Lecture 6 - Sep. 24

Review of OOP, Exceptions

Static Variables, Common Errors
Caller vs. Callee

Announcements/Reminders

- Lab1 released
- Mockup Programming Test this Fri (5pm or 6pm)
- Guides for WrittenTest1 and ProgTest1 to be released
- Reminder of rules for class attendance checks

Managing Account IDs: Manual


```
public class Account {
  private int id;
  private String owner;
  public int getID() { return this.id; }
  public Account(int id, String owner) {
    this.id = id;
    this.owner = owner;
  }
}
```

explicit parameter, presumably not duplicated

```
class AccountTester {
  Account acc1 = new Account(1, "Jim");
  Account acc2 = new Account(1, "Jeremy");
  System.out.println(acc1.getID() != acc2.getID());
}
```

1. burden on the user of Account to specify unique id.

2. not a compilation error if ids are duplicated.

Declaring Global Variables among Objects

if not init. in the same line, it will be init. to default value (not const.).

* each instance/object of Counter has its own copy (instance-specific value).
** all instances share the same copy (global).

```
public class Counter {
    private int l;
    static int g = 0;

    public Counter() {
        this.l = 0;
    }

    public int getLocal() {
        return this.l;
    }

    public void incrementLocal() {
        this.l ++;
    }

    public void incrementGlobal() {
        g ++;
    }
}
```

non-static var. (attributes)

static var (global).

only init. non-static var. in const.

this.g++ / Counter.g++ (warning)

To Ref. Static Var (no need of C.O.)

ClassName.VarName g

```
public class CounterTester {
    public static void main(String[] args) {
        Counter c1 = new Counter();
        Counter c2 = new Counter();

        System.out.println("c1's local: " + c1.getLocal());
        System.out.println("c2's local: " + c2.getLocal());
        System.out.println("Global accessed via c1: " + c1.g);
        System.out.println("Global accessed via c2: " + c2.g);
        System.out.println("Global accessed via Counter: " + Counter.g);

        c1.incrementLocal();
        c2.incrementLocal();
        c1.incrementGlobal();
        c2.incrementGlobal();

        Counter.g = Counter.g + 1; // Counter.global ++;
    }
}
```

compiles but warnings

0 0 0 0

c1 c2

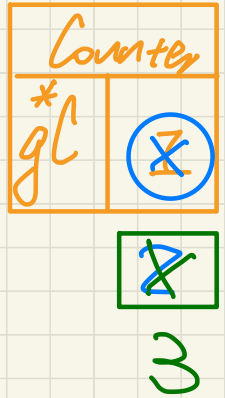
c1.g 1 c2.g 1

c1.g 2 c2.g 2

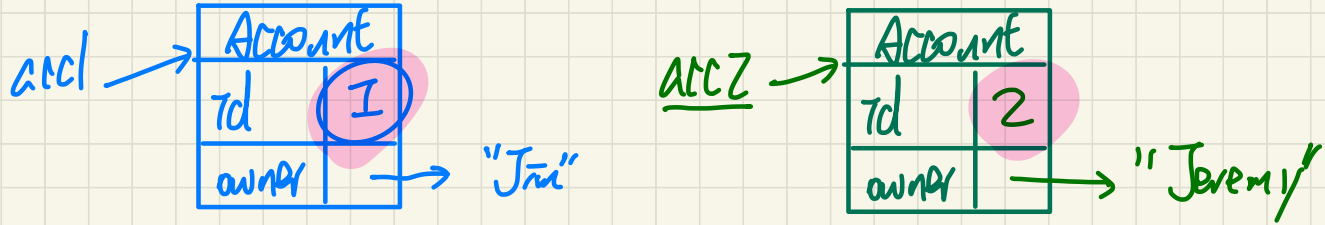
Managing Account IDs: Automatic


```
class Account {
  private static int globalCounter = 1;
  private int id; String owner;
  public Account(String owner) {
    this.id = globalCounter;
    globalCounter ++;
    this.owner = owner; } }
```

parameter list not including id. (not burden of user of Account anymore!)

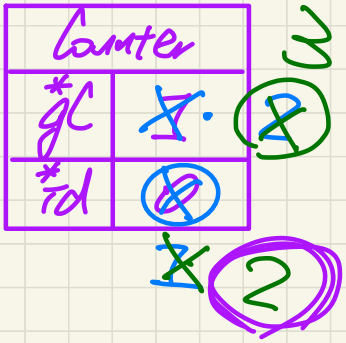


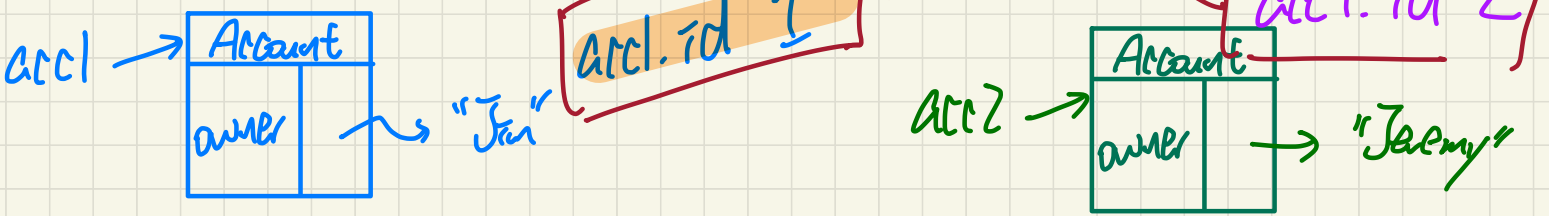

```
class AccountTester {
  Account acc1 = new Account("Jim");
  Account acc2 = new Account("Jeremy");
  System.out.println(acc1.getID() != acc2.getID()); }
```



