# Object-Oriented Programming (CS F213)

## Module I: Object-Oriented and Java Basics

BITS Pilani

**Dr. Pankaj Vyas**
Department of Computer Science, BITS-Pilani, Pilani Campus

# Today's Agenda

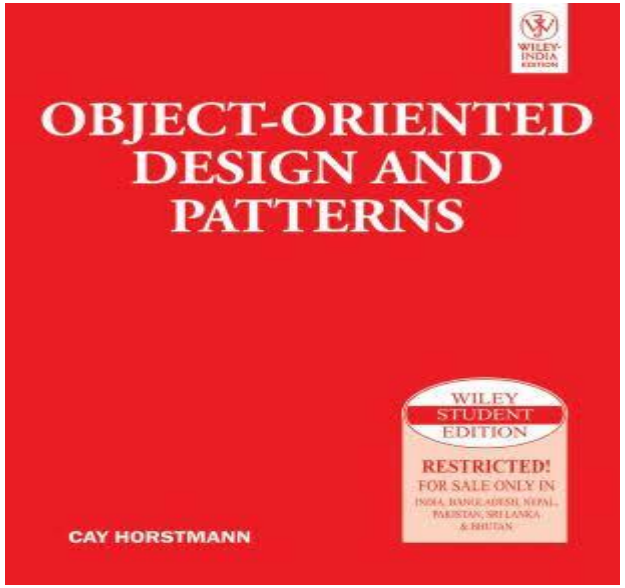
innovate achieve lead

- Course Overview
- What is an Object ?
- Graphical View of an Object
- Object Examples
- What is a Class ?
- Object vs Class ?
- Class Examples

# Course Overview

- Text Books

1. Java: The Complete Reference, Herbert Schildt, McGraw Hill Education and Oracle Press, Ninth Edition, 2014.
2. Object Oriented Design & Patterns, Cay Horstmann, John Wiley & Sons, 2004 [Must Procure]

# Course Overview ..

innovate achieve lead

## Evaluation Components

| Component | Duration | Date & Time | %Weight | Nature |
|---|---|---|---|---|
| Mid Semester Test | 90 Minutes | Monday , 03/10/2016<br>Time:11:00 AM – 12:30 PM<br>[Syllabus: Lecture 1 – 21] | 25%<br>[75 Marks] | CB |
| Lab Attendance | - | - | 5%<br>[15 Marks] | - |
| Online Test | 120 Minutes | 13th Nov, 2016 (Sunday)<br>Time: 3:00 PM – 6:00 PM<br>[Syllabus: Multithreading] | 30%<br>[90 Marks] | OB |
| Comprehensive | 180 Minutes | Thursday, 1st December, 2016 | 40%<br>[120 Marks] | CB-OB |

# Course Overview ..

innovate achieve lead

## Object-Oriented Lab Sessions

| Lab # | Topics to be Covered |
|---|---|
| 1 | Moving from C to Java [Java Basics] |
| 2 | Class Design Basics I |
| 3 | Class Design Basics II |
| 4 | Packages, Inheritance and Polymorphism |
| 5 | Arrays and Strings |
| 6 | Interfaces, Inner classes and anonymous inner classes |
| 7 | Exception Handling and Collections |
| 8 | JAVA GUI and Event Handling |
| 9 | File Handling |
| 10 | Multi-Threading-1 |
| 11 | Class Design Lab |
| 12 | Design Pattern Lab |

- There will be 12 Lab Sessions.
- Each Lab Carries 1.5 Marks. [Total : 15 Marks (10 out of 12)]

