

Managing Account IDs: Automatic

Slides 76 - 77

```
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

Slides 78 - 79

```
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);  
    /* [REDACTED] */  
    steve.addAccount(acc2);  
    /* [REDACTED] */  
}
```

Use of Static Variables: Common Error

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 }
```

Example: Reference to this

Slide 59 - 61

```
public class Person {  
    private String name;  
    private Person spouse;  
    public Person(String name) {  
        this.name = name;  
    }  
    public void marry(Person other) {  
  
    }  
}
```

```
Person jim = new Person("Jim");
Person elsa = new Person("Elsa");
jim.marry(elsa);
```

Reference-Typed Return Values

Slide 53

```
public class Point {  
    /* A mutator modifying the context Point object */  
    public void moveUp(double i) {  
        this.y = this.y + i;  
    }  
  
    /* An accessor returning a new Point object */  
    public Point movedUpBy(double i) {  
        Point np = new Point(this.x, this.y);  
        np.moveUp(i);  
        return np;  
    }  
}
```

```
public class PointTester {  
    public static void main(String[] args) {  
        Point p1 = new Point(2.5, -3.6);  
        p1.moveUp(7.8);  
        Point p2 = p1.movedUpBy(6.4);  
        System.out.println(p1 == p2);  
    }  
}
```



Anonymous Objects

Slide 56 - 58

```
1 double square(double x) {  
2     double sqr = x * x;  
3     return sqr; }
```

```
1 double square(double x) {  
2     return x * x; }
```

```
1 Person getP(String n) {  
2     Person p = new Person(n);  
3     return p; }
```

```
1 Person getP(String n) {  
2     return new Person(n); }
```

```
class Member {  
    private Order[] orders;  
    private int noo;  
    /* constructor omitted */  
    public void addOrder(Order o) {  
        this.orders[this.noo] = o;  
        this.noo++;  
    }
```

```
    public void addOrder(String n, double p, double q) {
```

```
    }  
}
```

Exercise