OBJECT ORIENTATION

PROGRAMMING MODELS

- Procedural Programming Model
- Object Oriented Programming Model

PROCEDURAL PROGRAMMING

- Divides each problem into simple problem
- Solves the problem using specified modules that act on data

OBJECT ORIENTED PROGRAMMING MODEL

- Perceives the entire software system as a collection of objects which are interrelated and interact with each other.
- Contains objects that have attributes and behaviour

ADVANTAGES OF OOP

- Real-world programming
- Reusability
- Modularity
- Information Hiding

KEY CONCEPTS IN OOPS

- Object
- Class
- Abstraction
- Encapsulation
- Polymorphism
- Inheritance

OBJECT

- Object is the basic unit of Object Oriented Programming
- An Object is a self-contained entity which has the data and operations working on that data.
- Some Examples:
  - Car, Person, Chair, Book, Marker, Table etc.

OBJECT

- An object has
  - State
    - properties of an object
  - Behaviour
    - methods which reflect the response of an object with other objects
  - Identity
    - unique name to an object and enables objects to interact with each other

OBJECT EXAMPLE

- Lets take an example of Car as an Object
  - State: Car is in a particular state at a given point of time whether it is in running state or Stopped state
  - Behaviour: The behaviour of the Car is to move people from one place to another
  - Identity: Each Car is different. Even if two cars looks identical and have same features, they are two different objects. There is always a unique identification associated with the objects to separate them

CLASS

- A class is a user-defined blueprint or prototype from which objects are created.
- A class represents the set of properties and methods that are common to objects of one type.
- Example,
  - If an urban planner has to plan and built an multiple 2 Bedroom flats, the planner is going to first prepare a blueprint for building a 2 bedroom flat. With the help of the blueprint, the urban planner develops multiple 2 bedroom flats
  - Here, the Blueprint is the class and the concrete houses are the objects

ABSTRACTION

- Abstraction
  - managing the complexity by separating interaction details from implementation details
  - In simple words – Knowing what to do without knowing how to do it

- Example
  - ATM machine: multiple operations like cash withdrawl, money transfer etc. are performed without knowing the internal details how they are actually performed.

ENCAPSULATION

- Combining the data and methods into a single capsule
- Process of hiding all of the details of an object that do not contribute to its essential characteristics
- Restricting access to certain components

ENCAPSULATION VS ABSTRACTION

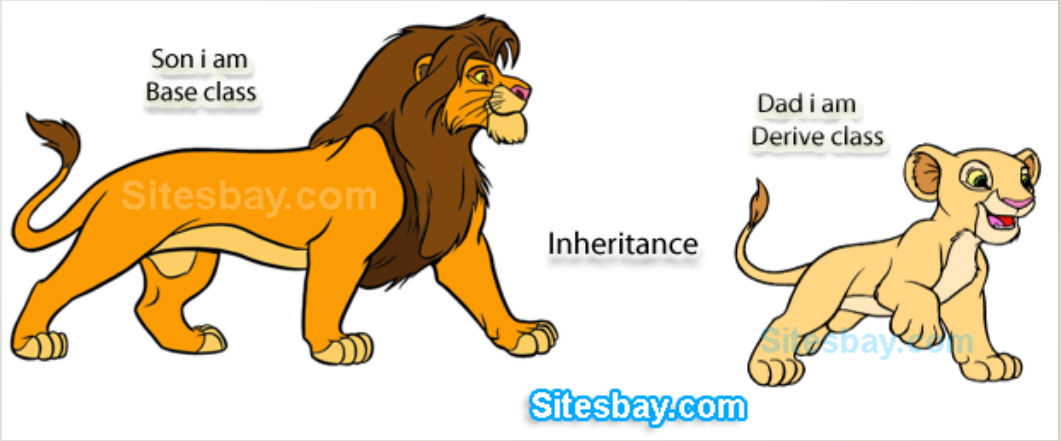
Encapsulation hides the irrelevant details of an object and **Abstraction** makes only the relevant details of an object visible.

