

# CS 617 Object Oriented Systems

## Lecture 5

Classes, Classless World: Prototypes, Instance  
Variables, Class Variables, This/Self  
3:30-5:00 pm Thu, Jan 17

**Rushikesh K Joshi**

Department of Computer Science and Engineering  
Indian Institute of Technology Bombay

# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes
- 4 Instance Variables and Class Variables
- 5 Embedded Vs. Shared Implementations

# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes
- 4 Instance Variables and Class Variables
- 5 Embedded Vs. Shared Implementations

# Object

- State
- Id
- Behavior

# Interface

Member functions accessible on an Object

Syntactic Descriptions

Member function names, input parameters, their types, output result type, directions of parameters (in/out/inout)

# Class

Description of Structure of similar objects

Description of Behavior of Similar objects

Are Classes themselves Objects?

# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes
- 4 Instance Variables and Class Variables
- 5 Embedded Vs. Shared Implementations

# Objects without Classes

How would you create new objects?

How would you create the first object?

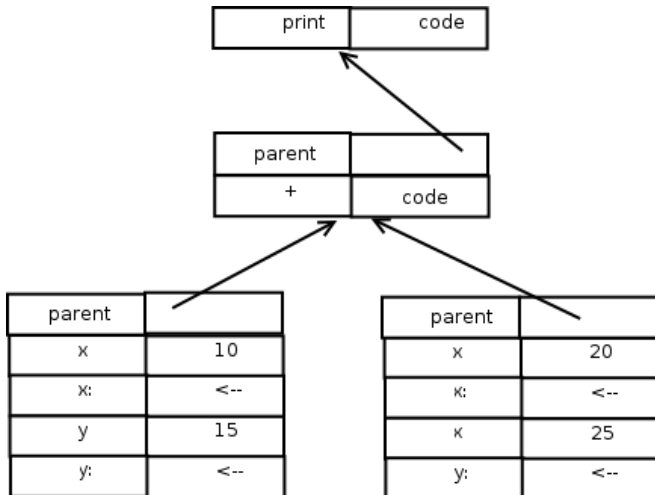
Examples?



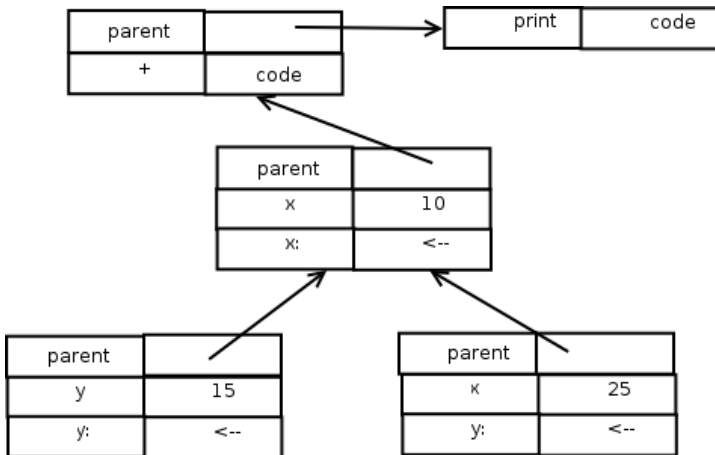
# Object Oriented Languages without Classes

- Prototype-based Programming
- Creation not by instantiation, but by cloning
- Objects can inherit from objects: shared properties
- Each object can be unique
- Example: Self

## A Snapshot of a Prototype-based System



# A Snapshot of Sharing of Variables between Objects



# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes**
- 4 Instance Variables and Class Variables
- 5 Embedded Vs. Shared Implementations

# Class: Describes a class of objects I

```
interface IAccount {
    int balance();
    Boolean deposit(int amount);
    Boolean withdraw (int amount);
}
class Account implements IAccount {
    private int bal;
    public Account () {bal=0;}
    public int balance() {return bal;}
    public Boolean deposit (int amount) {
        bal = bal+amount; return true;}
    public Boolean withdraw (int amount) {
        if (bal<amount) return false;
```

## Class: Describes a class of objects II

```
        if (bal >= amount) bal = bal - amount; return true;}  
public static void main (String args[]) {  
    IAccount a1 = new Account();  
    a1.deposit(200);  
    a1.withdraw(15);  
    System.out.println(a1.balance());  
    IAccount a2 = new Account();  
    a2.deposit(200);  
    a2.withdraw(215);  
    System.out.println(a2.balance());  
}  
}
```

# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes
- 4 Instance Variables and Class Variables**
- 5 Embedded Vs. Shared Implementations

## Instance Variables vs. Class Variables

Instance Variables: A copy per object

*int bal* in above example.

