IT2205: Programming I

Section 5

Object Orientation (13 hrs)

Objects and Classes

Definition of a(n) Object/Class

- **What is an Object :**
  - A ‘thing’ may have a physical presence such as a ‘table’ ,‘chair’ or an abstract concept such as ‘a job’ .
  - An object is an abstract representation of a ‘thing’ in the real world.
  - We simulate a problem domain in the real-world through objects.
  - An object has a unique identity, attributes (What it knows or data about it), and behavior (What it can do).

Object cont....

An Employee object (say employee1) will have the following attributes (what it knows):

name, age, salary

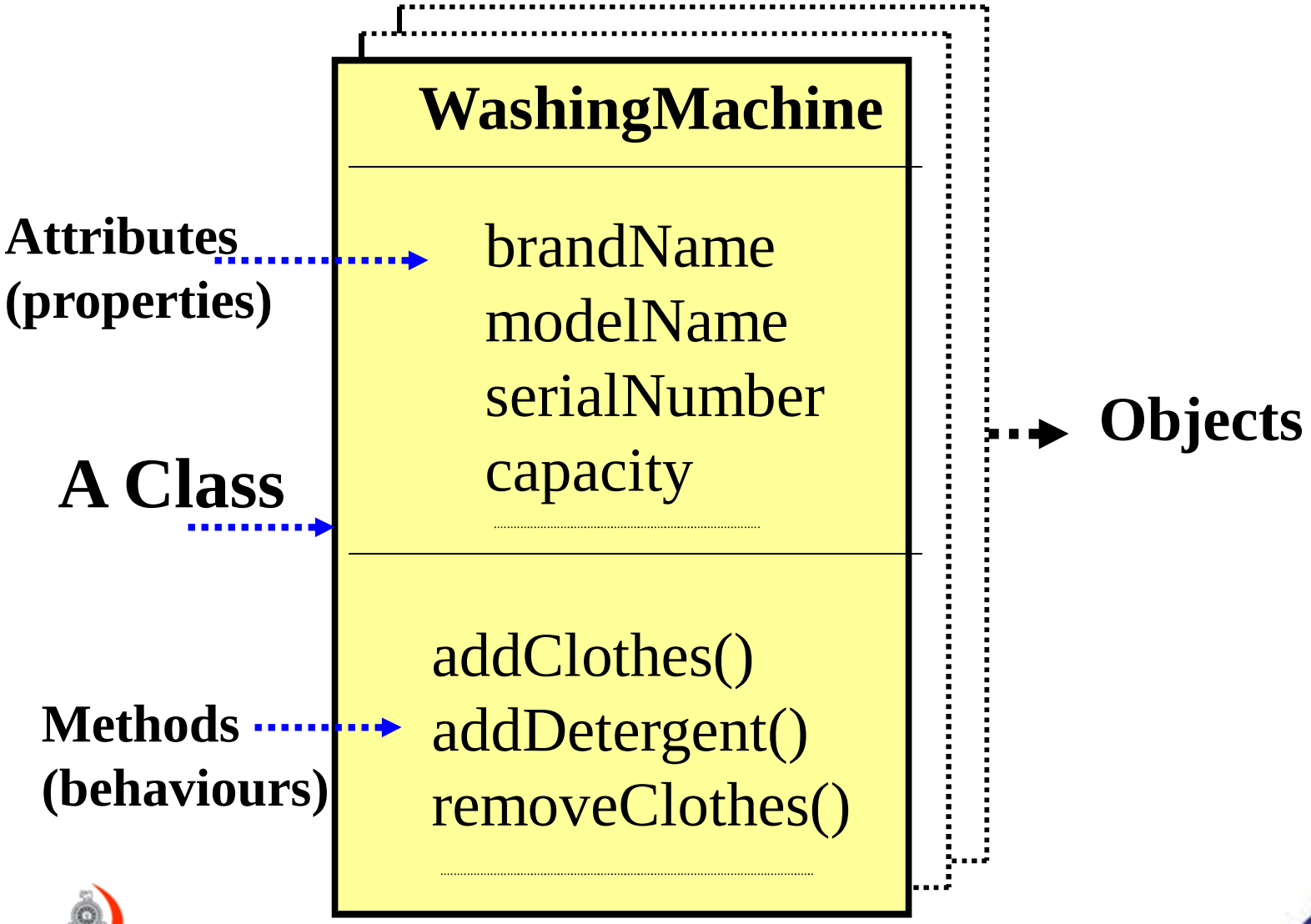
It will have the following behaviour (what it can do):