# Course Overview .. Handout

innovate achieve lead

| Module I : Object-Oriented and Java Basics | |
|---|---|
| **Lecture 1:**<br>**Object-Oriented Basics** | • Object and Class Basics<br>• Basic Pillars of Object-Oriented Programming (Abstraction, Encapsulation, Inheritance and Polymorphism] |
| **Lecture 2-3:**<br>**Java Programming Syntax** | • Java Program Structure, Compiling and Executing a Simple Java Application<br>• Types of Variables in Java<br>• Primitive Types in Java<br>• Type Promotion and Type Casting |
| **Lecture 4-5 :**<br>**Defining Classes and Object Creation** | • Defining Classes and Access Modifiers, Creating Objects, Role of Constructors<br>• Accessing Instance Fields and Methods<br>• Local Variables vs Instance Fields , Mutable and Immutable Objects<br>• Command-Line Arguments, Reading Input from console Using Scanner class |
| **Lecture 6:**<br>**Use of static final keywords in Java**<br>**Method Overloading** | • Use of static and final keywords in Java<br>• Method Overloading |
| **Lecture 7:**<br>**Objects as Parameters** | • Objects as Parameters to Methods and Object class in Java |

# Course Overview .. Handout

innovate achieve lead

| | |
|---|---|
| **Module II : Arrays and String in Java** | |
| Lecture 8-9 : Arrays in Java | • Implementing 1-D and 2-D Arrays in Java, Role of Arrays class<br>• Implementing Dynamic Arrays Using Vector class |
| Lecture 10-11: Strings in Java | • String class, Important String Methods<br>• StringBuffer and StringTokennizer class in Java |
| **Module III : Polymorphism and Inheritance in Java** | |
| Lecture 12-13 : Inheritance in Java | • Extending classes and Role of super keyword<br>• Method Overriding [Super Type vs Sub-Type Relationships] |
| Lecture 14-15: Abstract Classes, Abstract Methods and Interfaces | • Abstract Methods and Classes<br>• Interfaces in Java [ class vs interface]<br>• Comparable and Comparator Interfaces in Java<br>• Nested and Inner Classes |
| Lecture 16: Generic Programming | • Generic Form of a Class<br>• Generic Interfaces and Bounded Types |
| **Module IV: Exception Handling Mechanism** | |
| Lecture 17-18 : Exceptions in Java | • Exception Basics and Types<br>• Catching Exceptions<br>• Writing Your Own Exceptions |

# Course Overview .. Handout

innovate achieve lead

| Module V: Collections Framework of Java | |
|---|---|
| Lecture 19-21 : Collections in Java | • Introduction to Collection Framework in Java, Important Collection Interfaces and Their Methods<br>• ArrayList and LinkedList Classes in Java<br>• Iterators and ListIterators<br>• Wrapper classes and Autoboxing |
| **Module VI: Multithreaded Programming in Java** | |
| Lecture 22-24: Multithreading | • Multithreading vs Multitasking<br>• Thread Class in Java and its Important Methods<br>• Creating Your own Threads and Runnable Interface<br>• Thread Synchronization, Inter Thread Communication<br>• Suspending and Resuming Threads |
| **Module VII: GUI Programming** | |
| Lecture 25-27: GUI Programming with Swing | • Introduction to swing package<br>• Containers and Components and Layouts and LayoutManager Interface<br>• JLabel class, JTextField class<br>• Swing Buttons, JButton, JToggleButton<br>• Check Boxes, Radio Buttons<br>• JScrollPane , JMenu, JMenuBar and JMenuItem<br>• Designing Frames and Adding Components, Timer Class in Java |

# Course Overview .. Handout


| Module VIII: Event Handling in Java | |
|---|---|
| **Lecture 28-30: Event Handling** | • Delegation Event Model<br>• Event Classes, Listener Interfaces<br>➢ ActionEvent and AdjustmentEvent Classes<br>➢ ComponentEvent and ContainerEvent Classes<br>➢ FocusEvent and InputEvent Classes<br>➢ MouseEvent and ItemEvent Classes<br>• Listener Interfaces<br>➢ ActionListener and AdjustmentListener Interfaces<br>➢ ComponentListener and ContainerListener Interfaces<br>➢ FocusListener and ItemListener Interfaces<br>➢ MouseListener and MouseMotionListener |

