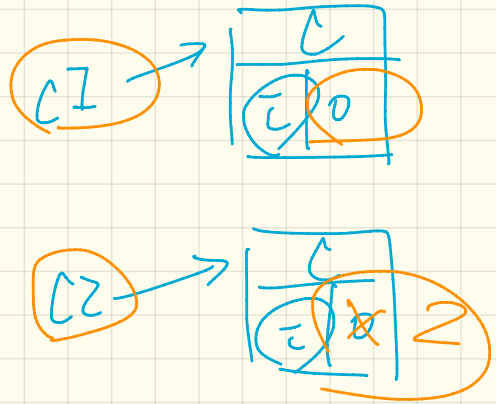
Wednesday March 20

Lecture 20



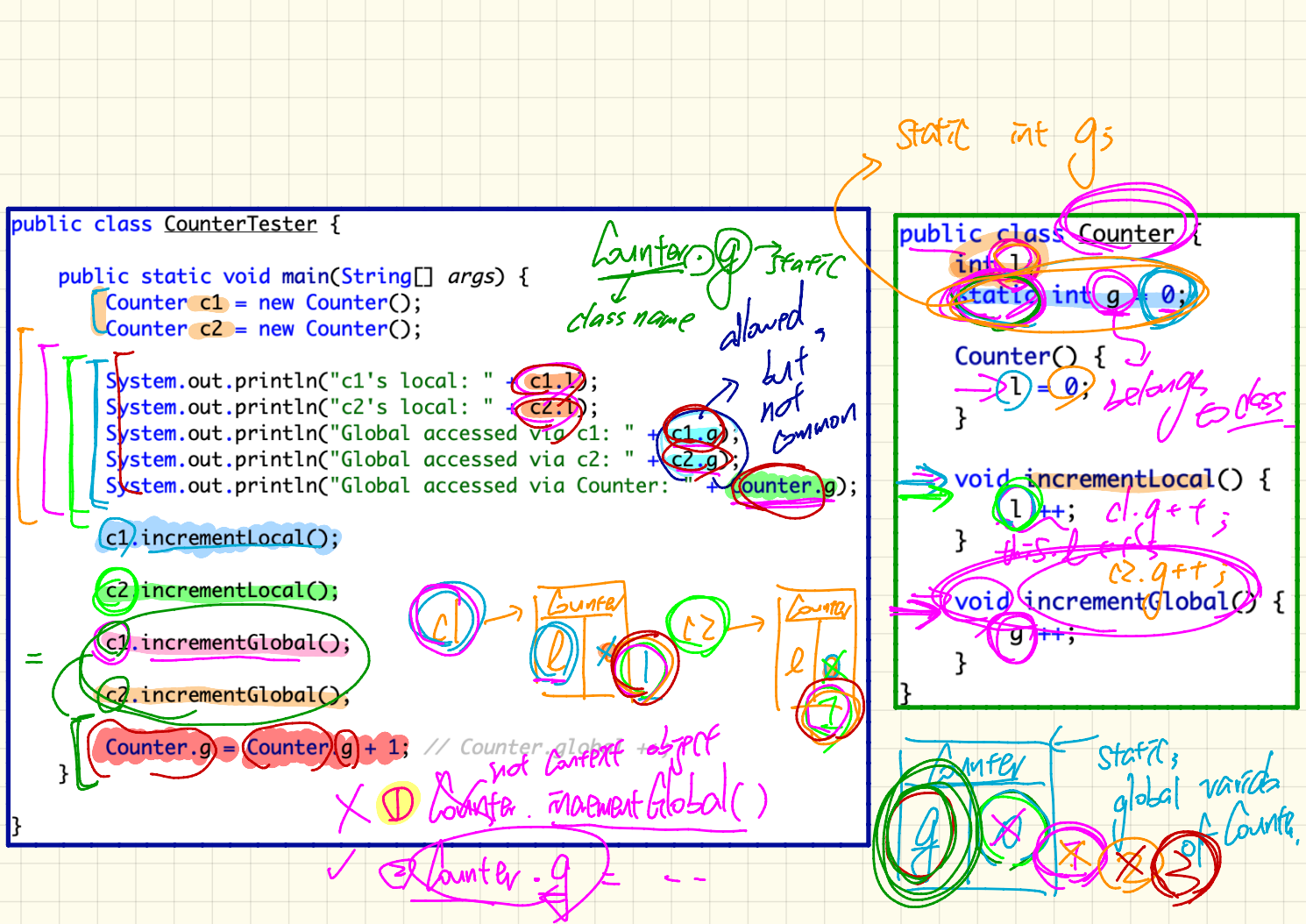

```
class C {
    int i;
    static int j;
}
```

non-static

c1.i

c2.i

all objects of type C share a single copy of j.