set salary, get salary, set name, set age

Object cont....

- **What is a Class?**
  - Objects are the representation of things in the real world
  - *Classes* are templates for objects, ie. They contain all the common properties and behaviour of a group of objects.
  - All objects with the same characteristics and behaviours belongs to the same class.

Attributes and Methods



Java Application

- A *Java apllication* is a collection of one or more Java classes.
- At least one of the classes is *public* and contains a method named *main()* except in case of applets..

Objects and Classes contd…

```
public class Employee {
    private String name;
    private int age;
    private float salary;
```

Attributes

```
public void setName(String tName) {
        Name = tName;
}

public void setSalary (float tSalary) {
        salary = tSalary; }
public float getSalary () {
        return salary; }
```

Methods

An example class Declaration

Creating Objects

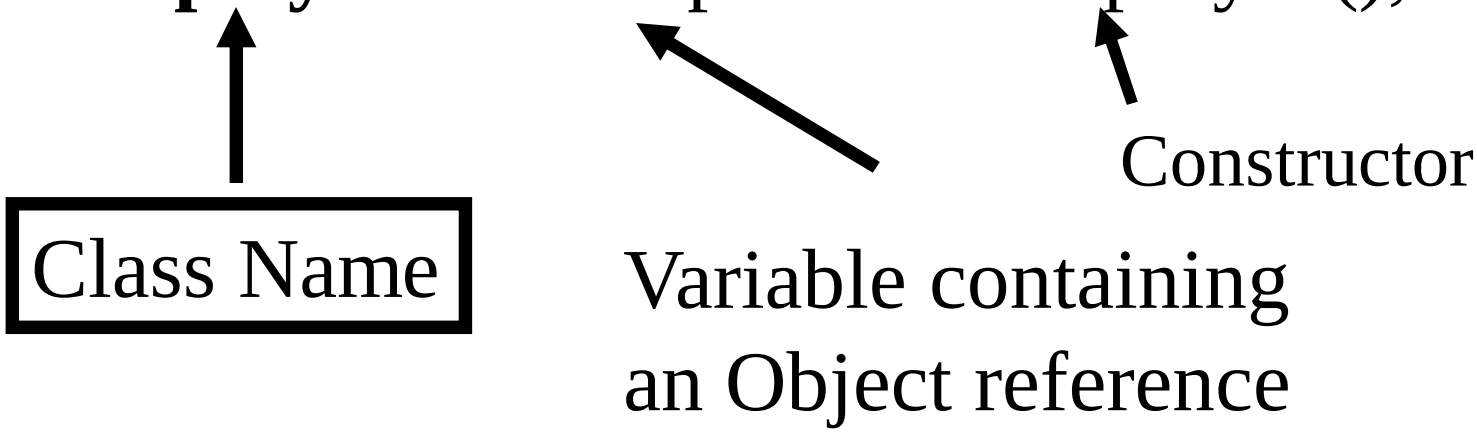
- Objects are created by instantiating *classes.*
- To use a class in a program, you must first create an instance of it.

Creating Objects

- Objects of a class can be created using the ***new*** operator.

Example

Employee newEmp = ***new*** Employee();



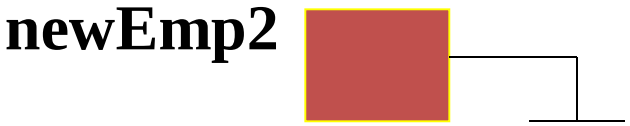
Creating Objects cont..

