CIS 110 Spring 2013 Final Exam, 29 April 2013, Answer Key

Miscellaneous

0. (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

Multiple Choice

1. (10 points) For each of the following questions, circle the correct answer:

   (a) What happens if you try to access a private variable from outside the class in which it is declared?

   i. The program runs normally.
   ii. The program runs, but treats the variable as having the value 0 or `null`.
   iii. The program runs, but the statement accessing the private variable is ignored.
   iv. The program runs, but behaves unpredictably.
   v. The program compiles, but a run-time error occurs.
   vi. The program does not compile. ⇐

   (b) `5 / 2 * Math.random()` generates a random number between...

   i. 0 and 2.5.
   ii. 0 and 2. ⇐
   iii. 2 and 5.
   iv. 2.5 and 5.
   v. 2 and ∞.
   vi. 2.5 and ∞.

Multiple Choice (Cont'd)

(c) What does the following function return?

```
public static boolean foo(boolean x, boolean y, boolean z) {
  if (x && y) return !x || z;
  else        return (x && y) || z;
}
```

i. `x`
ii. `y`
iii. `z` ⇐
iv. `x && y`
v. `true`
vi. `true` if either `x` and `y` are both `true` or `z` is `true`, and `false` otherwise

(d) The `toString()` method of an object should do which of the following?

i. Print the object to standard output in a sensible way.
ii. Return a sensible `String` representation of the object. ⇐
iii. Print the object to standard output in a sensible way and return that output as a `String`.
iv. Whatever the object's designer wants it to do.
v. Read in the contents of the object from standard input.
vi. None of the above

(e) Which of the following expressions computes the length of a string `s`?

i. `s.length`
ii. `s.length()` ⇐
iii. `String.s.length`
iv. `String.s.length()`
v. `String.length`
vi. `String.length()`

Easy as ABC

2. (12 points)

For each question below, what will the code snippet print out? If there is an error, your answer should show that you know what causes the error, but you do not need to reproduce Java's error message exactly. Put a box around your answer so we can distinguish it from your scratch work.

(a)
```
B b = new B(7, "bad");
A a = new A(b);
System.out.println(a.get());
```
```
Exception in thread "main" java.lang.NullPointerException
  at A.get(...)
  ...
```

(b)
```
B b = new B(7, "bad");
A a = new A(b, b.c);
System.out.println(a.get());
```
```
bad bad
```

(c)
```
B b = new B(7, "bad");
A a = new A(b, b.c);
System.out.println(a.c);
```
```
bad
```

(d)
```
C b = new C();
System.out.println(b.c);
```
```
Exception in thread "main" java.lang.NullPointerException
  at C.<init>(...)
  ...
```

Toying With TOY

3. (20 points)

Consider the following TOY program, then answer the questions on the following page. You may assume that all assembly comments are correct.

```
10: 81FF   read R[1]
11: 82FF   read R[2]
12: 7400   R[4] <- 0000
13: 7501   R[5] <- 0001
14: 7602   R[6] <- 0002
15: 2321   R[3] <- R[2] - R[1]
16: D31B   if (R[3] > 0) goto 1B
17: 1445   R[4] <- R[4] + R[5]
18: C31C   if (R[3] == 0) goto 1C
19: 2112   R[1] <- R[1] - R[2]
1A: C015   if (R[0] == 0) goto 15
1B: 2323   R[3] <- R[2] - R[3]
1C: 94FF   write R[4]
1D: C622   if (R[6] == 0) goto 22
1E: 1444   R[4] <- R[4] + R[4]
1F: 2665   R[6] <- R[6] - R[5]
20: 1333   R[3] <- R[3] + R[3]
21: C01D   if (R[0] == 0) goto 1D
22: C226   if (R[2] == 0) goto 26
23: 1646   R[6] <- R[4] + R[6]
24: 2225   R[2] <- R[2] - R[5]
25: C022   if (R[0] == 0) goto 22
26: 1563   R[5] <- R[6] + R[3]
27: 95FF   write R[5]
28: 0000   halt
```

(a) For each pair of numbers below, what will the program write to standard output? Write your answers in decimal (base 10) and put a box around it.

i. 1 and 1 1 and 4

ii. 2 and 1 2 and 8

iii. 1 and 2 0 and 4

iv. 0 and 1 0 and 0

v. 0 and 0 1 and 0

vi. 1 and 0 infinite loop (we also accepted the answer "nothing")

(b) In simple formulas, what values are outputted if the first input is **larger** than the second?

`a / b` and `4 * a`

(c) In simple formulas, what values are outputted if the first input is **smaller** than the second?

`0` and `4 * a`

The Tree of Knowledge

4. (25 points)

(a) If the following code is executed:

```
int[] vals = { 3, 7, 8, 2, 9, 1, 6, 4 };
Knowledge k = new Knowledge();
k.apple = 5;
for (int i = 0; i < vals.length; i++) k.insert(vals[i]);
```

i. What will `k.A()` print? `9 8 7 6 5 4 3 2 1`

ii. What will `k.B()` print? `5 7 8 9 6 3 4 2 1`

iii. What will `k.C()` print? `1 2 4 3 6 9 8 7 5`

(b) For each of the outputs below, say which of `A()`, `B()`, and/or `C()` could have produced it. For each answer, give an order in which you could insert the elements to produce the output. Write "none" if none of them could have produced it. Put a box around your answer so we can distinguish it from your scratch work.

Assume you create a single `Knowledge` object, then progressively add values using the `insert()` method, just like the code snippet in part (a).

i. `1 2 3 4 5` `B()` (insert e.g. 1, 2, 3, 4, 5) or
`C()` (insert e.g. 5, 4, 3, 2, 1)

ii. `5 4 3 2 1` `A()` (insert in any order),
`B()` (insert e.g. 5, 4, 3, 2, 1), or
`C()` (insert e.g. 1, 2, 3, 4, 5)

iii. `2 3 4 3` `B()` (insert e.g. 2, 3, 4, 3), or
`C()` (insert e.g. 3, 4, 2, 3)

Leave and Let Leave

5. (30 points)

(a) Write a recursive version of `interleave()`.

```
public static Leave interleave(Leave a, Leave b) {
    if (a == null) return b;        // base case
    a.next = interleave(b, a.next); // this is clever; draw a picture
    return a;                       // all done!
}
```

(b) Write an iterative version of `interleave()`.

```
public static Leave interleave(Leave a, Leave b) {
    if (a == null) return b;  // special case for a is null
    Leave head = a, tail = a; // interleaved lists starts at first node of a
    a = a.next;               // advance a because we already took its first node

    // as long as both lists have at least one more element
    while (a != null && b != null) {
        tail.next = b;            // append first node of b to interleaved list
        b = b.next;               // advance to next node of b
        tail.next.next = a;       // append first node of a to interleaved list
        a = a.next;               // advance to next node of a
        tail = tail.next.next;    // update tail of list (we just added two nodes)
    }
    if (b == null) tail.next = a; // append anything left at end of a or b
    else           tail.next = b;

    return head;                  // return interleaved list
}

-- or --

public static Leave interleave(Leave a, Leave b) {
  if (a == null) return b;  // special case for a is null
  Leave head = a, tail = a; // interleaved lists starts at first node of a

  for (a = a.next; b != null; tail = tail.next) {
    tail.next = b;          // append first node of b to interleaved list
    // swap a and remaining nodes of b
    Leave tmp = a;
    a = b.next;
    b = tmp;
  }

  tail.next = a; // append rest of a to the end of the interelaved list
  return head;   // return the interleaved list
}
```