

1 Give em the 'Ol Switcheroo

For each function call in the main method, write out the x and y values of both foobar and baz after executing that line. (Spring '15, MT1)

```
1 public class Foo {
2     public int x, y;
3
4     public Foo (int x, int y) {
5         this.x = x;
6         this.y = y;
7     }
8     public static void switcheroo (Foo a, Foo b) {
9         Foo temp = a;
10        a = b;
11        b = temp;
12    }
13    public static void fliperoo (Foo a, Foo b) {
14        Foo temp = new Foo(a.x, a.y);
15        a.x = b.x;
16        a.y = b.y;
17        b.x = temp.x;
18        b.y = temp.y;
19    }
20    public static void swaperoo (Foo a, Foo b) {
21        Foo temp = a;
22        a.x = b.x;
23        a.y = b.y;
24        b.x = temp.x;
25        b.y = temp.y;
26    }
27
28    public static void main (String[] args) {
29        Foo foobar = new Foo(10, 20);
30        Foo baz = new Foo(30, 40);
31        switcheroo(foobar, baz);    foobar.x: ___ foobar.y: ___ baz.x: ___ baz.y: ___
32        fliperoo(foobar, baz);    foobar.x: ___ foobar.y: ___ baz.x: ___ baz.y: ___
33        swaperoo(foobar, baz);    foobar.x: ___ foobar.y: ___ baz.x: ___ baz.y: ___
34    }
35 }
```

2 IntList to Array

For this problem we will implement a version of `arraycopy` that copies elements from an `IntList` into an array of `ints`. As a reminder, here is the `arraycopy` method:

```
1 System.arraycopy(Object src, int sourcePos, Object dest, int destPos, int len)
```

`System.arraycopy` copies `len` elements from array `src` (starting at index `source`) to array `destArr` (starting from index `dest`).

To simplify things, let's restrict ourselves to using only `int[]`, and assume that `srcList` and `destArr` are not null. Additionally, assume that `sourcePos`, `destPos`, and `len` will not cause an `IndexOutOfBoundsException` to be thrown.

For example, let `IntList L` be (1 -> 2 -> 3 -> 4 -> 5) and `int[] arr` be an empty array of length 3. Calling `arrayCopyFromIntList(L, 1, arr, 0, 3)` will result in `arr={2, 3, 4}`.

```
1 /** Works just like System.arraycopy, except srcList is of type IntList. */
2 public static void arrayCopyFromIntList(IntList srcList, int sourcePos,
3     int[] destArr, int destPos, int len) {
4
5     for ( _____; _____; _____ ) {
6
7         _____ = _____;
8     }
9
10    for ( _____; _____; _____ ) {
11
12        _____ = _____;
13
14        _____ = _____;
15
16    }
17 }
```

3 Static Books

Suppose we have the following `Book` and `Library` classes.

```

class Book {
    public String title;
    public Library library;
    public static Book last = null;

    public Book(String name) {
        title = name;
        last = this;
        library = null;
    }

    public static String lastBookTitle() {
        return last.title;
    }

    public String getTitle() {
        return title;
    }
}

class Library {
    public Book[] books;
    public int index;
    public static int totalBooks = 0;

    public Library(int size) {
        books = new Book[size];
        index = 0;
    }

    public void addBook(Book book) {
        books[index] = book;
        index++;
        totalBooks++;
        book.library = this;
    }
}

```

- (a) For each modification below, determine whether the code of the `Library` and `Book` classes will compile or error if we **only** made that modification, i.e. treat each modification independently.
1. Change the `totalBooks` variable to **non static**
 2. Change the `lastBookTitle` method to **non static**
 3. Change the `addBook` method to **static**
 4. Change the `last` variable to **non static**
 5. Change the `library` variable to **static**

- (b) Using the `Book` and `Library` classes from before, write the output of the `main` method below. If a line errors, put the precise reason it errors and continue execution.

```

1  public class Main {
2      public static void main(String[] args) {
3          System.out.println(Library.totalBooks);           -----
4          System.out.println(Book.lastBookTitle());        -----
5          System.out.println(Book.getTitle());             -----
6
7          Book goneGirl = new Book("Gone Girl");
8          Book fightClub = new Book("Fight Club");
9
10         System.out.println(goneGirl.title);               -----
11         System.out.println(Book.lastBookTitle());        -----
12         System.out.println(fightClub.lastBookTitle());   -----
13         System.out.println(goneGirl.last.title);         -----
14
15         Library libraryA = new Library(1);
16         Library libraryB = new Library(2);
17         libraryA.addBook(goneGirl);
18
19         System.out.println(libraryA.index);               -----
20         System.out.println(libraryA.totalBooks);         -----
21
22         libraryA.totalBooks = 0;
23         libraryB.addBook(fightClub);
24         libraryB.addBook(goneGirl);
25
26         System.out.println(libraryB.index);               -----
27         System.out.println(Library.totalBooks);          -----
28         System.out.println(goneGirl.library.books[0].title); -----
29     }
30 }
```