Object References

following declaration will create an Object reference

Employee newEmp2 ;

The newEmp2 variable will have value ***null***

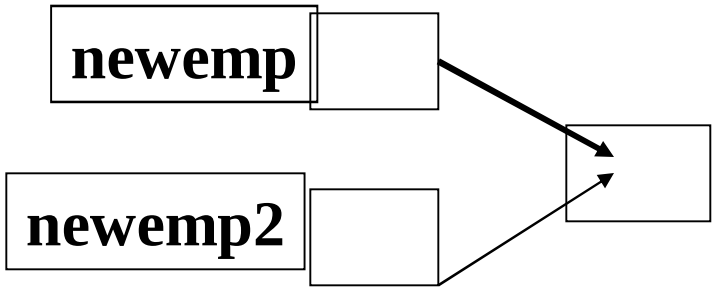


Creating Objects cont...

Object References

You can create multiple References to the same object

```
Employee newEmp;
newEmp = new Employee();
newEmp2 = newEmp;
```



Destroying Objects

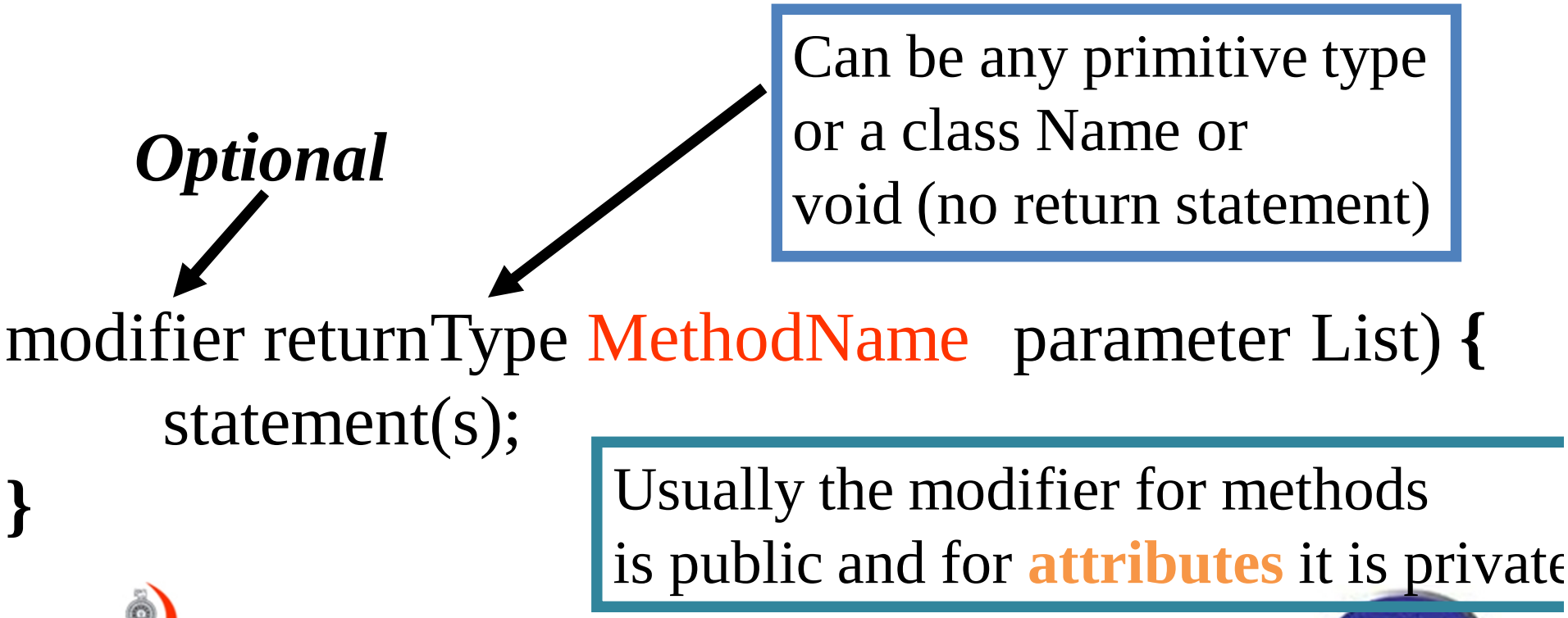
- Java's garbage collection is automatic
- There is no great need for an explicit ***destructor.***

