CSCI 135 Exam #0 **Name: _____**
Fundamentals of Computer Science I
Fall 2013

This exam consists of 7 problems on the following 6 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	7	
2	14	
3	9	
4	3	
5	5	
6	10	
7	9	
Total	57	

1. Conditionals and loops (7 points). Consider the following code fragment:

```
int n = 1;
for (int i = 0; i < 5; i++)
{
   System.out.println(i + " " + n);
   if (n < 8)
      n = n * 2;
}
```

a) Give the output produced by the code.

```
0 1
1 2
2 4
3 8
4 8
```

b) Add code into the boxes below to convert the above program to use a `while` loop instead of a `for` loop.

```
public static void main(String [] args)
{
    int n = 1;
    int i = 0;
    while (i < 5)
    {
        System.out.println(i + " " + n);
        if (n < 8)
            n = n * 2;
        i++;
    }
}
```

2. Java expressions (14 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 any value that could conceivably be generated by the expression.

Expression	Type	Value
`10 + 20 - 20.0`	double	10.0
`2.5 / 1.0`	double	2.5
`3 / 3`	int	1
`3 / 4`	int	0
`3 % 4`	int	3
`Math.random()`	double	e.g. 0.33812 anything in [0.0, 1.0)
`(int) (Math.random() * 5)`	int	e.g. 1 anything in [0, 4]
`"bob" + 123`	String	"bob123"
`1 + 2 + 3 + "bob"`	String	"6bob"
`Integer.parseInt(7)`	error	
`Integer.parseInt("7")`	int	7
`Integer.parseInt("7.1")`	error	
`(1 <= (5 - 4))`	boolean	true
`(1 > 2) || (1 > 3) || (1 > 0)`	boolean	true

3. Conditionals and loops (9 points). You are writing a program that can find the sum, minimum or maximum of a list of floating-point numbers. The program gets its data from ***two or more command-line arguments***. The first argument is an integer specifying the mode: **0 = sum, 1 = minimum, 2 = maximum**. Following the mode integer, there are one or more floating-point data points. Here are a number of sample runs:

```
% java Calc 0 1.0 2.5
Points: 2
Result: 3.5

% java Calc 1 1.0 2.5 -2.7
Points: 3
Result: -2.7

% java Calc 2 1.0 2.5 -2.7
Points: 3
Result: 2.5

% java Calc 0 16.2
Points: 1
Result: 16.2

% java Calc 1 16.2
Points: 1
Result: 16.2

% java Calc 2 16.2
Points: 1
Result: 16.2
```

Below is the skeleton of the program. In the empty boxes, write the letter of the code fragments that combine to create a correct implementation. ***Not all letters will be used and each letter can only be used once.***

```
public class Calc
{
   public static void main(String [] args)
   {
      int    mode  = [ E ]
      double value = [ H ]

      for (int i = 2; i < [ C ]; i++)
      {
         double d = [ J ];
         if [ L ]
            value += d;
         else if [ N ]
            value = [ P ];
         else
            value = [ O ];
      }
      System.out.println("Points: " + [ B ]);
      System.out.println("Result: " + value);
   }
}
```

A. (args.length + 1)	**B.** (args.length - 1)
C. args.length	**D.** Integer.*parseInt*(args)
E. Integer.*parseInt*(args[0])	**F.** Double.*parseDouble*(args[0])
G. Integer.*parseInt*(args[1])	**H.** Double.*parseDouble*(args[1])
I. Integer.*parseInt*(args[i])	**J.** Double.*parseDouble*(args[i])
K. (mode.equals("0"))	**L.** (mode == 0)
M. (mode.equals("1"))	**N.** (mode == 1)
O. Math.*max*(d, value)	**P.** Math.*min*(d, value)
Q. Math.*max*(d, Double.*POSITIVE_INFINITY*)	**R.** Math.*min*(d, Double.*POSITIVE_INFINITY*)

4. Static methods (3 points). Consider the following program:

```
public class Cube
{
   public static void cube(int i)
   {
      i = i * i * i;
   }

   public static void main(String[] args)
   {
      for (int i = 0; i < 1000; i++)
         cube(i);
   }
}
```

How many times does the `for` loop iterate in the `main` method?

