CS 61BL Quiz 02

Name: SID:

Section Number: (01) (02) (03) (04) (05) (06) (07) (08) (09) (10) (11) (12)

Write your name and Cal ID number above. Fill in the circle corresponding to your section number. Complete this quiz without using a Java interpreter, and without consulting anyone other than CS 61BL staff. **If you choose an answer, fill in that circle or square completely.**

Refer to the following code to answer questions 1 and 2.

```
public class SuperClass {
    int instVar; int foo;

    public SuperClass() { instVar = 0; }

    public void change() {
        instVar = instVar + 2;
    }

    public String toString() {
        return "" + instVar;
    }
}
```

```
public class SubClass extends SuperClass {
    int bar;

    public void change() {
        instVar = instVar - 1;
    }

    public void zoom() { bar = 1000; }
}
```

1. The Warm Up

For each of the rows, bubble one of the following: `Compile Error`, `Runtime Error` or `No Error`.

SuperClass a = new SuperClass(); a.bar = 5;	◯ Compile Error	◯ Runtime Error	◯ No Error
SubClass b = new SubClass(); b.foo = 10;	◯ Compile Error	◯ Runtime Error	◯ No Error
SuperClass c = new SuperClass(); c.zoom();	◯ Compile Error	◯ Runtime Error	◯ No Error
SuperClass d = new SuperClass(); ((SubClass) d).change();	◯ Compile Error	◯ Runtime Error	◯ No Error
SuperClass e = new SubClass(); e.zoom();	◯ Compile Error	◯ Runtime Error	◯ No Error
SuperClass f = new SuperClass(); ((SubClass) f).zoom(); System.out.println(f.bar);	◯ Compile Error	◯ Runtime Error	◯ No Error

2. Castaway

Consider the following program:

```
SuperClass obj = new SubClass();
// blank line to be filled in
System.out.println(obj);
```

For each of the following statements, determine whether using that statement to replace `//blank line to be filled in` will cause the program to print `-1`. Choose at least one answer, and there may be multiple correct answers.

- ☐ `((Subclass) obj).change();`
- ☐ `obj.change();`
- ☐ `((SuperClass) obj).change();`
- ☐ `None of the above`

3. I'm the Captain Now

Refer to the following `IntList` class to answer question 3.

```
public class IntList {
    private int item; private IntList next;

    public IntList(int item, IntList next) {
        this.item = item;
        this.next = next;
    }
}
```

Fill in the code below for the method `skipper`. `skipper` takes in an `IntList` and returns a *new* `IntList` with every other item from the original list. For example, if an `IntList` {1, 2, 3, 4, 5} is passed in, `skipper` would return a new `IntList` containing {1, 3, 5}. It must leave the original `IntList` unchanged.

```
public static IntList skipper(Intlist lst) {

    if (____________________) {

        return ____________________________________________________;
    }

    if (____________________________) {

        return ____________________________________________________;
    }

    return ____________________________________________________;
}
```