Class Variables: A copy per class of objects

*in noOfAccounts in the example on the next slide*



# Class: Describes a class of objects I

```
interface IAccount {
    int balance();
    int accNo();
    Boolean deposit(int amount);
    Boolean withdraw (int amount);
}
```

```
class Account implements IAccount {

    private static int noOfAccounts=0;
    public static int getAccID() {
        return noOfAccounts;}
}
```

## Class: Describes a class of objects II

```
private int acc_no;
private int bal;

public Account () {
    acc_no=getAccID(); bal=0; noOfAccounts++;}

public int balance() {return bal;}

public int accNo() {return acc_no;}

public Boolean deposit (int amount) { ..}
```

## Class: Describes a class of objects III

```
public Boolean withdraw (int amount) { ..}
```

```
public static void main (String args[]) {  
    IAccount a1 = new Account();  
    a1.deposit(200);  
    a1.withdraw(15);  
    System.out.println(a1.balance()+" in acc no." +  
                        a1.accNo());  
}
```

```
IAccount a2 = new Account();  
a2.deposit(200);  
a2.withdraw(215);
```

## Class: Describes a class of objects IV

```
System.out.println(a2.balance()+" in acc no." +  
                    a2.accNo());
```

```
IAccount a3 = new Account();
```

```
System.out.println(a3.balance()+" in acc no." +  
                    a3.accNo());
```

```
}  
}
```

# Class members Vs. Instance Members

Modeling Meta-level state and behavior

Modeling Metadata

Instance Members and Class Members

Accessibility?

# Outline

- 1 Objects, Interfaces and Classes
- 2 The Classless World
- 3 Classes
- 4 Instance Variables and Class Variables
- 5 Embedded Vs. Shared Implementations

## How many copies of Member Functions? I

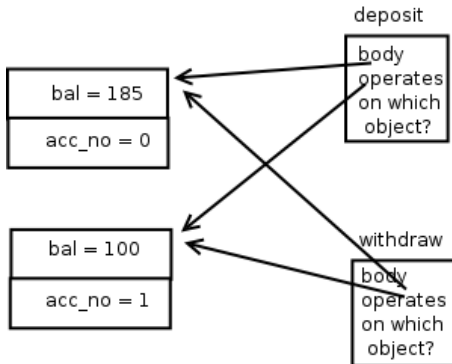
How many copies of instance variables are necessary?

How many copies of class variables are necessary?

How many copies of member functions are necessary?

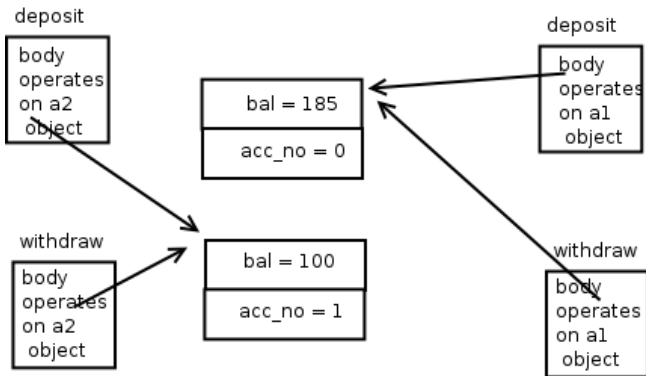
Reasons?

## How many copies of Member Functions? II



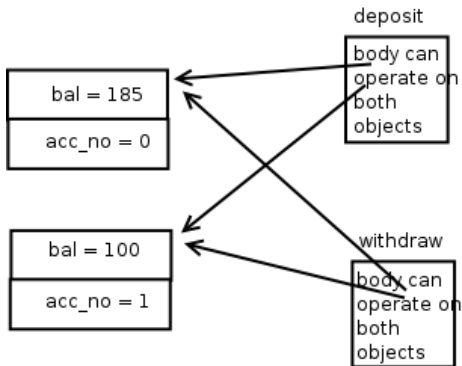


# Embedded Implementation



Fast: no indirections for variable accesses, More Space

## Shared Implementation



Slower: one indirection for every variable access, Less Space

## This (or Self)

The value of **this** comes from member function implementation sharing

Same **this** handle can be used for self references.

An implicit argument to every member function

What can be the user level applications of *this*?