CIS 110-001 — Introduction to Computer Programming

5 October 2012 — Midterm

Name: ______________________________

Recitation # (e.g. 201): ______________________________

Pennkey (e.g. bjbrown): ______________________________

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

______________________________ ______________________________
Signature Date

Scores:

1		1
2		4
3		7
4		8
5		15
6		15
Total:		50

CIS 110 Exam Instructions

- You have 45 minutes to finish this exam. Time will begin when called by a proctor and end precisely 45 minutes after that time. If you continue writing after the time is called, you will receive a zero for the exam.
- **Make sure your phone is turned off before the exam starts. If it vibrates or rings during the exam, you will receive a substantial penalty.**
- Because this exam is given twice, back-to-back, a few extra rules apply. Failure to comply may result in a substantial point deduction on your exam.
  - You may not leave the exam room early.
  - Food and drink are strictly forbidden, including water and gum.
  - 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 building**.
  - You must leave the exam room **and building** quickly and quietly through the designated exit. You may not discuss the exam on the way out.
- This exam is *closed-book, closed-notes, and closed-computational devices.* Except where noted, code included in the questions is correct and you may use it as a reference for Java syntax.
- This exam is long. If you get stuck part way through a problem, it may be to your advantage to go on to another problem and come back later if you have time.
- All code must be written out as normal, including all curly braces and semicolons, unless the question states otherwise.
- Do not separate the pages of the exam. If a page becomes loose, write your name on it and use the provided staplers to reattach the sheet when you turn in your exam so that we don't lose it. We reserve the right not to grade any answers on loose sheets of paper.
- Turn in all scratch paper that you use during the exam. Do not take any sheets of paper with you or leave them behind.
- If you require extra paper, please use the backs of the exam pages or the extra sheet(s) of paper 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"). Staple an extra sheets you use to the back of your exam when you turn it in using the provided staplers.
- Use a pencil, or blue or black pen to complete the exam. All other colors are reserved for grading. If you do not have an appropriate writing utensil, raise your hand, and we will give you a pencil.
- If you have any questions, raise your hand and an exam 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.

Good luck, have fun!

Miscellaneous

1. (1 points)

   (a) Write your name, recitation number, and PennKey (username) on the front of the exam.

   (b) Sign the certification that you comply with the Penn Academic Integrity Code

Money, money, money, money, money ...

2. (4 points) Consider the following recursive function:

```
public static double compoundInterest(double balance, int months, double rate) {
  if (months <= 0) return balance;
  balance = balance * rate;
  return compoundInterest(balance, months - 1, rate);
}
```

   (a) Is this function tail recursive (circle your answer)? YES NO MAYBE

   (b) How many times will `compoundInterest(0, 12, 1.05)` call itself recursively? ____________

   (c) What value will `compoundInterest(0, 12, 1.05)` return? ____________

   (d) Will the following non-recursive function produce the same or different results from the recursive function above? Circle your answer.

SAME DIFFERENT

```
public static double iterativeInterest(double balance, int months, double rate) {
  for (int i = months; i >= 0; i--)
    balance = balance * rate;
  return balance;
}
```

Cabin

3. (7 points) Given the following class, answer the questions below and on the next page:

```
public class Fever {
  public static int restless(int[] arr) {
    int sum = 0;
    for (int i = 0; i < arr.length; i++)
      sum = sum + arr[i];
    return sum / arr.length;
  }

  public static void main(String[] args) {
    int[] intArgs = new int[args.length];
    for (int i = 0; i < args.length; i++)
      intArgs[i] = Integer.parseInt(args[i]);
    System.out.println(restless(intArgs));
  }
}
```

(a) Circle the statements below that are true

i. will not compile because the `for` loops do not have curly braces;

ii. will not compile because `restless()` is called inside `System.out.println`;

iii. `intArgs[]` is `args[]` converted into an integer array;

iv. may crash with an `ArrayIndexOutOfBounds` error when run;

v. does not print anything because the `intArgs[]` array is different from the `arr[]` array ;

vi. prints the approximate average of the command line arguments.

(b) Assuming that any errors you identified above are corrected, what will the program print out in each of the following cases? We will consider your answer correct as long as it shows you understand what the program will print out.

i. `% java Fever`

ii. `% java Fever 1.3 2.5 7.1`

iii. `% java Fever 1 2 5`

iv. `% java Fever hello world`

Find the Bugs

4. (8 points) Identify eight errors in the following program that prevent it from compiling or from running and show how to correct them. Write your answers in the space provided on the next page. The line numbers are for your convenience and are not part of the program.

```
 1: public class Foo() {
 2:   public static int main(String[] args) {
 3:     int rows = args[0];
 4:     drawSomething(Rows);
 5:     double hyp = computeSomething(rows);
 6:     System.out.println("length(hypotenuse)" = hyp);
 7:   }

 8:   public static void drawSomething(int rows) {
 9:     for (i = 0; i < rows; i++) {
10:       for (j = 0; j < i + 1; j = j + 1)
11:         System.out.print("*");
12:       System.out.println();
13:     }
14:   }

15:   public static double computeSomething(int leg) {
16:     int sqrt2 = Math.sqrt(2);
17:     return sqrt2 * leg;
18:   }
19: }
```

Bug 1:

Bug 2:

Bug 3:

Bug 4:

Bug 5:

Bug 6:

Bug 7:

Bug 8:

Partial Sums

5. (15 points) The median, or middle value, of a list of numbers is extremely useful in a variety of computer and statistical algorithms. But it is notoriously slow to compute relative to the number of times it needs to be computed. Often, it is better to settle for an approximation of the median than to calculate it exactly. One of the simplest approximations is to pick three random elements, and calculate the median of those three.

Write a function `threeMedian` that takes a single argument `N`, reads in `N` `double`s from standard input using `StdIn.readDouble()`, and returns the median of three randomly chosen elements. Assume that `N` is always at least three, and that all calls to `StdIn.readDouble()` succeed without error. Use `Math.random()` to pick the three random elements (recall that `Math.random()` returns a random number between 0 and 1, but never exactly 1). Do not write the code for the class that contains `threeMedian`, only write the function itself. You do not need to comment your code.

Tracery

6. (15 points)

For each of the labeled points in the code fragment below, identify each of the assertions in the table as being *sometimes*, *always*, or *never* true. Assume that `bar` is only called from within `foo`, and that the values of all `int`s stay within the valid range for integers (i.e. no value will grow so large that it will wrap around become negative, or vice cersa).

Abreviate sometimes with **S**, always with **A**, and never with **N**.

```
public static void foo(int x, int y, int z) {
  if (x > y && y <= 0)
    y = Math.abs(y - x) + 1;
  else if (x < y)
    z = Math.abs(z) + 1;
  else
    x = y - 1;

  // Point A

  if (z > x) {
    z = x;
    x = y * y + 2;
    // Point B
  } else {
    z = x;
    z = bar(y, Math.abs(x), z * z);
    // Point C
  }

  if (y == x)
    y--;

  // Point D
}

public static int bar(int z, int x, int y) {
  for (int i = 0; i < x; i++)
    y = y - x;

  // Point E

  return z + y;
}
```

	x > 0	x > y	z > y
A			
B			
C			
D			
E			

Postscript (extra paper)

Postscript (extra paper)