POLYMORPHISM

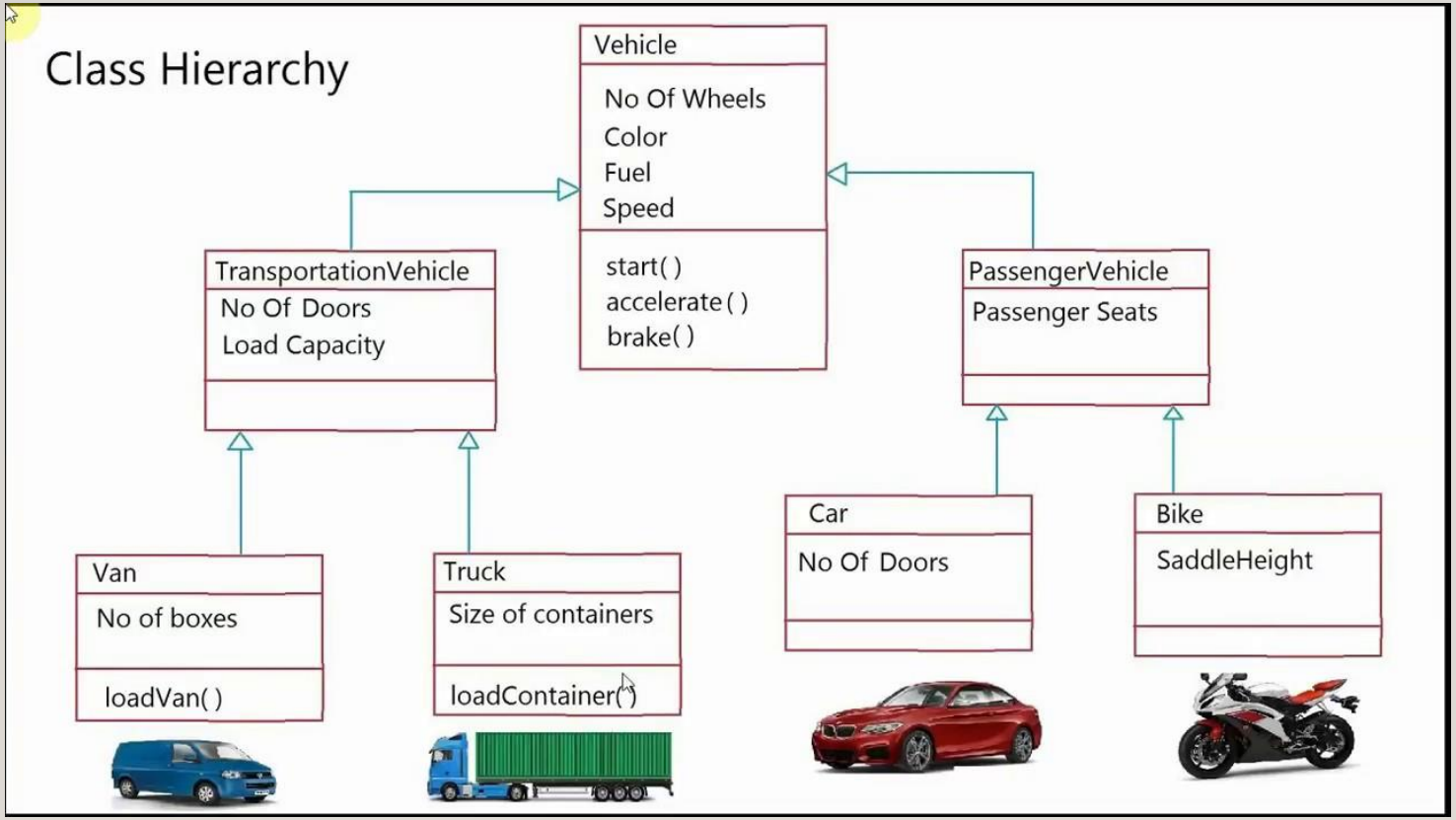
- Process of representing one form into multiple forms.
- For example, in English, the verb ***run*** has a different meaning if you use it with *a laptop*, *a race*, and *business*. Here, we understand the meaning of *run* based on the other words used along with it.
- In object-oriented programming, *polymorphism* refers to a programming language's ability to process objects differently depending on their data type or class.

INHERITANCE

- Inheritance is a mechanism in which one object acquires all the properties and behaviour of parent object.
- Helps in reuse of attributes and methods
- The concept of inheritance is also known as re-usability or extendable classes or sub classing or derivation.



