1.00/1.001
Introduction to Computers and Engineering Problem Solving

# Recitation 5
# Recursion, Inheritance

March 12th, 13th 2011

## Recursion

Recursion illustration removed due to copyright restrictions.

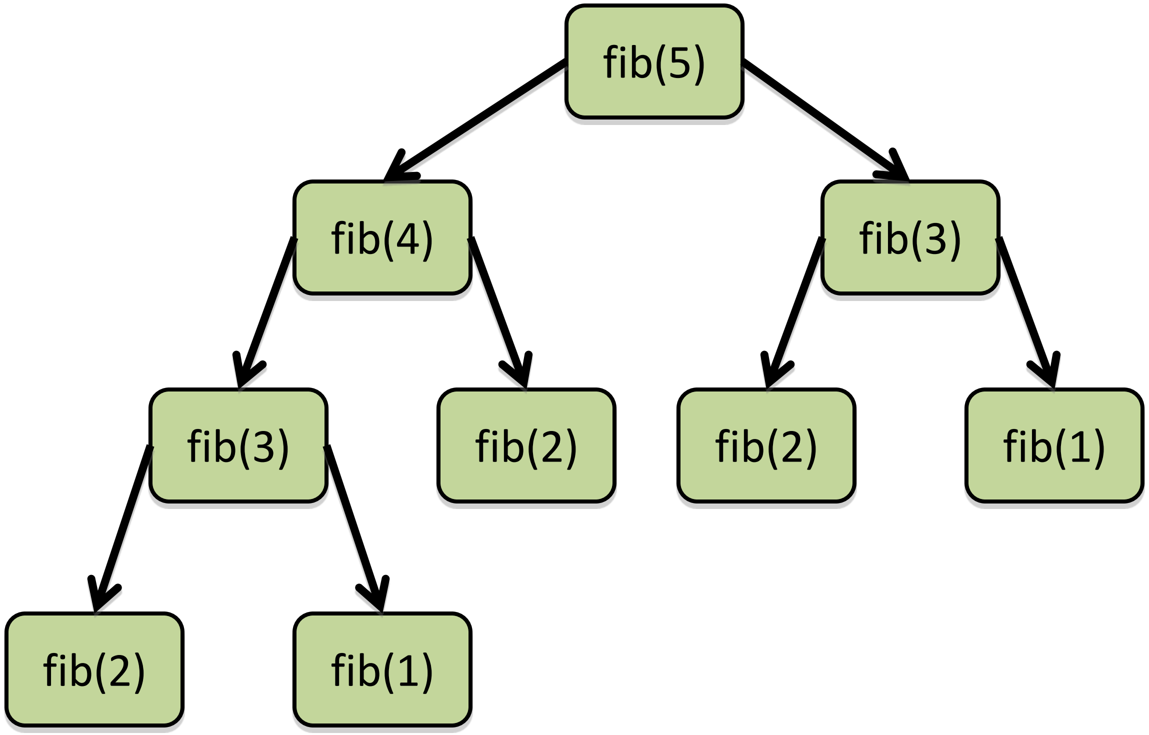
## Designing Recursive Methods

1. Define the base case
2. Divide big problem into smaller problems
3. Recursively solve the smaller problems
4. Combine the solutions to the smaller problems

Be aware that a recursive method may not be the most efficient solution

## Example

- Fibonacci Sequence: $F_n = F_{n-1} + F_{n-2}$ $F_0 = 0,\ F_1 = 1$
- Formula: fib(n)=fib(n-1)+fib(n-2)



# Fibonacci Sequence

```
public class fib{

  public static int fib( int n) {
        if( n <= 1 )
              return n;
        else
              return fib(n-1) + fib(n-2);

  }
}
```

Base case

Recursive case

Smaller Problems

## Exercise 1

- Design a recursive method to calculate the factorial (!) of a number

- $n! = n \times (n-1) \times (n-2) \times \ldots 3 \times 2 \times 1$

## Exercise 2

- You are given a positive integer n and you need to recursively print out all numbers from n to 1, in descending order.

