COS 126 **General Computer Science** **Spring 2012**

Written Exam 1

This test has 9 questions, weighted as indicated in the table below. The exam is closed book, except that you are allowed to use a one-page cheatsheet. No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided.

Write out and sign the Honor Code pledge before turning in the test:
"I pledge my honor that I have not violated the Honor Code during this examination."

_____
Signature

Problem	Score
0	/1
1	/7
2	/8
3	/5
4	/8
Sub 1	

Problem	Score
5	/9
6	/8
7	/8
8	/6
Sub 2	

Total	

Name:

NetID:

Precept: (circle your precept)

P01	12:30	Christopher Moretti
P01A	12:30	Donna Gabai
P01B	12:30	Jude Nelson
P02	1:30	Chris Miller
P02A	1:30	Donna Gabai
P02B	1:30	Maia Ginsburg
P02C	1:30	Thiago Pereira
P03	2:30	Chris Miller
P03A	2:30	Christopher Moretti
P04	3:30	Maia Ginsburg
P04A	3:30	Dom Kao
P05	7:30	Adi Dror
P06	10:00	Deep Ghosh
P07	1:30	Ian Davey
P07A	1:30	Jude Nelson
P07B	1:30	Dmitry Drutskoy
P08	12:30	Deep Ghosh
P08A	12:30	Mark Browning

0. **Honor Code** Write out the honor code on the cover (now!) and sign it when you complete the test.

1. **Expressions** Give the type and value of each of the following Java expressions. If an expression will not compile, write `Illegal` under type and put an X in the value column. You must fill in every entry. Entries left blank will be marked incorrect.

Expression	Type	Value
`"100" + 99`		
`3 + (int) Math.random()`		
`100 + 99 + "A"`		
`!(!false && true) || false`		
`(double)(3 / 2)`		
`3 / 2.0 + 2 / 5`		
`1 <= 2 && 2e11 <= 11e2`		

2. **Number Systems** You are a TOY machine. That means you store numbers as 16 bit 2's Complement binary and display them as 4-digit hexadecimal.

   Fill out the missing numbers in the following table. The first is done for you as an example.

Decimal	Binary in Memory	Hex on display
0	0000 0000 0000 0000	0000
7		
-7		
	0000 0000 1010 0010	
		FEED

3. **Methods and Input/Output** Consider the following program:

```
 1 public class MethodMadness {
 2     private static int printX( int y ) {
 3         int x = 2;
 4         StdOut.println( y );
 5         return 0;
 6     }
 7
 8     private static int printY( int x ) {
 9         int y = -1;
10         StdOut.println( x );
11         return x;
12     }
13
14     public static void main(String[] args) {
15         int x = StdIn.readInt() + 1;
16         int y = printY( printX( x ) );
17         printX(y);
18     }
19 }
```

(A.) What is printed when we run this command:
`% java MethodMadness 10`
and type in `1` on standard input? Please circle your final answer.

(B.) What is printed when the we run the series of commands:
`% java MethodMadness 10 | java MethodMadness 20`
and type in `3` on standard input? Please circle your final answer.

4. **Loops and Arrays** Consider the following code.

```
public class IntegerSort {
    public static void main(String[] args) {
        int MAX = 10;                // integers are between 0 and MAX-1
        int[] freq = new int[MAX];   // freq[i] = number of occurrences of i

        while (!StdIn.isEmpty()) {
            int i = StdIn.readInt();
            freq[i]++;
        }

        for (int i = 0; i < MAX; i++) {
            for (int j = 0; j < freq[i]; j++) {
                StdOut.print(i + " ");
            }
        }
        StdOut.println();
    }
}
```

Suppose the input to this program is:

```
9 2 3 1 0 2 2
```

(A.) Fill in the trace table with the non-zero values of the `freq` array. Each line of the trace represents one iteration of the while loop above.

0	1	2	3	4	5	6	7	8	9

(B.) What is printed when we run the remainder of the code with the above input? Circle your answer.

(C.) What is printed if we run this program with an empty input file? Circle your answer.