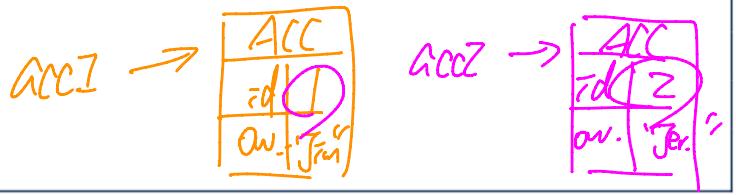

public class CounterTester {
public static void main(String[] args) {
Counter c1 = new Counter();
Counter c2 = new Counter();
System.out.println("c1's local: " + c1.l);
System.out.println("c2's local: " + c2.l);
System.out.println("Global accessed via c1: " + c1.g);
System.out.println("Global accessed 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
}
}
Counter.g
static
class name
allowed,
but
not
common
=
c1
Counter
l
g
1
c2
Counter
l
g
1
not context object
Counter.incrementGlobal()
Counter.g
static int g;
public class Counter {
int l;
static int g = 0;
Counter() {
l = 0;
}
belongs to class
void incrementLocal() {
l++;
c1.g++;
c2.g++;
}
void incrementGlobal() {
g++;
}
}
Counter
g
1
2
3
static;
global variable
Counter

Managing Account ID: Manually

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

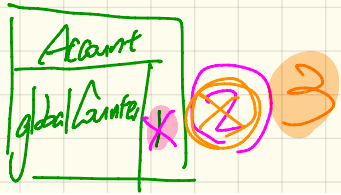
non-static

non-static




```
class AccountTester {
  Account acc1 = new Account(1, "Jim");
  Account acc2 = new Account(2, "Jeremy");
  System.out.println(acc1.id != acc2.id);
}
```

Managing Account ID: Automatically



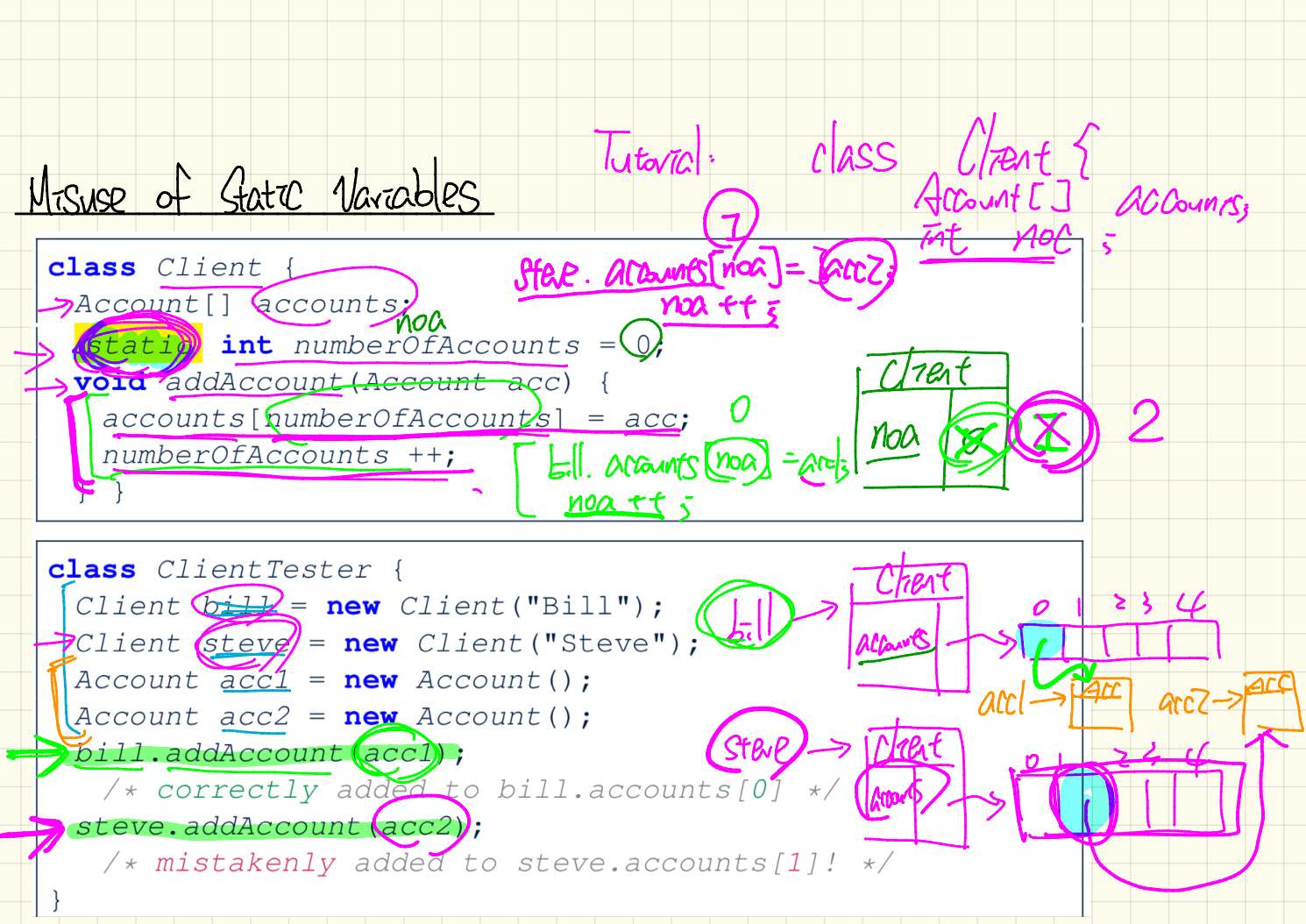

```
class Account {
  static int globalCounter = 1;
  int id; String owner;
  Account(String owner) {
    this.id = globalCounter; globalCounter ++;
    this.owner = owner; } }
```

