CS61B - Guerrilla Section 1 Solutions

Q1. Potpourri

a. Fix the bugs in the Knapsack class below, so that it works as expected:

```
class Knapsack {
      public String thing;
      ~~public String amount;~~
      public double amount;

      public Knapsack(String str, double amount) {
            ~~String~~ thing = str;
            this.amount = amount;
      }

      public Knapsack(String str) {
            ~~Knapsack(str, 100.45);~~
            this(str, 100.45)
      }

      public static void main(String ... args){
            Knapsack sack = new Knapsack("Doge coin");
            System.out.println(sack.thing + " : " + sack.amount);
      }
}
```

b. Type Casting of a variable (e.g. `(String) x`) permanently changes:

a) Its static type
b) Its dynamic type
c) a) and b)
d) None of the above.

c. What is the output of the following code fragment:

```
String s1 = "I <3 61B";
String s2 = s1;
String s3 = new String("I <3" +
                       " 61B";)
String s4 = new String("I <3 61B");
```

```
System.out.println(s1 == s2);
True
System.out.println(s1 == s3);
False
System.out.println(s1 == s4);
False
```

Q2. References

Quick terminology reminder: Java variables are simple containers that can hold either primitive values or references to Objects. The contents of the container are its value.

a. Which (if any) of the following method calls has an effect on the value of x or y? Which of these method calls has an effect that we could observe using a print statement on the next line after the call?

```
static void copy1(int a, int b) {
   a = b;
}

int x = 5;
int y = 10;
copy1(x, y) /* effect on x/y val? */
               No effect.
            /* possible to print evidence of copy1()? */
               No.

static void copy2(String a, String b) {
   a = b;
}

String x = "melted";
String y = "cheese";
copy2(x, y); /* effect on x/y val? */
                No effect.
             /* possible to print evidence of copy2()? */
                No.

static void copy3(String[] a, String[] b) {
   a[0] = b[0];
}

String[] x = new String[] {"friendly", "pickly"};
String[] y = new String[] {"bodacious", "horse"};
copy3(x, y);  /* effect on x/y val? */
No effect since x and y still contain the same object
references (although, one could argue that x[0] has been
changed).
              /* possible to print evidence of copy3()? */
                 Yes.
```

b. Suppose we wanted to write a helper method of type void to be used for doubling the size of an int array. Explain why it's impossible to do this, i.e. in the code below, why is it impossible for the call to doubleSize to affect the size of a?

```
private static void doubleSize(int[] a) {
      /* … */
}

int[] x = new int[]{1, 2, 3};
doubleSize(x); /* why desired effect impossible? */
```

Array size is immutable and we can't just make x point to another array since java is pass-by-value.

Q3. Probably Equal? - Su '14 MT1

Write a probablyEquals method that takes in two objects and returns true if one or more of the following are true:

- The two objects are equal (.equals) to each other.
- The two objects are equal (==) to each other.
- The two objects have the same .toString() representation.

Otherwise, probablyEquals returns false. Your method should never crash given ***any*** input. You may assume that for any object instances x and y, x.equals(y) will return the same value as y.equals(x). You may also assume that every object has a unique non-random toString representation.

Note: .equals(Object o) and .toString() are methods that every object subclass inherits from the Object class.

```
public static boolean probablyEquals(Object x, Object y) {

      if (x == y) {
            return true;
      }

      if(x == null || y == null) {
            return false;
      }

      if (x.equals(y)) {
            return true;
      }

      if (x.toString().equals(y.toString())) {
            return true;
      }

      return false;
}
```

Q4. Static vs. Dynamic Types

For each problem below, cross out any lines that cause errors and state whether it is a run-time or compilation error. Give the results of every print statement. You may need to consult the official documentation for the object class.

```
public class Animal {
    public int numLegs = 4;
    public void eat() {
        System.out.println(
            “Generic” +
            “ animal is eating”);
    }
}
```

```
public class Bird extends Animal {
    public int numLegs = 2;
    public void eat() {
        System.out.println(
            “The bird” +
            “ is eating.");
    }
```

```
public class Run {

    public static void main(String[] args) {
        Animal a = new Animal();
        Bird b = new Bird();
        Animal birdAnimal = b;

        // What will java print?
        System.out.println(a.numLegs);//prints 4
        System.out.println(b.numLegs);//prints 2
        System.out.println(birdAnimal.numLegs);//prints 4
        System.out.println(((Animal) b).numLegs);//prints 4
        System.out.println(((Bird) birdAnimal).numLegs);//prints 2

        // What will java do?
        a.eat();//prints “Generic animal is eating”
        b.eat();//prints “The bird is eating.”
        birdAnimal.eat();//prints “The bird is eating.”
        ((Animal) b).eat();//prints “The bird is eating.”
        ((Bird) birdAnimal).eat();//prints “The bird is eating.”

        ~~double d = ((Object) "string").length();~~ Compile time error
    }
}
```