Destroying Objects

- ***finalize*** method plays the role of a destructor.
- This method is useful to ***disconnect network connections*** as well as ***close any open databases***

Objects and Classes contd...

A basic method definition is as follows:



Objects and Classes cont...

- The **return Type, Method Name**, and the **parameter list** defines the ***Signature*** of the method
- it is possible to define two ore more methods with the same name within the same class (Method Overloading) with different signatures

```
public void CreatePoint( )
public void CreatePoint(int x , y)
```

Objects and Classes cont...

- if the return type is an array object,
  - the square brackets can go after the return type or after the parameter list

```
int [ ] SortedList (int List [ ]);
OR
int SortedList (int List [ ]) [];
```

Objects and Classes cont...

Accessing methods

a ***instance method*** can be accessed using the dot(.) operator as shown below.

```
ObjectName.method();
```

Objects and Classes cont...

Accessing methods cont...

A ***class method*** can be accessed by using the class name followed by a period and then the method name

```
Integer.parseInt("25");
```

Objects and Classes cont...

Passing parameters

- all parameters (except ***Object types*** such as instances of classes and Arrays etc.) are **passed by value.**
- i.e. any modifications done within the method does not affect the original variable.

Objects and Classes cont...

- **Passing parameters cont...**
  - if you require to modify the original variables, and need to pass those to a method
    - Declare those as instance variables in a class and pass an object of that class to the method

Objects and Classes cont...

Constructor Methods

- These methods are used to ***initialize objects***
- they have the ***same name as the class*** and have ***no return type***
- these methods are called automatically when the ***new*** operator is used to allocate memory for an object.

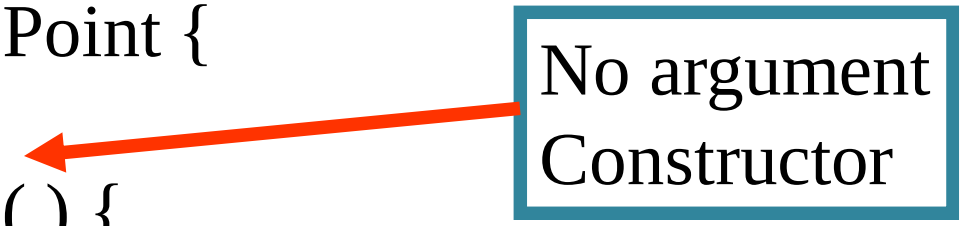
Objects and Classes cont...

- **Overloading Constructors**
  - A class can have ***multiple Constructors*** (Overloaded Constructors)
  - All carries the ***same name***
  - They have either ***different number of arguments*** or ***different types of arguments***

Objects and Classes cont...

Example for Constructors

```
public class twoDPoint {
        float x,y;
        twoDPoint ( ) {
                x= y = 0;
        }
        }
```



Objects and Classes cont...

Example for Constructors

```
public class twoDPoint {
        float x,y;
        twoDPoint ( ) {
                x= y = 0;
        }
        twoDPoint ( float  a , float b) {
                x=a;
                y=b;
        }
}
```

No argument Constructor

Overloaded constructor

Objects and Classes cont...

this keyword

- ***this*** keyword is used to refer to the current object.

