CS 312 – Midterm 2 – Fall 2013

Your Name____________________________________

Your UTEID ____________________________________

Circle your TAs Name: EDAENA LEIF MUBASHIR

Problem Number	Topic	Points Possible	Points Off
1	code trace	28	
2	arrays	14	
3	strings	16	
4	program logic	16	
5	scanner	23	
6	arrays and strings	15	
7	simulation	30	
		TOTAL POINTS OFF:	
		SCORE OUT OF 142:	

Instructions:

1. Please turn off your cell phones
2. You have 2 hours to complete the test.
3. You may not use a calculator.
4. Please make your answers legible.
5. When code is required, write Java code.
6. You may break problems up into smaller methods. (In other words you can add helper methods.)
7. Style is not evaluated when grading.
8. The proctors will not answer questions. If you believe there is an error or a question is ambiguous, state your assumptions and answer based on those assumptions.
9. When you finish, show the proctor your UTID, turn in the exam and all scratch paper.

1. Evaluating Code. 28 points, 2 points each. Answer each question below. If the snippet contains a syntax error answer **syntax error**. If the snippet results in a runtime error or exception answer **runtime error**. If the code results in an infinite loop answer **infinite loop**. Assume all necessary imports have been made.

A. What is output by the following code?

```
int xa = 3;
int ya = 2;
boolean pa = xa > ya;
boolean qa = xa % ya == 1;
System.out.print( (pa && qa) + " " + (pa || qa));
```

Answer: _____________________________

B. Are the two `boolean` expressions below equivalent? In other words do the two expressions always evaluate to the same value given the `boolean` variables `p` and `q` store the same values?

Expression 1: `!(p && q)`
Expression 2: `!p && !q`

Answer: _____________________________

C. What is output by the following code?

```
String strc = "PORTER";
String subc = strc.substring(2, 5);
System.out.print(strc + " " + subc);
```

OUTPUT: _____________________________

D. What is output by the following code?

```
String strd = "KLIVANS";
String subd = strd.substring(3);
System.out.print(subd);
```

Answer: _____________________________

E. What is output by the following code?

```
int xe = 3;
int ye = 2;
int[] datae = new int[xe * ye];
xe += 2;
System.out.print(datae.length);
```

Answer: _____________________________

F. What is output by the following code?

```
int[] dataf = new int[10];
dataf[1] += dataf[3] * dataf[6];
System.out.print(dataf[1]);

Answer: __________________________
```

G. What is output by the following code?

```
int[] data1g = {1, 2, 3, 4};
int[] data2g = {1, 2, 3, 3};
data2g[3] = 4;
System.out.print(data1g == data2g);

Answer: __________________________
```

H. What is output by the following code?

```
int[] data1h = {2, 4, -5};
int[] data2h = {1, 2, 1};
data2h = data1h;
data1h[1] += 2;
data2h[2] -= 2;
System.out.println(Arrays.toString(data2h));

Answer: __________________________
```

I. What is output by the following code?

```
int x = 5;
int[] datai = new int[x / 2 - 4];
System.out.print(datai.length);

Answer: __________________________
```

J. What is output by the following code?

```
int[] dataj = {-5, 2, 1, 2};
dataj[2] += dataj[1];
dataj[3] -= dataj[2];
System.out.print(Arrays.toString(dataj));

Answer: __________________________
```

K. What is output by the following code?

```
int[] datak = {0, 2, 4};
datak[1] += datak[datak[1]];
datak[2] -= datak.length;
System.out.print(Arrays.toString(datak));

Answer: __________________________
```

L. What is output by the following code?

```
int[][] mat = new int[5][7];
System.out.print(mat.length + " " + mat[0].length);

Answer: __________________________
```

M. What is output by the following code?

```
int[] datam = {-5, 2, 7};
methodM(datam);
System.out.print(Arrays.toString(datam));

public static void methodM(int[] values) {
      values[2] += 3;
      values[1] = 12;
      values = new int[4];
      values[1] = 5;
}

Answer: __________________________
```

N. What is output by the following code?

```
int[] datan = new int[10];
datan.length += 10;
int xn = 5;
System.out.print(datan[xn - 10]);

Answer: __________________________
```

2. Arrays. 14 points. Write a method `sumArrays`. The method has two parameters. Each parameter is an array of `int`s. You may assume neither of the array variables equals `null`. The method returns an array equal in length to the smaller of the two parameters. The elements of the returned array equal the sum of the corresponding elements in the two parameters.

Neither of the parameters is altered as a result of this method.

