the same series. But use a loop. You may use either a for-loop or a while-loop.

```
int base = 0;
int count = 4; // do not really need this
for (int i=0; i<count; i++) {
    base = base + base*(i+2) + (i+1)*2;
}
System.out.println(base);
```

Question 3.3 (7 points)

The author of the following code tried to write a program that prints powers of 2 between the variables **start** and **end**, inclusive:

```
1       for (int i = start; i < end; i += 2) {
2           int k = 1;
3           while (k < i) {
4               k = k * 2;
5           }
6           if (k == i) {
7               System.out.println(k);
8           }
9       }
```

This code correctly prints "8 16" when **start** = 6 and **end** = 25. However:

- The code prints "8 16" when **start** = 6 and **end** = 32, even though it should print "8 16 32", i.e. it's missing the "32" at the end
- The code doesn't print anything at all when **start** = 3 and **end** = 9, even though it should print "4 8"

There are *two* separate and distinct mistakes in this implementation. In the space below, describe each of the mistakes *and* how they can be fixed. Be as specific as possible.

Describe the mistake, i.e. the code that is incorrect	**How can it be fixed?**
missing 32: loop ends too early	change the termination condition on the for loop to i<=end

No 4 or 8:
The loop is incrementing on odd numbers rather than even. So the if statement will never match
	There are several fixes: here are two:
	Before line 1 add the following code:
	if (start%2!=0) {
	start = start + 1;
	}
	
	OR
	in line 1 change i+=2 to i++

Question 4 (20 Points)

Write a complete Java program named **Calculator** that takes three program inputs (runtime arguments) as follows:

- The first is an int that indicates which operation to perform:
  - 1 means compare,
  - 2 means multiply,
  - 3 means raise the first number to the power of the second number
- The next two inputs are doubles that represent the operands, i.e. the values to compare, multiply or powerify (which is definitely not a word).

The program should then perform the specified operation on the two operands and then print the result. For instance:

```
java Calculator 1 4.5 5.6
false
java Calculator 2 5.1 5.0
25.5
java Calculator 3 6.25 0.5
2.5
```

Note that your output may be slightly different from mine. For instance, rather than 2.5 in the third example you might get 2.4999999999.

The program should display an error message and exit without performing any calculations if fewer than three program inputs are specified:

```
java Calculator 2 5.6
Wrong number of inputs
```

Assuming the three inputs exist, you can assume that the first is an int and the other two are doubles, but the program should display an error if the first input (the operation) is not 1 or 2 or 3:

```
java Calculator 4 3.1 9.3
Invalid operation
```

```
public class Calculator {
    public static void main(String[] args) {
        int operator = Integer.parseInt(args[0]);
        if (operator < 1 || operator > 3) {
            System.out.println("Invalid Operation");
            return;
        }
        // my mistake, we did not get to this so this is
extra credit (+2)
        if (args.length != 3) {
            System.out.println("Wrong number of inputs");
            return;
        }
        double n1 = Double.parseDouble(args[1]);
        double n2 = Double.parseDouble(args[2]);

        if (operator == 1) {
            System.out.println(n1 == n2);
        }
        if (operator == 2) {
            System.out.println(n1 * n2);
        }
        if (operator == 3) {
            System.out.println(Math.pow(n1, n2));
        }
    }
}
```

Question 5 (10 points) Given three integer variables, **x**, **y**, **z** (assume they are already defined) write Java **commands** to assign to a variable **min** (you have to define it) the smallest value in **x**, **y**, and **z**.

```
int x = 5;
int y=7;
int z=2;

int min=0;

if (x<y) {
    if (x<z) {
       min=x;
   } else {
       min=z;
   }
} else { // y<x
       if (y<z) {
       min=y;
   } else {
       min=z;
   }
}
```

Question 6 (15 points) Here is an algorithm for computing a function of two numbers **a** and **b**:

```
while (a ≠ b)
    if a > b
         a←a−b
     else
         b←b-a
```

("←" is the assignment operator)

Assuming **a**, and **b** are already defined (as integers), write Java **commands** to code the above algorithm.

```
int a=20;

int b = 12;

while (a!=b) {
    if (a>b) {
        a=a-b;
   } else {
        b = b-a;
   }
}
```

Question 7 (12 points)

Answer the questions below about the following code:

```
1         int n = 5;
2         for (int i = 0; i <= n; i++) {
3             for (int j = n; j > i; j = j - 1) {
4                 if ((i + j) >=7) {
5                     System.out.println("big");
6                 } else if ((i+j) > 4) {
7                     System.out.println("medium");
8                 }
9             }
10         }
```

A. How many times will “big” be printed?

4

B. How many times will “medium” be printed?

5

C. How many times will the boolean expression on line 4 be evaluated?

15

D. How many times will the boolean expression on line 6 be evaluated?

11

Question 8: (1 point)
True or false: George Boole is the father of Donald Knuth.

FALSE: George died long before Don was born.

Question 9 (20 Points)

Write a **complete Java program** called QuestionNine that simulates the tossing of two six-sided dice and computes the estimated probability of obtaining a sum of 9. Your program should take one integer input from the command line; the number of trials. The output should be the estimated probability of rolling 9. Here is a sample output:

```
java QuestionNine 9999
The probability of obtaining a 9 in 9999 trials is 0.11155
```

(note that if you actually test your program, the computed probability should be pretty close to 4/36 (or 0.11111111111111))

```
public class QuestionNine {
    public static void main(String[] args) {
        int trials = Integer.parseInt(args[0]);
        int nine = 0;
        for (int i=0; i< trials; i++) {
            int d1 = (int)(Math.random()*6 + 1);
            int d2 = (int)(Math.random()*6 + 1);
            if ((d1+d2)==9) {
                nine++;
            }
        }
        System.out.println("Probability of 9 in " + trials +
"trials is " + (nine*1.0/trials));
    }

}
```

(page intentionally left blank)