- Example: given n = 3, your program will print

```
3
2
1
```

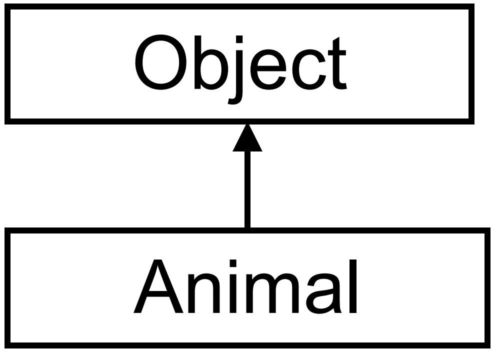
# Understand Inheritance

Just as you inherited qualities from your parents, a class can *inherit* the data members and methods of another class.

Here is class Animal:

```
public class Object {
```

```
  public class Animal {
      private String foodtype;
      public Animal(String f){
            foodtype = f;}
      public void feed(){
            //not shown
      }}
```



*All* classes in Java automatically inherit from class "`Object`".

"Object" is the *parent* or **`super`** class of Animal

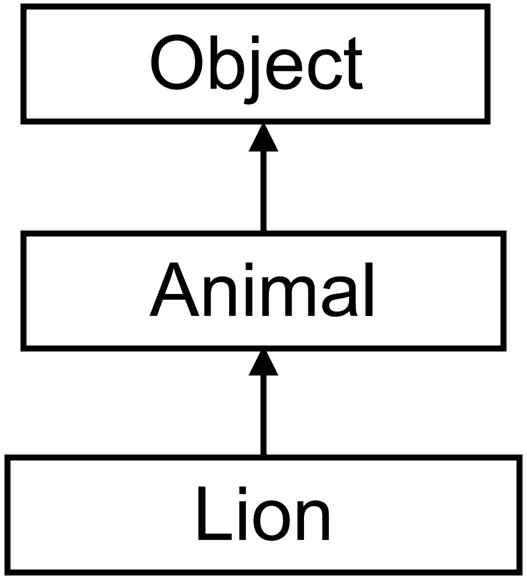
`Animal` *inherits* from / *is a subclass of* / or **`extends`** `Object`

# Inheritance Example

Here class **`Lion extends Animal`**

```
public class Animal {
    private String foodtype;
    public Animal(String f){
            foodtype = f;}
    public void feed(){
            //not shown
```

```
public class Lion extends Animal{
      private boolean isAfrican;
      public Lion(boolean fromAf){
            super("carnivorous");
            isAfrican = fromAf;
      }
   }
```



`super` refers to the parent class.

What is the super class here?

What does this line do?

To inherit from another class use keyword `extends`

## Inheritance Question

- Which of the following declarations is NOT allowed and why?

```
Animal a1 = new Animal(“herbivore”);
Animal a2 = new Lion(true);
Lion   a3 = new Lion(false);
Lion   a4 = new Animal(“carnivore”);
Animal a5 = new Lion(“carnivore”);
Object o = new Lion(true);
```

# Method Overriding

```
public class Lion extends Animal {
  private boolean isAfrican;
  /*constructor & hidden code HERE*/

  public void feed() {
    System.out.println(“feed() method of Lion");
  }
}
```

```
public class Cow extends Animal {
  private String breed;
  /*constructor & hidden code HERE*/

  @Override //This is optional
  public void feed() {
     super.feed();
     System.out.println(“feed() method of Cow");
  }
}
```

**What does super do here?**

## Method Overriding (cont.)

```
public class Animal {
   private String foodType;

   /*constructor hidden*/

   public void feed() {
      System.out.println(“feed() method of Animal");
   }

   public static void main(String [] args){
      Animal [] a = {new Lion(true), new Cow(“dairy”)};
      a[0].feed();
      a[1].feed();
   }
}
```

What is the output from the main() method ?