CSCI 135 Exam #0 **Name: _____**
Fundamentals of Computer Science I
Fall 2014

This exam consists of 7 problems on the following 7 pages.

You may use your single-side hand-written 8 ½ x 11 note sheet during the exam. No calculators, computers, or communication devices of any kind are permitted.

If you have a question, raise your hand and I will stop by. Since partial credit is possible, **please write legibly and show your work**.

Problem	Points	Score
1	10	
2	12	
3	10	
4	11	
5	8	
6	6	
7	6	
Total	63	

1. Conditionals (10 points). The following program that takes a single integer as input from the command line. Recall that the static method `Math.abs()` computes the absolute value of a number (drops any negative sign).

```
public class Prob1
{
    public static void main(String [] args)
    {
        int a = Integer.parseInt(args[0]);

        if (a > 100)
            System.out.print("big ");

        if (a < 0)
            System.out.print("yuck");
        else
            System.out.print("yum");

        int d = Math.abs(a - 10);
        if (d < 2)
            System.out.println("!");
        else if (d < 4)
            System.out.println(".");
        else
            System.out.println("?");
    }
}
```

Give the output of `Prob1` for each of the following commands. ***If the program would crash, write "error"***:

Command	**Program output**
% java Prob1 200	big yum?
% java Prob1 -200	yuck?
% java Prob1	error
% java Prob1 11	yum!
% java Prob1 14 fee fi fo fum	yum?

2. Debugging, arrays, boolean expressions (12 points). The following program reads a single integer value *N* from the command line. *N* specifies the size of an array that the program is supposed to create and fill with values. Unfortunately the program has a bug:

```
 1 public class Debugging
 2 {
 3     public static void main(String [] args)
 4     {
 5         int N = Integer.parseInt(args[0]);
 6         int [] a;
 7         for (int i = 0; i < a.length; i++)
 8             a[i] = i * i;
 9         System.out.println("Bye bye!");
10     }
11 }
```

A. The program fails to compile with a "`variable a might not have been initialized`" compile-time error. Identify the line number containing the bug and give a corrected version of this line of code.

Line number __6__

Correct version: `int [] a = new int[N];`

B. Assume the bug has been fixed. Java is currently on line 9 of the program. Circle the names of variables (including arrays) that are in scope (i.e. you could use them in some way in the print statement and Java would know the variable exists). There may be ***0 or more*** correct answers:

I. **args**
II. **N**
III. length
IV. i
V. **a**
VI. String

C. Assume the bug have been fixed. The program has been executed via the following command:

```
% java Debugging 5
```

Assume Java is currently on line 9 of the program. Circle the correct value of each of the following `boolean` expressions:

boolean expression	**value**
`a[0] == 0`	**true** false
`a[2] < 4`	true **false**
`a[0] != 0`	true **false**
`a.length == N`	**true** false
`(a[0] < a[1]) && (a[3] < a[2])`	true **false**
`(a[0] < a[1]) || (a[3] < a[2])`	**true** false
`9999.9 * 10 < Double.POSITIVE_INFINITY`	**true** false

3. Java expressions (10 points). Give the type and value of each of the following expressions. If an expression causes a compile or runtime error, write "error" in the type column (and leave the value blank). If an expression involves randomness, provide a single particular value that could be generated by the expression.

Expression	Type	Value
`3 * 4`	int	12
`1 + 2 / 3`	int	1
`Double.parseDouble("23.1")`	double	23.1
`Integer.parseInt("16.1")`	error	
`Integer.parseInt(16 / 3)`	error	
`5.0 + 3 * 2`	double	11.0
`Math.random() - 1.0`	double	-0.267655 or some other number in [-1.0, 0.0)
`(int) Math.random()`	int	0
`23 % 2`	int	1
`"BB" + 4 + "N"`	String	"BB4N"

4. Loops (11 points). You are building a program OddUpEvenDown that reads a single integer value *N* from the ***command line***. The program prints out all the odd numbers from 1 up to *N* on the first line of output. On the second line of output, it prints all the even numbers from *N* down to 1. Here are some example runs:

% java OddUpEvenDown 7 1 3 5 7 6 4 2	**% java OddUpEvenDown 1** 1
% java OddUpEvenDown 10 1 3 5 7 9 10 8 6 4 2	**% java OddUpEvenDown 0**

Here is a partially completed program. Put a single letter in each box such that the program works correctly. Each letter may be used ***0 or more times*** (i.e. a letter may appear in one box, multiple boxes, or in no boxes).

```
public class OddUpEvenDown
{
    public static void main( [E] args)
    {
        int N = [G] ;

        [C] ( [K] ; [T] ; [N] )
        {
            System.out.print(i + " ");
        }

        System.out.println();

        if ( [X] )
            N = N - 1;

        [C] ( [L] ; [Q] ; [M] )
        {
            System.out.print(i + " ");
        }

        System.out.println();
    }
}
```

A. `do`
B. `while`
C. `for`
D. `int []`
E. `String []`
F. `int`
G. `Integer.parseInt(args[0])`
H. `Integer.parseInt(args[1])`
I. `StdIn.readInt()`
J. `int i = 0`
K. `int i = 1`
L. `int i = N`
M. `i = i - 2`
N. `i = i + 2`
O. `i--`
P. `i++`
Q. `i >= 1`
R. `i >= 0`
S. `i < N`
T. `i <= N`
U. `System.out.println()`
V. `System.out.print()`
W. `N % 2 == 2`
X. `N % 2 == 1`
Y. `N % 2 == 0`
Z. `N / 2`

5. Static methods (8 points). Consider the following program which makes use of the static method `mystery`:

```
public class MysteryPrint
{
    public static String mystery(String s, int numRepeat)
    {
        String result = "hi";
        for (int i = 0; i < numRepeat; i++)
            result = result + s;
        s = "oh oh";
        return result;
    }

    public static void main(String[] args)
    {
        System.out.println("Line 1: " + mystery("triskaidekaphobia", 0));

        String s = "bye";
        System.out.println("Line 2: " + mystery(s, 2));
        System.out.println("Line 3: " + s);

        System.out.println("Line 4: " + mystery(mystery("yum", 1), 1));
    }
}
```

The above program outputs four lines of text, each starting with the prefix "Line X: ". Fill in the text occurring after the prefix on each line:

```
Line 1: hi



Line 2: hibyebye



Line 3: bye



Line 4: hihiyum
```

6. Arrays (6 points). What values does the following code put in the array `a[]`?

```
int N = 6;
int[] a = new int[N];
a[0] = 1;
a[1] = 2;
for (int i = 2; i < N; i++)
   a[i] = a[i-2] * a[i-1];
```

Complete the following table showing the array elements and values. The first row has been done for you:

Array element	Value
a[0]	1
a[1]	2
a[2]	2
a[3]	4
a[4]	8
a[5]	32

7. Drawing (6 points). The following program draws using StdDraw. The program uses the default coordinate system with (0.0, 0.0) being the lower-left corner and (1.0, 1.0) being the upper-right corner. In the first half, the program draws some grid lines which are already shown. The second half draws some additional shapes. Draw these shapes in their correct location and size.

```
public class Drawing
{
    public static void main(String[] args)
    {
        // Draw vertical grid lines in red
        StdDraw.setPenColor(StdDraw.RED);
        for (double x = 0.0; x <= 1.0; x = x + 0.2)
            StdDraw.line(x, 0.0, x, 1.0);

        // Draw horizontal grid lines in blue
        StdDraw.setPenColor(StdDraw.BLUE);
        for (double y = 0.0; y <= 1.0; y = y + 0.2)
            StdDraw.line(0.0, y, 1.0, y);

        // Now draw some mystery shapes in black!
        StdDraw.setPenColor(StdDraw.BLACK);
        double radius = 0.1;
        StdDraw.filledCircle(0.8, 0.8, radius);
        StdDraw.filledCircle(0.2, 0.8, radius);
        StdDraw.filledCircle(0.5, 0.5, radius);

        double halfWidth  = 0.4;
        double halfHeight = 0.1;
        StdDraw.filledRectangle(0.5, 0.2, halfWidth, halfHeight);
    }
}
```