5. **Debugging** Consider the following program, which is supposed to read in a list of `N` doubles from `StdIn` and sum them, where `N` is the first command-line argument. For example, we might run this command to sum 100 numbers:

```
% java SumThemUp 100 < MyNumbers.txt
```

```
 1 public class SumThemUp {
 2     public static void main(String[] args) {
 3         int N = Integer.parseInt(args[1]);
 4         double total = 0.0;
 5         for (int i = 0; i < N; i++); {
 6             double x = StdIn.readDouble;
 7             total += x;
 8         }
 9         StdOut.println("Your sum is: " + total);
10     }
11 }
```

This program has several bugs.

(A.) Which bug prevents the program from *compiling* successfully? Identify the line number where the bug appears and give a correct version of this line of code.

Line number:

Correct version:

(B.) After fixing the first bug, which bug causes the program to *crash* when run with the command above? Identify the line number where the bug appears and give a correct version of this line of code.

Line number:

Correct version:

(C.) After fixing the first two bugs, which bug causes the program to produce *incorrect output*? Identify the line number where the bug appears and give a correct version of this line of code.

Line number:

Correct version:

6. **TOY** Note that all numbers in the following program are in hexadecimal. Consider what happens when the following TOY program is executed by pressing `RUN` with the program counter set to 10 (also hexadecimal).

```
0F:     0006
10:     840F     R[4] <- Mem[0F]
11:     4555     R[5] <- 0

12:     7601     ____________________________
13:     1554     R[5] <- R[5] + R[4]

14:     2446     ____________________________
15:     D413     if(R[4] > 0) pc <- 13
16:     95FF     Mem[FF] <- R[5]
17:     0000     Halt
```

The following questions are based on the above program. Please circle your answers.

(A.) Fill in the appropriate pseudocode for the instructions at memory addresses 12 and 14.

(B.) How many times will the instruction at memory address 14 (2446) be executed? Please write your answer in decimal.

(C.) Assume that Mem[0F] may contain any positive integer value, rather than being preloaded with 0006. Let us call this N. Which statement best describes the function of the above TOY program, in terms of N? Circle only one answer.

The program computes the product of all integers from 1 to N

The program computes the sum of all even integers from 1 to N

The program computes the sum of all integers from 1 to N

The program computes the sum of all odd integers from 1 to N

None of the above

(D.) This program will not work correctly for all possible positive TOY integer values of N. **Briefly** describe why not.

TOY REFERENCE CARD

INSTRUCTION FORMATS

```
             | . . . . | . . . . | . . . . | . . . . |
  Format 1:  | opcode  |    d    |    s    |    t    |  (0-6, A-B)
  Format 2:  | opcode  |    d    |       addr        |  (7-9, C-F)
```

ARITHMETIC and LOGICAL operations

```
    1: add              R[d] <- R[s] +  R[t]
    2: subtract         R[d] <- R[s] -  R[t]
    3: and              R[d] <- R[s] &  R[t]
    4: xor              R[d] <- R[s] ^  R[t]
    5: shift left       R[d] <- R[s] << R[t]
    6: shift right      R[d] <- R[s] >> R[t]
```

TRANSFER between registers and memory

```
    7: load address     R[d] <- addr
    8: load             R[d] <- mem[addr]
    9: store            mem[addr] <- R[d]
    A: load indirect    R[d] <- mem[R[t]]
    B: store indirect   mem[R[t]] <- R[d]
```

CONTROL

```
    0: halt             halt
    C: branch zero      if (R[d] == 0) pc <- addr
    D: branch positive  if (R[d] >  0) pc <- addr
    E: jump register    pc <- R[d]
    F: jump and link    R[d] <- pc; pc <- addr
```

Register 0 always reads 0.
Loads from mem[FF] come from stdin.
Stores to mem[FF] go to stdout.
pc starts at 10