Examples of calls to `sumArray`:
`sumArrays({1, 2}, {2, 3})` returns the array `{3, 5}`
`sumArrays({}, {2, 3})` returns the array `{}`
`sumArrays({1, 2, 2, 4}, {4, 6})` returns the array `{5, 8}`
`sumArrays({}, {})` returns the array `{}`

You may not use any other methods or classes in your answer other than Java's built in, native arrays.
You may not use the static methods from the Arrays class.

```
public static int[] sumArrays(int[] a1, int[] a2) {
```

3. Strings 16 points. (Based on a question from codingbat.com)Write a method `repeatedReversedEnd`. The method has 3 parameters, a `String str`, an `int n`, and an `int num`. The method creates and returns a new `String` that consists of the last `n` characters of `str` in reverse order, repeated `num` times.

Examples of calls to `repeatedReversedEnd(String str, int n, int num):`

```
repeatedReversedEnd("ABCD", 2, 3) returns "DCDCDC"
repeatedReversedEnd("ABCD", 0, 3) returns ""
repeatedReversedEnd("ABCD", 3, 0) returns ""
repeatedReversedEnd("ABCD", 4, 1) returns "DCBA"
```

The `String` in the next example contains a single space

```
repeatedReversedEnd("A *D", 4, 3) returns "D* AD* AD* A"
repeatedReversedEnd("AAABB", 4, 3) returns "BBAABBAABBAA"
```

Assume `0 <= n <= str.length(), num >= 0,` and `str != null`

The only methods you may use from the String class are the `length`, `charAt`, and `substring` methods plus `String` concatenation. You may not use any other Java classes or methods in your answer.

```
public static String repeatedReversedEnd(String str, int n, int num) {
```

4. Program Logic 16 Points. Consider the following method. For each of the four points labeled by comments and each of the four assertions in the table, write whether the assertion is ***always*** true, ***sometimes*** true, or ***never*** true at that point in the code. Abbreviate ***always*** with an A, ***sometimes*** with an S and ***never*** with an N.

```
public static int assertionPractice(int a, int b) {
    int result = -1;
    // POINT A
    if(a > 0 && b > 0) {
        while(a != b) {
            // POINT B
            if(a > b) {
                a -= b;
            }
            else {
                b -= a;
            }
        }
        result = a;
        // POINT C
    }
    // POINT D
    return result;
}
```

Abbreviate ***always*** with an A, ***sometimes*** with an S and ***never*** with an N.

	`result == b`	`b > a`	`result == -1`	`a == b`
`POINT A`				
`POINT B`				
`POINT C`				
`POINT D`				

5. Scanner 23 Points. Write a method `tokensPerLine`. The method has one parameter, a `Scanner` object. The `Scanner` is already connected to a file. The file contains at least one line. The method returns the average number of tokens (based on whitespace) per line.

For example, if the `Scanner` object was connected to the following file,

```
10    cat    20
30

50   20cat20
cat
dog  32
```

then the method would return `1.5 (9 / 6).`

The given file contains 9 tokens (10, cat, 20, 30, 50, 20cat20, cat, dog, and 32) and has 6 lines. There is one blank line with no tokens between the line that contains 30 and the line that contains 50 20cat20.

You may only use `Scanner` objects and the constructors and methods from the `Scanner` class. You may not use any other Java classes or methods in your answer. Write the complete method, including the header, below:

6. Arrays and Strings 15 Points. Write a method `getNGrams`. The method takes 2 parameters, a `String str` and an `int n`. The method returns an array of Strings that are the *n-grams* of the `String str`.
An n-gram *is a contiguous sequence of n characters from a given String*.

For example if n = 2 and the String is "dogs" the n grams (in this case *bigrams*) are "do", "og", and "gs".

If n = 3 and the String is "computer" the n grams are (in this case *trigrams*) are "com", "omp", "mpu", "put", "ute", and "ter".

If n = 4 and the String is "***" there are no *quadgrams* and the method shall return an empty array.

You may `String`s including the `length`, `substring` and `charAt` methods and Java's built in, native arrays. You may not use any other methods or classes.

You may assume `str != null` and `n > 0`.

```
public static String[] getNGrams(String str, int n) {
```

7. Simulation 30 Points. Write a method `rollUntilSame`. The method has one parameter, `int` num.

The method rolls `num` 6-sided dice per round. The method prints out the results of each round.

The method repeats rounds of rolling the `num` 6-sided dice until a round occurs where all `num` dice have the same value. All dice are rerolled every round.

Here is one sample run with `num` dice equal to 5. Your method's output must match the format below.

```
4 4 5 6 6
4 5 4 2 5
1 6 1 5 6
1 3 6 6 3
3 3 3 6 6
4 5 6 5 6
5 1 6 3 2
4 4 4 4 4
Number of rounds before all dice the same: 8
```

Recall how to create an object of type `Random`:

```
Random r = new Random();
```

and the method from the `Random` class, `nextInt(n)` that returns an `int` from 0 to n - 1.

You may not use any other Java classes or methods except Java's built in arrays and the `Random` class. You may not use the static methods from the Arrays class.

Assume the parameter `num` is greater than or equal to 2.

```
public static void rollUntilSame(int num) {
```

Complete this method on the next page.

```
// assume num >= 2
public static void rollUntilSame(int num) {
```