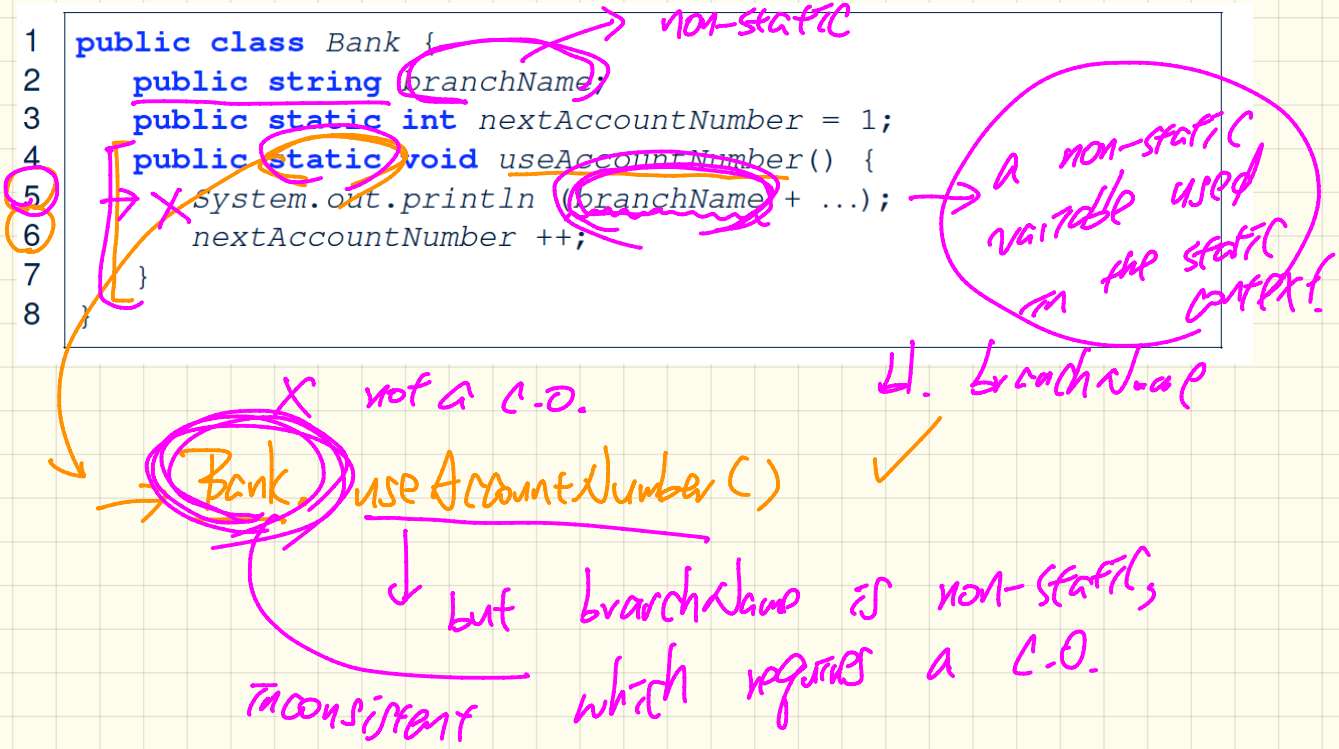

```
class AccountTester {
  Account acc1 = new Account("Jim");
  Account acc2 = new Account("Jeremy");
  System.out.println(acc1.id != acc2.id); }
```

Misuse of Static Variables

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

```
class ClientTester {
  Client bill = new Client("Bill");
  Client steve = new Client("Steve");
  Account acc1 = new Account();
  Account acc2 = new Account();
  bill.addAccount(acc1);
    /* correctly added to bill.accounts[0] */
  steve.addAccount(acc2);
    /* mistakenly added to steve.accounts[1]! */
}
```




1 public class Bank {
2 public string branchName;
3 public static int nextAccountNumber = 1;
4 public static void useAccountNumber() {
5 System.out.println (branchName + ...);
6 nextAccountNumber ++;
7 }
8 }
non-static
a non-static variable used in the static context
Bank.useAccountNumber()
not a C.O.
but branchName is non-static, which requires a C.O.
inconsistent