Objects and Classes cont...

this keyword

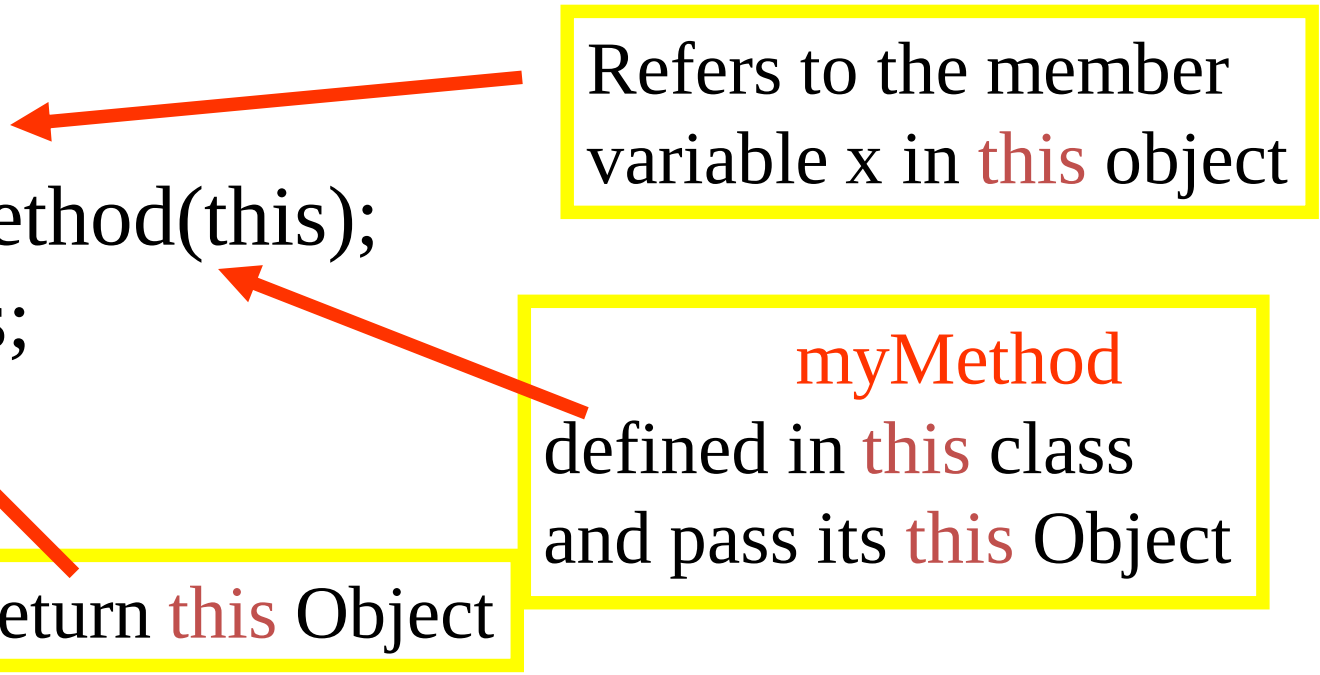
It can be used to :

- refer to the current objects member variables,
- refer to current objects methods
- pass a reference to the current object to a method
- return a reference current object

Objects and Classes cont...