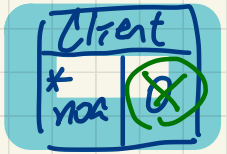
```
class Account {
  private static int globalCounter = 1;
  private int id; String owner;
  public Account(String owner) {
    this.id = globalCounter;
    globalCounter ++;
    this.owner = owner; } }
```

```
class AccountTester {
  Account acc1 = new Account("Jim");
  Account acc2 = new Account("Jeremy");
  System.out.println(acc1.getID() != acc2.getID()); }
```





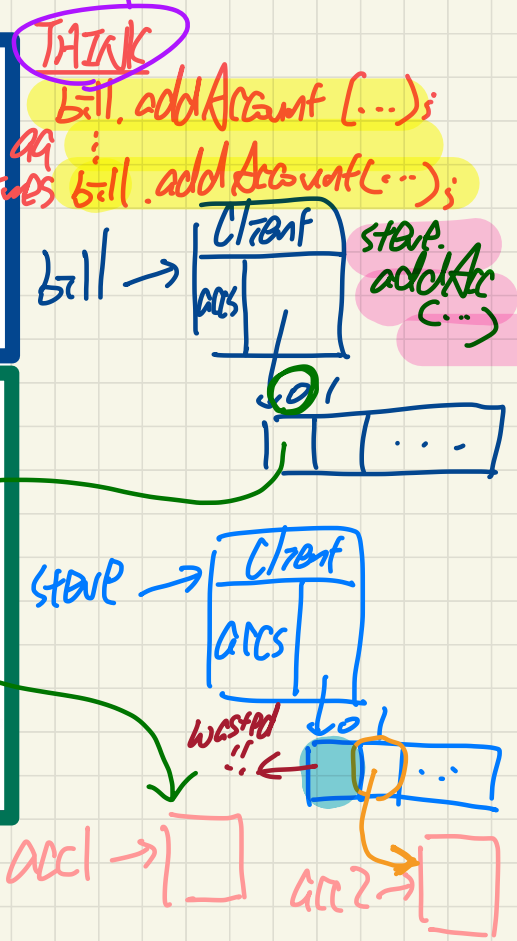
Misuse of Static Variables



x 2

```
public class Client {
  private Account[] accounts;
  private static int numberOfAccounts = 0;
  public void addAccount(Account acc) {
    accounts[this.numberOfAccounts] = acc;
    this.numberOfAccounts ++;
  } }
```

```
public class ClientTester {
  Client bill = new Client("Bill");
  Client steve = new Client("Steve");
  Account acc1 = new Account();
  Account acc2 = new Account();
  bill.addAccount(acc1);
  /*                                                    */
  steve.addAccount(acc2);
  /*                                                    */
}
```



Use of Static Variables: Common Error

Bank cibc = new Bank();
cibc branchName

Slides 80 - 82

```
1  public class Bank {
2      private string branchName;
3      public String getBrachName() { return this.branchName; }
4      private static int nextAccountNumber = 0;
5      public static String getInfo() {
6          nextAccountNumber++;
7          return this.branchName + nextAccountNumber;
8      }
9  }
```

should be replaced by some obj.

class name

Expected Usage: Bank.getInfo() [no context obj.]

Bank.branchName X ∵ branchName non-static

Fix 1

```
1  public class Bank {
2     private string branchName;
3     public String getBrachName() { return this.branchName; }
4     private static int nextAccountNumber = 0;
5     public static String getInfo() {
6        nextAccountNumber++;
7        return this.branchName + nextAccountNumber;
8     }
9  }
```

S = String "23" + "46"

S "2346"

Fix 2

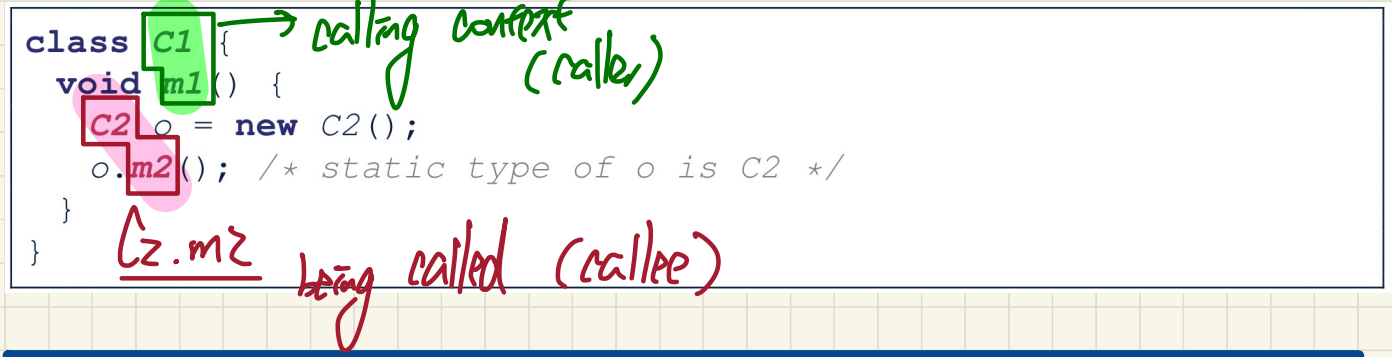
start

not ideal :(each Bank obj should have diff. branchName.

```
1  public class Bank {
2     private string branchName;
3     public String getBrachName() { return this.branchName; }
4     private static int nextAccountNumber = 0;
5     public static String getInfo() {
6        nextAccountNumber++;
7        return this.branchName + nextAccountNumber;
8     }
9  }
```

Caller vs. Callee

- caller is the client using the service provided by another method.
- callee is the supplier providing the service to another method.



Q: Can a method be a caller and a callee simultaneously?