Programming (OOP)

Classes → Objects → what is this class representing

Definition of an entity

Student : 1) data elements (Members)

Has a: name, ID, address, units, phone

2) interface Methods to access or modify the student data (Accessors/getters and Modifiers/setters)

Made once

Real life (Engineering problem)

Radio → what are the requirements

Objects : Representation of a student

int	num ;	Student	st1;	
Data type	variable	Class	Object	
float	f1;	String	name;	name.equals("Joe");

System classes used so far: File, Scanner, String, Arrays, Math, Math.sqrt(78) upper= st.toUpperCase()

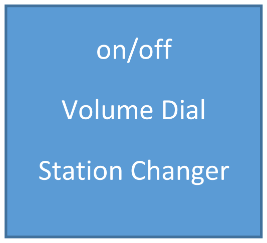
Class Object is the root of all classes in Java

objectName●method();

Radio → come up with a plan of what is involved in building a Radio:

Schematic → Electrical parts + Interface

Made once Unit of Radio many times

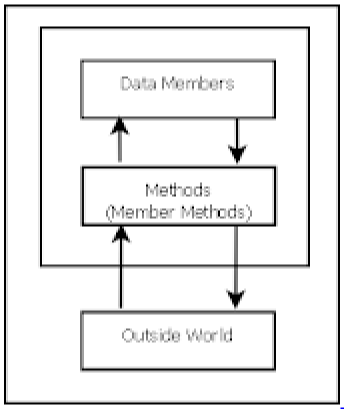


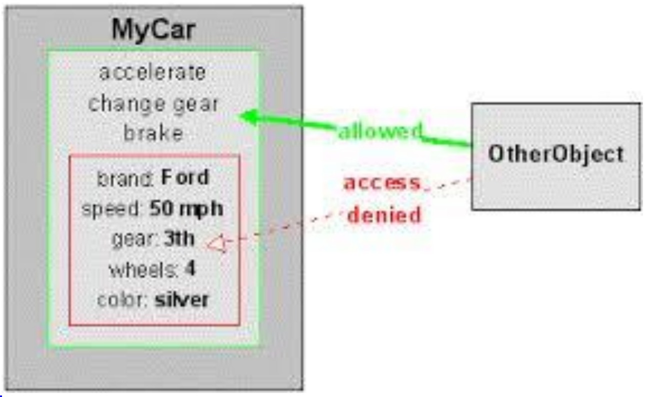
Object Oriented Programming: (OOP)

Digitizing entities to store them on computers

- Data Hiding
  - This insulation of data from direct access by the program is called **data hiding.**
- Encapsulation:
  - The wrapping up of data and methods into a single unit (class) is known as encapsulation. The data is not accessible to the outside world and only those methods, which are wrapped in the class can access it. Methods provide interface between the objects data and the program.
- Polymorphism
  - Polymorphism means the ability to take more than one form. For example, an operation may exhibit different behavior in different instances. For example, consider the operation of addition - for two numbers the operation will generate a sum, but for two strings the operation would produce a third string by concatenation. Polymorphism plays an important role in allowing objects having different internal structures to share the same external interface.

- Data Abstraction
  - Abstraction refers to the act of representing essential features without including the background details or explanations. Classes use the concept of abstraction and are defined a list of abstract attributes such as size, weight, and cost and methods that operate on these attributes. Consider the example of a mobile phone where different manufacturers make mobile phones that are different instances of the common abstraction of a generic mobile phone, which involves actions like dialing a call, receiving a call, sending and receiving text messages etc.





Classes
The entire set of data and code of an object can be made a user-defined data type using the concept of class A class may be thought of as a data type and an object as a variable of that data type. A class is thus a collection of objects of similar types. For example, mango, apple, and orange are members of the class fruit.

Objects
Definition: An object is an entity that has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. The terms instance and object are interchangeable.

Objects are basic runtime entities in an object oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program may handle. Program objects should be chosen such that they match closely with the real world objects.

UML: Unified Modeling language (OOD)
Class diagram

Class Name
Class Attribute/data (Usually private)
Class methods/Behavior Interface / interaction (Usually public)

Sample:
Person:
Class Name: Person
Attributes: name, id, address, age
Methods: getName, getId , getAddress, setAddress , getAge, initializeObject (Constructor, getters, Setter)

Person
- name: String - id : long - address : String - age: int

+ Person() // Default Constructor
+ Person(String newName, long newId) // Alternative
+ Person(String newName)
+ Person(Person newPerson) // copy constructor
 	
+ getName() : String
+ getId() : long
+ getAge() : int
+ getAddress() : String
+ setAddress(String newAddress) : void

System String Class:

Name: String

Attributes: array of characters, length (size), ….

Method: st.length, st.toUpperCase(), ……