# Course Overview .. Handout

innovate achieve lead

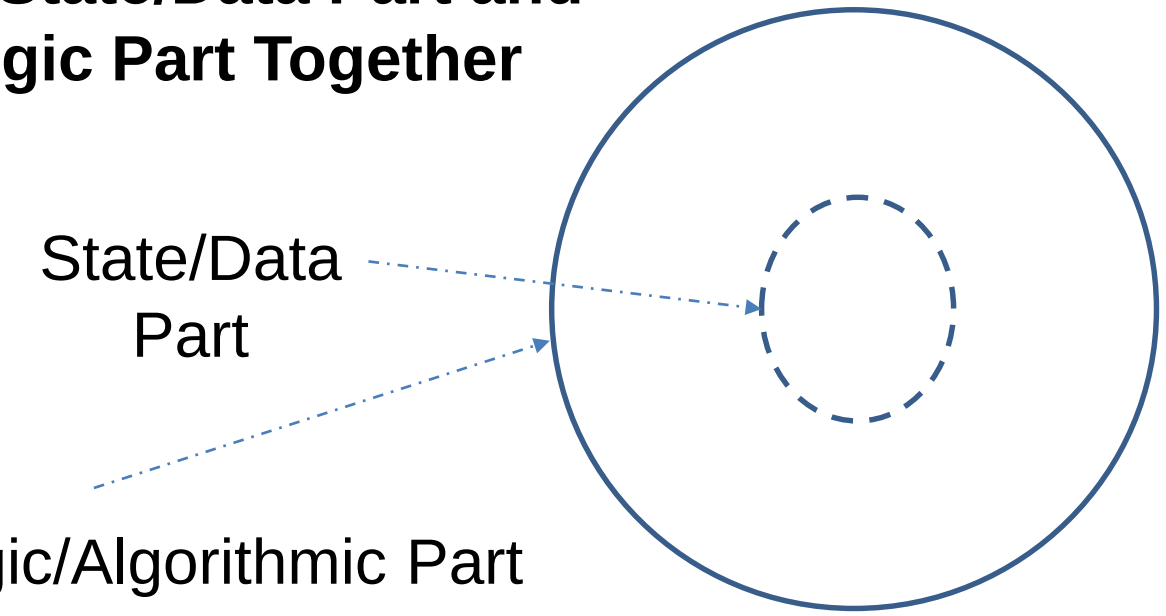
| Module IX: Object-Oriented Analysis and Design | |
|---|---|
| Lecture 31-33 : Object-Oriented Analysis | • Object Relationships and their representation in UML<br>• What are Use-Case Models and Use-Case Realization Templates<br>• UML Activity Charts<br>• Identifying Classes Using Noun-Phrase Analysis |
| Lecture 34-35: Object-Oriented Design | • Goals of Object-Oriented Design Phase<br>➢ Identifying Attributes and Methods of Each class<br>➢ Class Diagram, Sequence Diagrams, State Diagrams |
| **Module X: Object-Oriented Patterns** | |
| Lecture 36-40: Object-Oriented Design Patterns | • Design Pattern Basics<br>➢ Creational Patterns (Singleton Pattern , Factory Pattern , Factory Method Pattern)<br>➢ Structural Patterns (Adapter Pattern, Composite Pattern, Decorator Patterns, Proxy Pattern)<br>➢ Behavioral Patterns (Iterator Pattern, Chain of Responsibility, Strategy Pattern, Proxy Pattern, Visitor Pattern, Command Pattern ) |

# What is Object ?

- Object Means Combination of Data (Attributes) and Logic (Algorithm, Behavior, Functions, Operations) of some real world entity. For example Student, Box, Account, Time
- Every real-world object has two characteristics :
  - Data-Part/State [Also known as attributes or properties]
  - Behavior [Also known as operations / Algorithmic / Logic Part]
- Software Object is conceptually similar to a every real-world object.