1000

5. Static methods (5 points). What does the following program output?

```
public class MethodJumping
{
   public static void printWorld()
   {
      System.out.print("mundo");
   }
   public static int addNums(int num1, int num2)
   {
      return num1 + num2;
   }
   public static void main(String [] args)
   {
      System.out.print("Hola ");
      printWorld();
      System.out.print(", 1 + 2 = ");
      int a = addNums(20, 30);
      System.out.println(a);
   }
}
```

"Hola mundo, 1 + 2 = 50"

6. Standard input (10 points). All code fragments below read data from standard input. In each box, write the letter corresponding to the best description of what each program does. ***Letters may be used 0 or more times.***

```
int N = StdIn.readInt();
String [] a = new String[N];
for (int i = 1; i <= N; i++)
   a[i - 1] = StdIn.readString();
for (int i = N - 1; i >= 0; i--)
   System.out.print(a[i] + " ");
```

A

```
int N = StdIn.readInt();
int [] a = new int[N];
for (int i = 0; i < N; i++)
   a[i] = StdIn.readInt();
double b = 0.0;
for (int i = 0; i < N; i++)
   b += a[i];
System.out.println(b);
```

K

```
int N = StdIn.readInt();
int [] a = new int[N];
for (int i = 0; i < N; i++)
   a[i] = StdIn.readInt();
double b = 0.0;
for (int i = 0; i < N; i++)
   b += a[i];
System.out.println(b / a.length);
```

J

```
int N = StdIn.readInt();
String [] a = new String[N];
for (int i = 0; i < N; i++)
   a[i] = StdIn.readString();
for (int i = 0; i < N; i++)
   System.out.print(a[N - 1 - i] + " ");
```

A

```
int N = StdIn.readInt();
int    [] a = new int[N];
String [] b = new String[N];
double [] c = new double[N];
double d = 0.0;
for (int i = 0; i < N; i++)
{
   a[i] = StdIn.readInt();
   b[i] = StdIn.readString();
   c[i] = StdIn.readDouble();
   d += a[i] * c[i];
}
System.out.println(d);
```

G

A. Reads in N strings into an array. Prints out the words in reverse order.

B. Reads in N - 1 strings into an array. Prints out the words in reverse order.

C. Reads in N strings into an array. Prints out every other word.

D. Reads in N strings into an array. Sorts the array in alphabetical order.

E. Reads in N strings into an array. Converts the strings to integers and prints the number of integers that were equal to the length of the array.

F. Reads in N web site products into parallel arrays. Each product is specified by a quantity, a description and a price (in that order). Prints out a sequence of lines where each line describes one product in the order.

G. Reads in N web site products into parallel arrays. Each product is specified by a quantity, a description and a price (in that order). Prints out the order total.

H. Reads in N numbers into parallel arrays. Outputs the minimum, maximum and sum of each array.

I. Reads in N integers into an array. Outputs the count of the number unique integers in the array.

J. Reads in N integers into an array. Outputs the average of the numbers in the array.

K. Reads in N integers into an array. Outputs the sum of all the numbers in the array.

7. Debugging (9 points). You are working on program to print a square grid of alternating symbols. The user specifies three command-line arguments: an integer square size and two symbols. Each symbol is specified by a string which you can assume is always a single character. Here is the expected output for some examples:

```
% java Square 4 a B
aBaB
BaBa
aBaB
BaBa
```

```
% java Square 1 a B
a
```

```
% java Square 3 . #
.#.
#.#
.#.
```

Here is your current program which has three bugs:

```
 1 int    N    = args[0];
 2 String sym1 = args[1];
 3 String sym2 = args[2];
 4
 5 for (int i = 0; i < N; i++)
 6 {
 7    for (int j = 0; j < N; i++)
 8    {
 9       if ((i + j) % 2 == 0)
10          System.out.print(sym1);
11       else
12          System.out.print(sym2);
13       System.out.println();
14    }
15 }
```

A. Which bug prevents the program from *compiling* successfully? Identify the line number with the mistake that causes the compile error. Give a correct version of this line of code.

Line number __1__

Correct version:

```
int N = Integer.parseInt(args[0]);
```

B. After fixing the compile error, your program now runs but gets stuck in an *infinite loop*. Identify the line causing the infinite loop. Give a correct version of this line of code.

Line number __7__

Correct version:

```
for (int j = 0; j < N; j++)
```

C. After fixing the infinite loop bug, your program now terminates. It outputs the correct number of symbols and they alternate, but each appears on its own line. What do you need to do to fix this final bug?

Move line 13 to after curly brace of for-j loop.