CIS 110 — Introduction to Computer Programming
8 October 2013 — Midterm

Name: _____

Recitation # (e.g., 201): _____

Pennkey (e.g., eeaton): _____

My signature below certifies that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this examination.

_____ _____
Signature Date

Instructions:

- **Do not open this exam until told by the proctor.** You will have exactly 110 minutes to finish it.
- **Make sure your phone is turned OFF (not to vibrate!) before the exam starts.**
- Food, gum, and drink are strictly forbidden.
- **You may not use your phone or open your bag for any reason**, including to retrieve or put away pens or pencils, **until you have left the exam room**.
- This exam is *closed-book, closed-notes, and closed-computational devices.*
- If you get stuck on a problem, it may be to your benefit to move on to another question and come back later.
- All code must be written out in proper java format, including all curly braces and semicolons.
- Do not separate the pages. If a page becomes loose, reattach it with the provided staplers.
- Staple all scratch paper to your exam. Do not take any sheets of paper with you.
- If you require extra paper, please use the backs of the exam pages or the extra pages provided at the end of the exam. **Clearly indicate on the question page where the graders can find the remainder of your work (e.g., "back of page" or "on extra sheet").**
- Use a pencil, or blue or black pen to complete the exam.
- If you have any questions, raise your hand and a proctor will come to answer them.
- When you turn in your exam, you may be required to show ID. **If you forgot to bring your ID, talk to an exam proctor immediately**.
- We wish you the best of luck. Have a great fall break!

Scores: [For instructor use only]

Question 1		9 pts
Question 2		11 pts
Question 3		18 pts
Question 4		11 pts
Question 5		10 pts
Question 6		20 pts
Question 7		11 pts
Question 8		10 pts
Total:		100 pts

1.) MISCELLANEOUS (9 points total)

1.1) (1 point) The Easy One:

- Check to make certain that your exam has all 12 pages (excluding the cover sheet).
- Write your name, recitation number, and PennKey (username) on the front of the exam.
- Sign the certification that you comply with the Penn Academic Integrity Code

1.2) (2 points) How would you access the string "man" from the command line arguments in the following example: `java MyPalindrome a man a plan a canal Panama`

(a) `String man = args[0];`

(b) `String man = args[1];`

(c) `String man = args[2];`

(d) `String man = "man";`

(e) You cannot access the string "man"

1.3) (2 points) Circle the line that has a syntax error.

(a) `String x = "x";`

(b) `int x = 10.0;`

(c) `double x = 10/20;`

(d) `boolean x = (5 > 4);`

(e) None of these lines contain syntax errors

1.4) (2 points) What value is printed as a result of the following code?

```
String str = "Java";
int z = 14 / str.length();
System.out.println(z);
```

(a) 14 / 4

(b) 3

(c) 3.5

(d) None, this code will result in an error

1.5) (2 points) Which ordering is correct from fastest to slowest computational complexity? (For sorting algorithms, assume that the array is in random order; for searching algorithms, assume that the array is already in sorted order.)

(a) merge sort, binary search, linear search, insertion sort

(b) binary search, linear search, merge sort, selection sort

(c) selection sort, merge sort, insertion sort

(d) insertion sort, selection sort, merge sort

(e) None of the above are correct

2.) OPERATORS AND EXPRESSIONS (11 points total)

2.1) (7 points) Give the data type and value that `z` contains in each of the following expressions. Your answers should not involve implicit or explicit casting. If the code fragment would result in an error, write "N/A" in the first column, and give the reason for the error in the second column (you do not need to write the exact error message). The first problem has been completed for you.