# Object: Graphical View

**Object Keeps State/Data Part and Behavior/Logic Part Together**




**A Software Object**

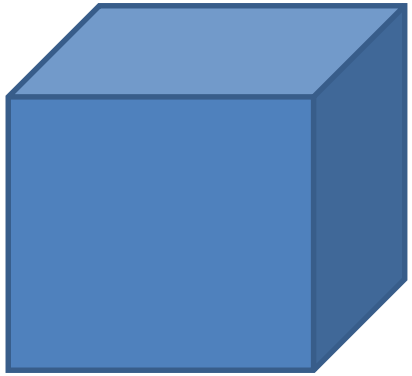
# Object Examples

- **Box Object :**
  - State/Data Part : length, width, height, Color [Attributes/Instance Fields]
  - Behavior Part: computing area, computing volume [Operations, Methods]
- **Dog Object :**
  - State/Data Part : name, breed, color [Attributes / Instance Fields]
  - Behavior Part : barking, fetching, wagging [Operations, Methods]
- **Account Object :**
  - State/Data Part : account number, account holder name, balance, type of account [Attributes/ Instance Fields]
  - withdrawing an amount, depositing an amount, checking balance of a account [ Behavior, Operations, Methods]

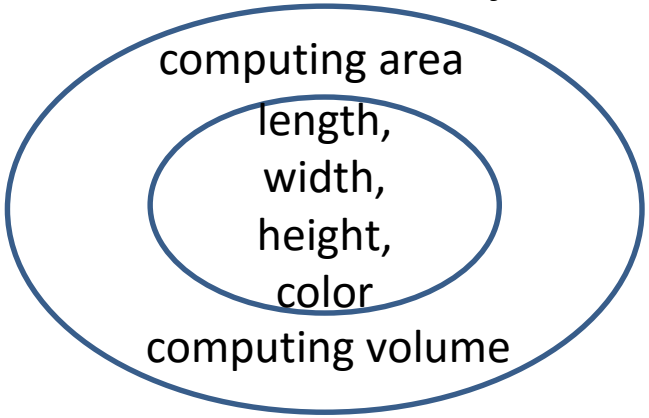
# Object Examples



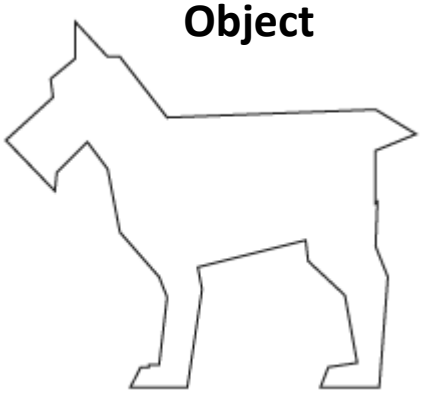
**Box as a Real World Object**



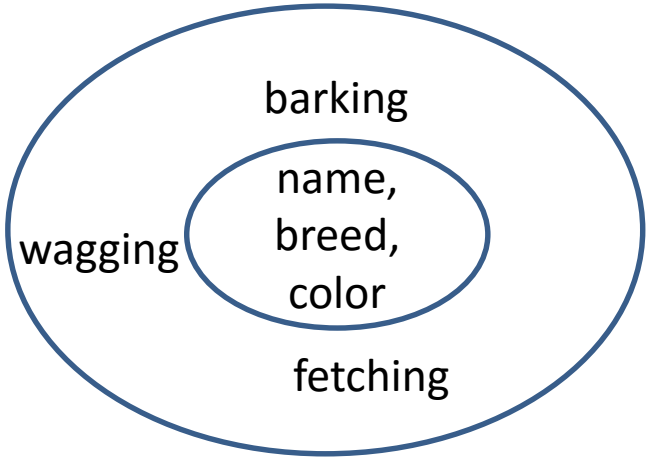
**Box as a Software Object**




**Dog as a Real World Object**



**Dog as a Software Object**



# What is Class ?

innovate achieve lead

- Objects are grouped in classes
- A class is a collections of objects having similar behavior and attributes
- An object is simply a single instance of class.
- Objects can not be instantiated (or created) without defining a class
- Classes are defined whereas objects are created.
- In order to create an object, you have to first define the class of that object