INHERITANCE



ADVANTAGES OF OBJECT ORIENTATION

- hides the implementation details of an object
- ignores non-essential details and concentrates on essential details only
- is capable of making objects react differently to a message
- allows reusability of code

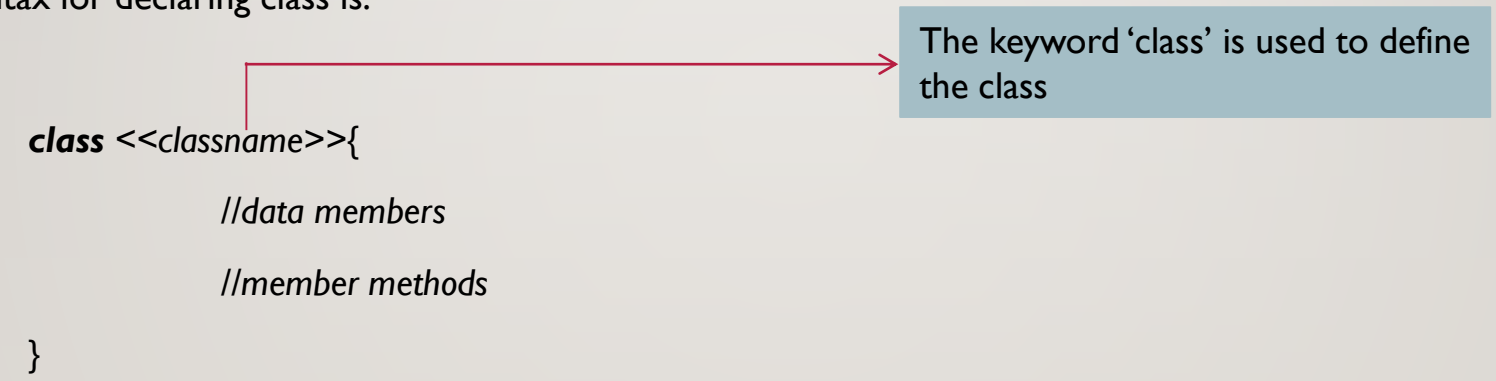
JAVA AND OBJECT ORIENTATION

- Defining Classes
- Instantiating new Objects
- Defining Data and methods inside a class
- Constructors
- Using this operator

DEFINING CLASS

- A class contains data members and methods which perform actions on the data members.

- The syntax for declaring class is:



- Both Data members and member methods are optional. Depending on the requirement, we can have any of those or both.

EXAMPLE OF A CLASS

```
public class Account {
    int accountNumber;
    String accountHolderName;
    double accountBalance;
}
```

- The example declares a class "Account" with three data members.
- Note that its not necessary to have methods in a class.

DECLARING OBJECTS

- When an object of a class is created, it is also called as **instantiating** a class.
- All instances of a class share the data members and behaviour of the class
- A single class may have one or more than one instances
- In Java we use the **new** keyword to create an object
- Example:

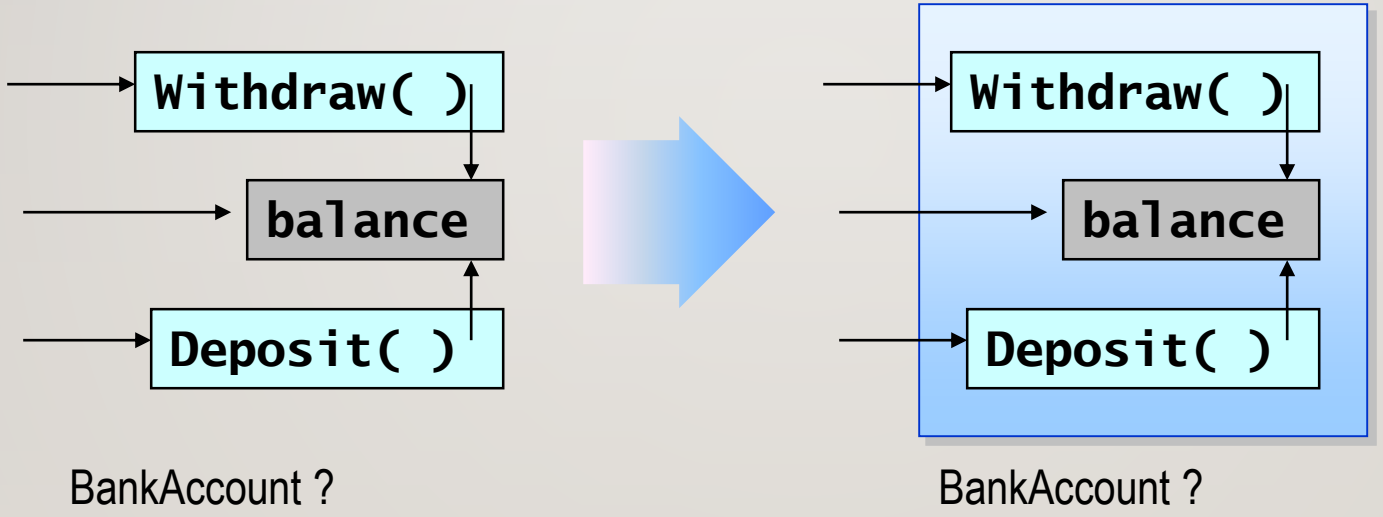
```
Account account;   //declaring a reference variable
account = new Account();  // creating new object
```