	Type	Value
`????? i = 2 + 4;`	`int`	`6`
`????? z = 7; z /= 5;`		
`double y = 9.5; ????? z = (y > 8.2);`		
`int x = 11; ????? z = x % 5;`		
`????? z = "Life = " + 4 + 2`		
`int y = 0, x = 1; ????? z = (y >= 0 && x < 0);`		
`String s = Math.max(4, 2); ????? z = s;`		
`double d = 3.14; ????? z = d - (int) d;`		

2.2) (4 points) Consider the following code fragment:

```
boolean b = false;
if (x > 10) {
    if (y > 10) {
        b = true;
    }
} else if (y <= 0) {
    b = true;
}
```

Re-write this code fragment as a **single boolean expression**, using only comparison operators (`>`, `<`, `<=`, `>=`, `==`, `!=`), logical operators (`!`, `&&`, `||`), literals, and variables (`b`, `x`, `y`). Put your answer in the box below to replicate the code above in a single line.

`boolean b =` [ ] ;

3.) CONDITIONALS (18 points total)

3.1) (6 points)

Consider a function which toggles (i.e., switches) the value for a class-level static boolean variable that tracks whether a light is on or off. Three people, creatively named Alice, Bob, and Claire, who are claiming to be expert light switchers have written different implementations of this function.

All people have the same class definition...

```
public class LightSwitcher {

    // the class-level variable
    static boolean lightOn = false;

    public static void main (String[] args) {
        toggleLight();
        System.out.println(lightOn);
        toggleLight();
        System.out.println(lightOn);
        toggleLight();
        System.out.println(lightOn);
    }

    static void toggleLight() {
        ...
    }
}
```

...but, different implementations of the toggleLight() function:

Alice's Version:

```
static void toggleLight() {
    if (lightOn) {
        lightOn = false;
    }
    else {
        lightOn = true;
    }
}
```

Bob's Version:

```
static void toggleLight() {
    if (lightOn)
        lightOn = false;
    if (!lightOn)
        lightOn = true;
}
```

Claire's Version:

```
static void toggleLight() {
    lightOn = !lightOn;
}
```

Are these implementations equivalent? Are all three people expert light switchers, or is one (or more) frauds? Why? Explain your answer in 30 words or less!

3.2) (12 points) Many fields, such as economics, utilize a framework known as the Producer-Consumer model. In this model, the producer generates something (like a factory manufacturing an item) and the consumer consumes that item if one is available. If no item is available, the consumer does nothing.

Write a program that simulates this Producer-Consumer model using two functions: `produce()` and `consume()`. The `produce()` function prints "Producing...", and `consume()` prints "Consuming..." if there was a previous corresponding `produce()` call (otherwise, it doesn't print anything).

Your implementation must not modify the `main()` given below, but you may add additional functions or variables to the class. (Hint: Recall that static class-level variables can be used to coordinate between multiple functions, since the value in a class-level variable persists between function calls.) You do not need to provide header documentation for your `produce()` and `consume()` functions, but you may wish to include comments to help explain your code. Comments are not required, however. If you need additional space, you can write your program on the following blank page.

```
1  public class ProducerConsumer {
2
3      public static void main (String[] args) {
4          produce();
5          consume();   // matches produce() on line 4
6          produce();
7          consume();   // matches produce() on line 6
8          consume();   // no match, so doesn't output anything
9          produce();
10         produce();
11         consume();   // matches produce() on line 9
12         consume();   // matches produce() on line 10
13         consume();   // no match, so doesn't output anything
14         consume();   // no match, so doesn't output anything
15     }
 .
 .     // YOUR CODE GOES HERE
 .
   }
```

Program Output:

```
Producing...
Consuming...
Producing...
Consuming...
Producing...
Producing...
Consuming...
Consuming...
```

Scrap Paper

(This page is intentionally blank.)

4.) LOOPS (11 points total)

```
int k = 20, i = 10;
while (i > 0) {
    k -= i;
    i -= 4;
    System.out.println(k);
    if (k == 2 * i)  break;
}
```

4.1) (3 points) What is the output of the above code fragment?

4.2) (8 points) Rewrite the above code fragment using a `for` loop instead of a `while` loop. The results of executing the two code fragments should be identical for *any* initial values of `k` and `i`. (The scope of each variable can be different between the `for` and `while` loop versions; only the output must be identical.)