# Class Example : Box Class

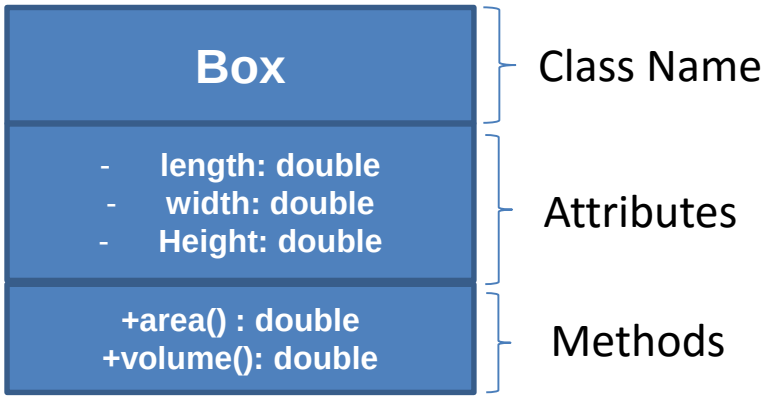
```
class Box
{
    private          double    length;
    private          double    width;
    private          double    height;
    public           double    area()
    {
            return 2* (length * width + width * height + height * length);
    }
    public           double    volume()
    {
            return length * width * height;
    }
}// End of class
```

Instance Fields

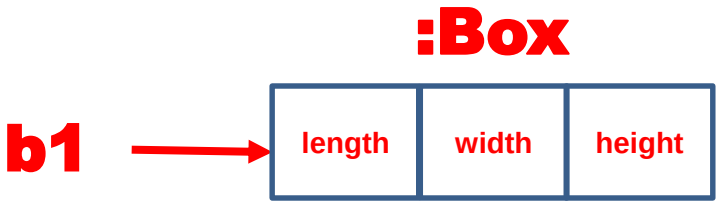
Methods

# Class Example : Box Class

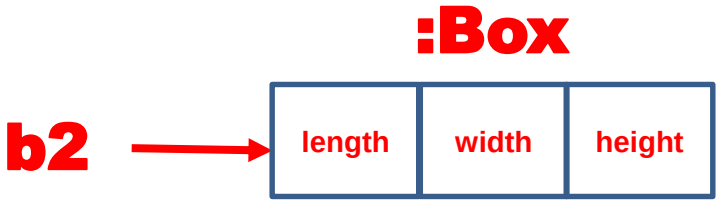
```
class Box
{
        private  double         length;
        private  double         width;
        private  double         height;
        public   double         area()
        {
                return 2* (length * width + width * height + height * length);
        }
        public   double         volume()
        {
                return length * width * height;
        }
}// End of class
```




**Box b1 = new Box();**




**Box b2 = new Box();**

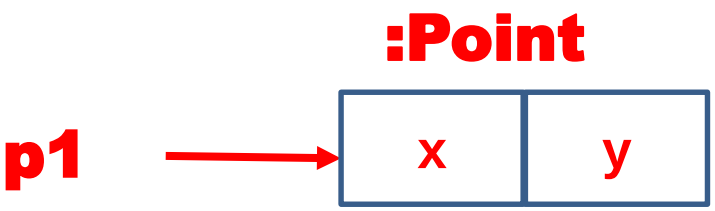


# Class Example : Point

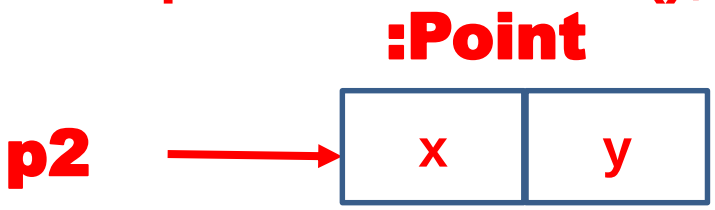
```
class Point
{
        private  double         x;
        private  double         y;
        public   double         getX()
        {
                 return         x;
        }
        public   double         getY()
        {
                 return         y;
        }
        public   void           setX(double value)
        {
                 x = value;
        }
        public   void           setY(double value)
        {
                 y = value;
        }
}// End of class
```

| Point |
| --- |
| - x: double<br>- y: double |
| +getX() : double<br>+getY(): double<br>+setX(value:double):void<br>+setY(value:double):void |

Point p1 = new Point();




Point p2 = new Point();



# Exercise

- Think about the class named ‘Student’. Define its state and operations
- Define the instance fields (attributes) and methods of class named ‘Line’.