DECLARING OBJECTS

- As can be seen in the example, the object creation in Java is a two-step process. First we need to declare a variable of Class type. This is a reference variable.

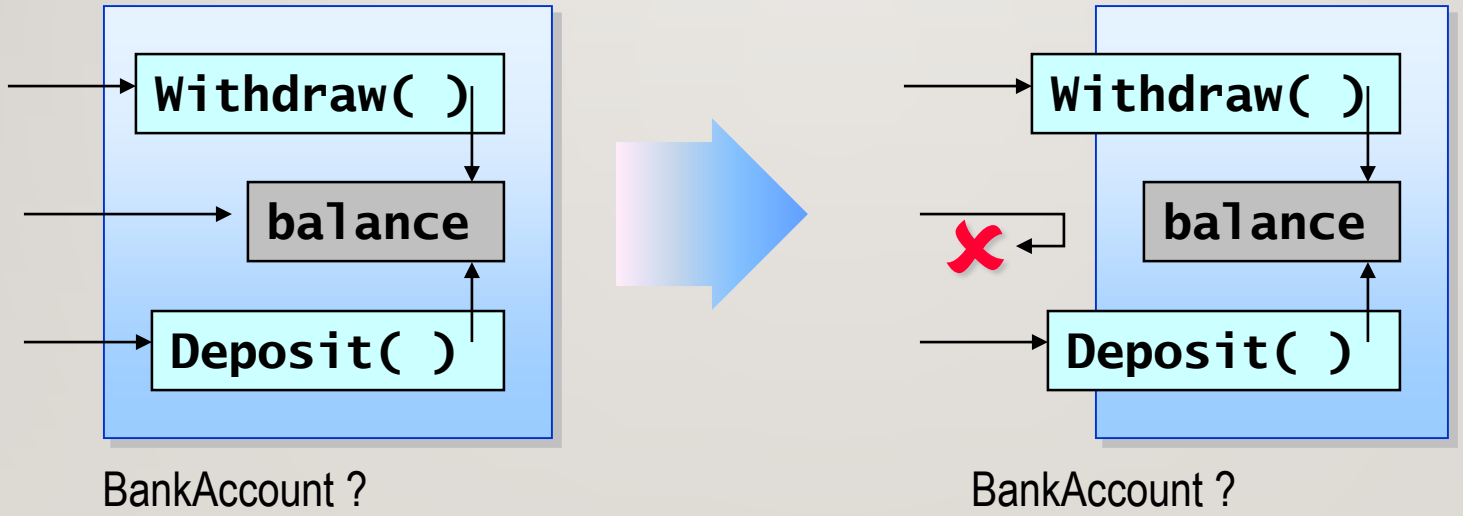
ENCAPSULATION - COMBINING DATA AND METHODS

- The capsule boundary forms an inside and an outside



CONTROLLING ACCESS VISIBILITY

- Methods are *public*, accessible from the outside
- Data is *private*, accessible only from the inside



WHY ENCAPSULATE?

- Allows control
  - Use of the object is solely through the public methods
- Allows change
  - Use of the object is unaffected if the private data type changes