Q5. Superclass Method Overriding and Hiding

(From Oracle's tutorials) Consider the following two classes (feel free to use a computer to run experiments):

```
public class ClassA {
    public void methodOne(int i) {
    }
    public void methodTwo(int i) {
    }
    public static void methodThree(int i) {
    }
    public static void methodFour(int i) {
    }
}

public class ClassB extends ClassA {
    public static void methodOne(int i) {
    }
    public void methodTwo(int i) {
    }
    public void methodThree(int i) {
    }
    public static void methodFour(int i) {
    }
}
```

a. Which method overrides a method in the superclass?
methodTwo()

b. Which method hides a method in the superclass?
methodFour()

c. Which methods cause compile errors?
methodThree() and methodOne()

Q6: Method Overriding and Hiding (CS61B Fall 2013 Discussion 4)

```
/* The following program doesn't compile. Fix the program by crossing out all
five erroneous lines of Main, and then determine the static and dynamic types
of each variable, as well as the output of the program. Also,
String.valueOf(x) converts a primitive x to a String. */
public class Main {
      public static void main(String[] args) {
            Machine m1 = new Machine();
            Machine m2 = new Drone();
            FlyingMachine f = new Plane();
            Plane p = new Drone();
            Drone d = p;
            System.out.println(m1.run(5));
            System.out.println(m2.run(6)); '6?'
            System.out.println(f.run(9.0)); '9.0-P'
            System.out.println(p.run(15.0)); '15.0!'
            System.out.println(m2.run("hello"));
            System.out.println(f.run("world")); 'world cannot fly.'
            System.out.println(p.run("hello world"));
            'hello world is inferior to plane'
            System.out.println(p.s); 'foo'
            System.out.println(d.s);
      }
}
interface Machine {
      public String run(int x);
}
abstract class FlyingMachine {
      public static String run(String s) { return s + " cannot fly."; }
      public abstract String run(double f);
}
class Plane extends FlyingMachine {
      String s = "foo";
      public String run(double x) { return String.valueOf(x) + "-P"; }
      public static String run(String s) {
            return s + " is inferior to plane.";
      }
}
class Drone extends Plane implements Machine {
      String s = "bar";
      public String run(double x) { return String.valueOf(x) + "!"; }
      public String run(int x) { return String.valueOf(x) + "?"; } }
```

Bonus Problems: More Practice with Recursive Data Structures and References

Q7. Merging IntLists

```
/** Given two sorted IntLists A and B, return an IntList that contains the
 *  elements of both A and B in sorted order. This should be done recursively
 *  A -> {1, 5, 7}
 *  B -> {2, 3, 6, 9}
 *  mergeRecursive(A, B) -> {1 ,2 , 3,  5, 6, 7, 9}
public static IntList mergeRecursive(IntList A, IntList B) {
      //YOUR CODE HERE
      if (B == null) {
            return A;
      }
      if(A == null) {
            return B;
      }
      if (A.head < B.head) {
            A.tail = mergeRecursive(A.tail, B);
           return A;
      } else {
           B.tail = mergeRecursive(A, B.tail);
           return B;
      }


}
```

```
/** Given two sorted IntLists A and B, return an IntList that contains the
 *  elements of both A and B in sorted order. This should be done iteratively
 *  A -> {1, 5, 7}
 *  B -> {2, 3, 6, 9}
 *  mergeIterative(A, B) -> {1 ,2 , 3,  5, 6, 7, 9}
public static IntList mergeIterative(IntList A, IntList B) {
      //YOUR CODE HERE
     if (B == null) {
            return A;
     }
     if(A == null) {
      return B;
     }
      IntList m, res;
     if (A.head < B.head) {
      m = A;
      A = A.tail;
     } else {
           m = B;
           B = B.tail;
     }
      res = m;
     while (A != null && B != null) {
      if (A.head < B.head) {
            m.tail = A;
            A = A.tail;
            } else {
            m.tail = B;
            B = B.tail;
            }
            m = m.tail;
     }
     if (A == null) {
      m.tail = B;
     } else {
            m.tail = A;
     }
     return res;
}
```

Q8. Every Other - Summer '14 Final

Write an evenOdd method that destructively sets the passed in IntList to contain every other IntList node of the original IntList, starting with the first node. Your method must also return an IntList that contains every other IntList node of the original IntList, starting with the second node. Your method should work destructively and should not create any new IntList objects.