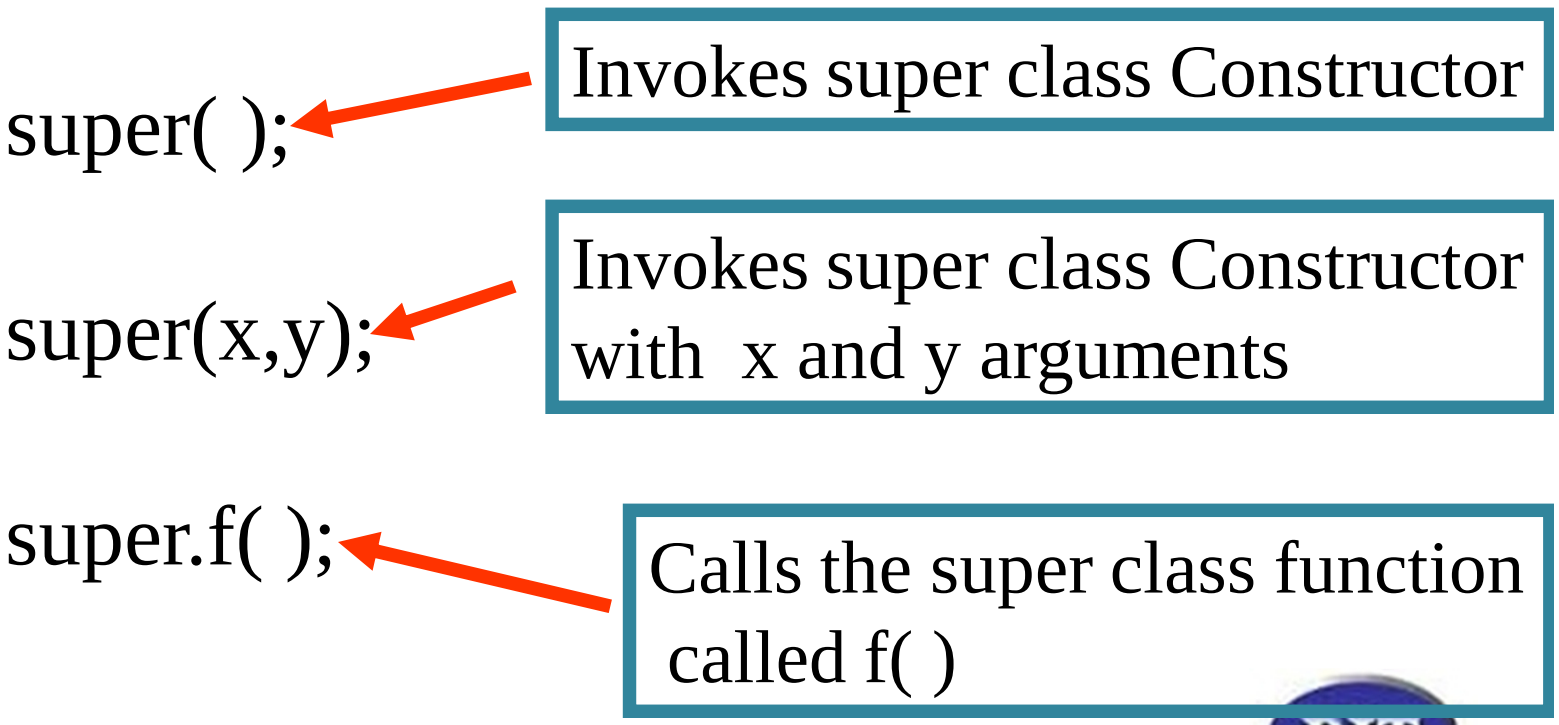
- ***this* keyword**

```
t=this.x;
this.myMethod(this);
return this;
```



Objects and Classes cont...

- ***super* keyword**
  - This keyword is used to refer to the Super or parent class



Objects and Classes cont...

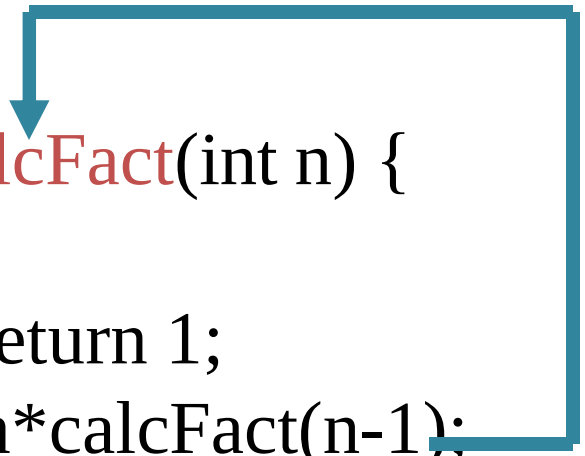
Recursion

- Methods that invoke themselves are known as Recursive methods.