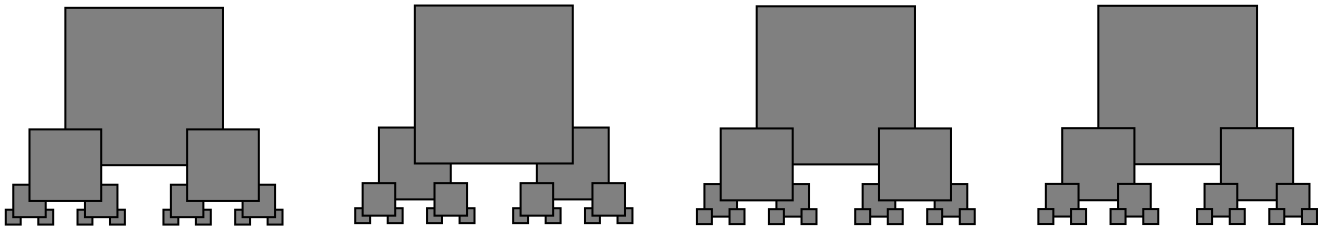
16-bit registers
16-bit memory locations
 8-bit program counter

7. **Recursive Graphics** Consider the following program, recalling that `StdDraw.square(x, y, s)` draws the outline of a square centered at `(x, y)` with side-length `2*s`, and that `StdDraw.filledSquare(x, y, s)` does the same, but colors in the entire area:

```
1  public class Recursion {
2
3      private static void drawShadedSquare( double x, double y, double s) {
4          StdDraw.setPenColor(StdDraw.GRAY);
5          StdDraw.filledSquare(x, y, s);
6          StdDraw.setPenColor(StdDraw.BLACK);
7          StdDraw.square(x, y, s);
8      }
9
10     private static void draw( double x, double y, double s, int level, int k ) {
11         if ( level == 0 ) {
12             drawShadedSquare( x, y, s );
13         }
14         else if ( level % 2 == k ) {
15             drawShadedSquare( x, y, s );
16             draw( x - s, y - s, s/2.2, level-1, 1-k ); // bottom left
17             draw( x + s, y - s, s/2.2, level-1, 1-k ); // bottom right
18         }
19         else {
20             draw( x - s, y - s, s/2.2, level-1, 1-k ); // bottom left
21             draw( x + s, y - s, s/2.2, level-1, 1-k ); // bottom right
22             drawShadedSquare( x, y, s );
23         }
24     }
25
26     public static void main(String[] args) {
27         StdDraw.setScale(-4, 4);
28         draw( 0, 0, 1, Integer.parseInt(args[0]), Integer.parseInt(args[1]));
29     }
30 }
```

Answer the questions about this program on the next page.

(A.) Which of the pictures below would the `draw()` method generate when called with the parameters `level = 3` and `k = 1` ? Circle your answer.



(B.) How many shaded squares would be drawn **uncovered** (that is, with no part of the square obscured by another square) if the `draw()` method is called with parameters `level = 2` and `k = 0`?

(C.) Change lines 16 and 17 so that the leftmost picture from (A) above is generated when draw() is called with the parameters level = 3 and k = 1 . Write your answer below:

8. **Sorting and Analysis** You are provided a method called `tigerSort()`, with this signature:

```
public static void tigerSort(int [] array)
```

The method takes an array of integers as an argument and sorts the array in ascending order using one of the sorting algorithms that we have seen in detail this semester. You called this method with some sample arrays and recorded the execution times in this table:

Array size	Execution time (sec)
50000	5
100000	22
200000	90
800000	1451

(A.) How long would you expect it to take to sort an array of 1.5 million items? Circle your answer:

Under half an hour

1-3 hours

4-6 hours

Approximately half a day

(B.) Based on your timing analysis, which sort could `tigerSort()` be? Circle only one answer:

Insertion Sort

Mergesort

Insertion Sort or Mergesort

Neither Insertion Sort nor Mergesort

(C.) I take an unsorted array of size `N`, execute the following code snippet and record the execution time of **ONLY the second call** to `tigerSort()`.

```
tigerSort(randomUnsortedArray);
tigerSort(randomUnsortedArray);
```

What order of growth will I find? Circle only one answer:

Constant

Linear

Logarithmic

Quadratic