If an IntList contains zero elements or only one element, a call to evenOdd should return null.

Example: If an IntList initially contains the elements [5, 2, 3, 1, 4], then a call to evenOdd should return an IntList with the elements [2, 1], and after the call, the original IntList should contain the elements [5, 3, 4]

```
public static IntList evenOdd(IntList L) {
      // YOUR CODE HERE
      if (L == null || L.next == null) {
            return null;
      }
      IntList curr = L;
      IntList ret = L.tail;
      IntList retC = ret;
      while(curr != null && retC != null) {
            curr.tail = retC.tail;
            curr = curr.tail;
            if (curr != null) {
                  retC.tail = curr.tail;
                  retC = retC.tail;
            }
      }
      return ret;
}
```

Q9. Round-Robin - Fall '11 MT1

Write a function distribute(IntList L, IntList[] R) that distributes the elements of L to the lists in the array R in round-robin fashion.
That is, if m = R.length, then the first item in L is appended to R[0], the second to R[1], ..., the mth item to R[m-1], the m + 1st to R[0], etc. So if R starts out containing the IntList sequences [1, 2], [3, 4, 5], and [], and L starts out containing [6, 7, 8, 9], then distribute(L, R) causes R to end up containing [1, 2, 6, 9], [3, 4, 5, 7], and [8]. This method may destroy the original list L. You must not create any new IntList elements. *HINT:* A helper function might be useful.

```
public class IntList {
      public int head;
      public IntList tail;
}

public static void distribute(IntList L, IntList[] R) {
      //YOUR CODE HERE
      int i = 0;
      while (L != null) {
            IntList p = L;
            L = L.tail;
            R[i % R.length] = insertAtEnd(R[i % R.length], p);
            i += 1;
      }
}

static IntList insertAtEnd(IntList a, IntList b) {
      IntList c = a;
      if (b == null) {
            return a;
      }
      b.tail = null;
      if (a == null) {
            return b;
```

```
        }
        while (a.tail != null) {
              a = a.tail;
        }
        a.tail = b;
        return c;
}
```

Q3. Inheritance - HKN review

Consider the following two classes, and answer the questions that follow:

```
public class Superhero {
	String s;

	public Superhero() {
		s = "I'M A SUPERHERO";
		System.out.println(s);
	}
	public void punch() {
		 System.out.println("Punch! Punch!");
	}
	public void punch(Superhero a) {
		 System.out.println("BOOM " + s);
	}
}
```

```
public class Batman extends Superhero {
	String s;

	public Batman() {
		 s = "NANANANANANANA";
	}

	public Batman(String s) {
		this.s = s;
		System.out.println(this.s);
	}

	public void punch(Superhero v) {
		s = "BATMAN!";
		super.punch(v);
		System.out.println("BOOM " + s);
	}

	private void punch(Batman b) {
		System.out.println("Wat.");
	}
}
```

Q. What, if any, is the output of the following code fragment:

```
Superhero superhero = new Superhero();
superhero.punch(superhero);
```

“I’M A SUPERHERO”
"BOOM I'M A SUPERHERO"

Q. What, if any, is the output of the following code fragment:

```
Batman batman = new Batman(“I’m Batman!”);
batman.punch(batman);
```

"I'M A SUPERHERO"
"I'M BATMAN!"
"wat."

Q. What, if any, is the output of the following code fragment:

```
Batman batman = new Superhero();
batman.punch(batman);
```

COMPILE-TIME ERROR

Q. What, if any, is the output of the following code fragment:

```
Superhero superhero = new Batman();
superhero.punch((Batman) superhero);
```

"I'M A SUPERHERO"
"BOOM I'M A SUPERHERO"
"BOOM BATMAN!"

Q. What, if any, is the output of the following code fragment:

```
Batman batman = new Batman();
((Superhero) batman).punch(batman);
```

"I'M A SUPERHERO"
"BOOM I'M A SUPERHERO"
"BOOM BATMAN!"

Q. What, if any, is the output of the following code fragment:

```
Superhero superhero = new Superhero();
superhero.punch((Batman) superhero);
RUNTIME ERROR
```

Q. What, if any, is the output of the following code fragment:

```
Superhero superhero = new Batman();
superhero.punch((Batman) superhero);
"I'M A SUPERHERO"
"BOOM I'M A SUPERHERO"
"BOOM BATMAN!"
```

Q. Now, imagine that we didn't have the punch(Superhero) method in the Superhero class. What, if any, is the output of the following code fragment:

```
Batman batman = new Superhero();
batman.punch(batman);
COMPILE-TIME ERROR

//Intended question:
Batman batman = new Batman();
((SuperHero) batman).punch(batman);
COMPILE-TIME ERROR
```