5.) VARIABLE SCOPING (10 points total)

```
1  /************************************************
2  * Usage:
3  *    java MyProgram <a> <b>
4  *    where <a> and <b> are integers
5  *    e.g.,  "java MyProgram 1 2"
6  ************************************************/
7  public class MyProgram {
8      static int val = 5;
9
10     public static void main (String[] args) {
11         int val = Integer.parseInt(args[0]);
12         int val2 = Integer.parseInt(args[1]);
13         System.out.println(val + val2);
14         System.out.println(val * myFunction(val2));
15         System.out.println(val + val2);
16         System.out.println(val * myFunction(val2));
17     }
18
19     public static int myFunction(int x) {
20         int val2 = x * val;
21         for (int i = 0; i < x; i++) {
22             val += x;
23         }
24         return val2;
25     }
26 }
```

5.1) (7 points) What is the output of the program when it is executed by `java MyProgram 3 1`?

5.2) (3 points) What value will the static variable `val` (declared on line 8) contain when the program terminates? (Assume the program was executed by `java MyProgram 3 1`.)

6.) ARRAYS (20 points total)

Write the function for the given header below. (You should not implement the enclosing public class or `main()`; only implement the function itself. You may wish to include comments to clarify your code, but comments are not required.)

```
/*****************************************************************************
* Function Name:  partialRandomArray
* Returns an array of (2 * n) values where:
*    - All even-numbered indices contain consecutive values (starting at zero)
*    - All odd-numbered indices contain random values between 0 and 1
* Example:
*    partialRandomArray(5) =>
*         [0.0, 0.2348, 1.0, 0.9917, 2.0, 0.08742, 3.0, 0.3324, 4.0, 0.7809]
* Arguments:
*     n : the number of random values to generate.
*         The resulting array will have (2 * n) entries.
*****************************************************************************/
```

7.) RECURSION (11 points total)

```
public class MyOperator {
    public static void main (String[] args) {
        int n = op(8, 3);
        System.out.println(n);
    }

    static int op (int a, int b) {
        if (a < b) {
            return a;
        } else {
            return op(a - b, b);
        }
    }

}
```

7.1) (6 points) Trace through the above code by filling in the table below. Each time the `op()` function is called, list the values of the two arguments as a new row in the table (in-order). The first row has been completed for you. When you're finished, cross off any extra rows that you don't use.

	Function	**a = ?**	**b = ?**
1	`op()`	8	3
2	`op()`		
3	`op()`		
4	`op()`		
5	`op()`		
6	`op()`		
7	`op()`		
8	`op()`		

// called by main()

7.2) (4 points) What does the above program print? Justify your answer if you're not certain.

7.3) (1 point) What built-in operator does the `op()` function simulate?

8.) DEBUGGING (10 points total)

George Lucas, the creator of *Star Wars*, appears to have hidden secret information in his script for Episode I. If you take the first character from every script page, it spells out the following java program. Unfortunately, like in the rest of the script for Episode I, George made a number of mistakes.

```
 1  public class StarWars {
 2      public static void main(String[] args) {
 3          System.out.println(starWars());
 4      }
 5
 6      public static String StarWars() {
 7          String[] jedi = {"l","f","4","e","8","w","k","3","j","0","s","7","!"}
 8          int[] phantomMenace = {1, 2, 4, 8};
 9          String[] newHope = "";
10
11          for (int s = 0; s < phantomMenace.length(); s++) {
12             for (int j = 0; j <= phantomMenace.length; j++) {
13                if (j > s) {
14                     newHope += jedi[phantomMenace[i] + phantomMenace[s]];
15                }
16             }
17          }
18          return newHope;
19      }
20  }
```

8.1) (6 points) Find and correct the six bugs in George's program. If you're uncertain how to correct a bug, identify the bug for partial credit. A single line of code may contain multiple bugs.

Bug #	**Line #**	**Corrected Bug**
1		
2		
3		
4		
5		
6		

8.2) (4 points) Find George's secret plot device by determining the output of this (now-completely corrected) program.

Scrap Paper

(This page is intentionally blank.)

Scrap Paper

(This page is intentionally blank.)