Objects and Classes cont...

- **Recursion Example**

```
class Factorial {
        static long calcFact(int n) {
                if n < 2
                        return 1;
                return n*calcFact(n-1);
        {
}
```



Recursive call

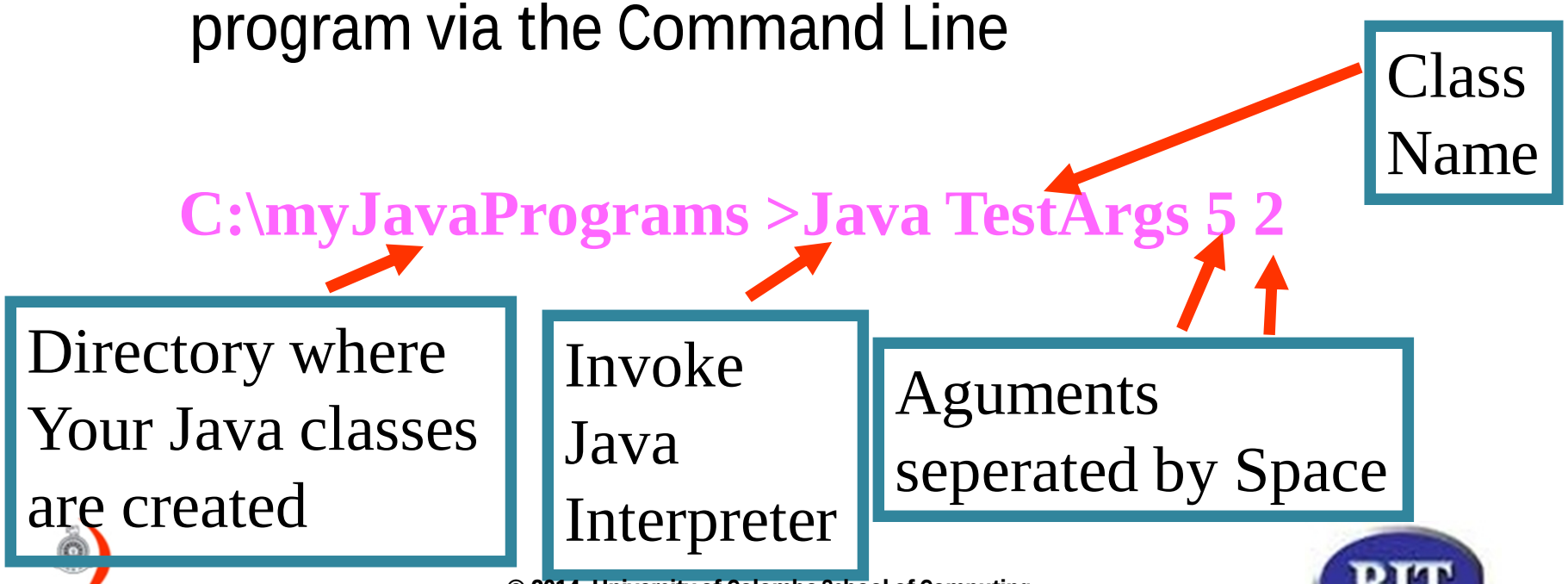
Objects and Classes cont...

To test the Factorial Class

```
Class TestFactorial {
        public static void main (String args[ ]) {
                System.out.print("Factorial of 5 is");
                System.out.println( Factorial.calcFact(5));
        }
}
```

Objects and Classes cont...

- **Using Command Line arguments**
  - in Windows environment, you can pass arguments to the main method of a Java program via the Command Line



Example for handling Command Line Arguments

```
class EchoArgs {
        public static void main(String args[]) {
                for (int i = 0; i < args.length; i++) {
                System.out.println("Argument " +
                                i + " " +args[i]);
                 }
        }
}
```

The End of Section