Object-Oriented Programming- An Introduction

Lecture 2
9/1/19

Prof. Anita Agrawal

OOP why????

- Computer scientists have struggled for decades to design new languages and techniques for writing software.
- Unfortunately, experience has shown that writing for large systems was virtually impossible.
- Small programs seemed to be no problem, but scaling to large systems with large programming teams could result in $100M projects that never worked and were thrown out.

- The only solution seemed to lie in writing small software units that communicate via well-defined interfaces and protocols like computer chips.

- The units must be small enough that one developer can understand them entirely and most importantly the units must be protected from interference by other units so that programmers can code the units in isolation

- The Object-Oriented paradigm fits these guidelines as designers represent complete concepts or real world entities as objects with approved interfaces for use by other objects.

Object Oriented Programming

- Four major principles:
- 1. Data Abstraction
- 2. Encapsulation
- 3. Polymorphism (dynamic binding)
- 4. Inheritance

- For many tasks, object-oriented programming has proven to be a very successful paradigm.
- Interestingly, the first object-oriented language (called Simula), was designed in the 1960's, but object-oriented programming has only come into fashion in the 1990's.

Other HL Programming languages/Procedural Programming

- Good for simple programs- Structured and splits tasks into smaller ones.
- But bigger the program, more lengthy and tougher to manage by programmers
- Example: C programming language

OOP ???

- **O**bject-**O**riented **P**rogramming is a programming paradigm based on the concept of "objects" and "classes"

- Entire world depends on **classes** and **objects** which are at the heart of the concept

Object???

- **Object**: is a software bundle of related **state** and **behavior**.

- Software objects are often used to model the real-world objects that you find in everyday life.

Class???

- A class is a blueprint or prototype from which objects are created.
- A class models the state and behavior of the object.

Object Oriented Programming Languages

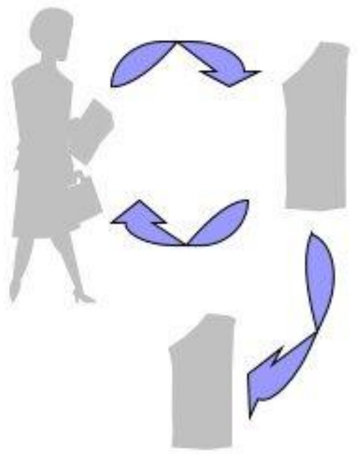
A methodology of programming used by popular programming languages like Java, C++, C#, Ruby, etc.

Procedural Programming vs Object Oriented Programming

- The unit in procedural programming is ***function***, and unit in object-oriented programming is ***class.***
- **Procedural** programming concentrates on creating **functions**, while **object-oriented programming** starts from isolating the **classes**, and then look for the methods inside them.

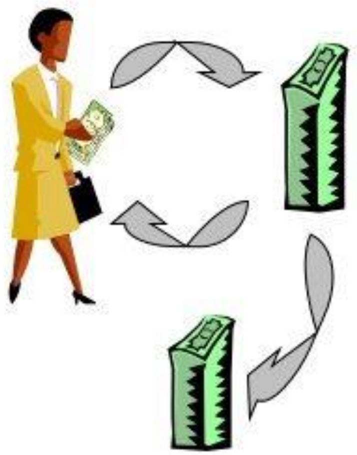
Procedural vs. Object-Oriented

- Procedural



Withdraw, deposit, transfer

- Object Oriented



Customer, money, account

Object.....

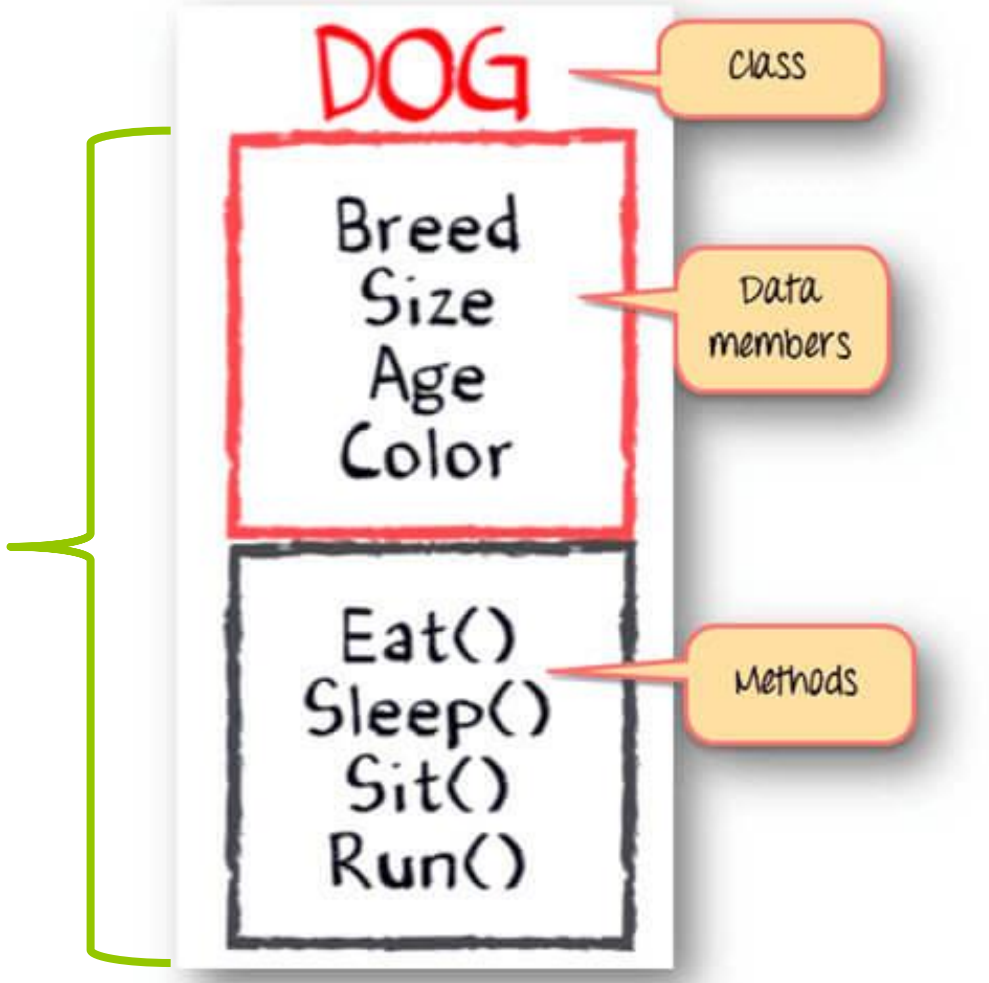
- It is a basic unit of Object Oriented Programming and represents the real life entities.
- It consists of :
  - State
  - Behavior
  - Identity

- **State-** It is represented by **attributes** of an object. It also reflects the properties of an object.
- **Behavior** - It is represented by **methods** of an object. It also reflects the response of an object with other objects.
- **Identity**-It gives a unique name to an object and enables one object to interact with other objects.

- **Attributes:**
  - These are the **instance variables**. Each object will have its unique set of instance variables.
  - An object's state is created by the values assigned to these instance variables

- **Method:**
  - It is basically a behavior.
  - A class may contain many methods.
  - It is in methods that the logics are written, data is manipulated and all the actions are taken

Class members



An example of an Object- Dog

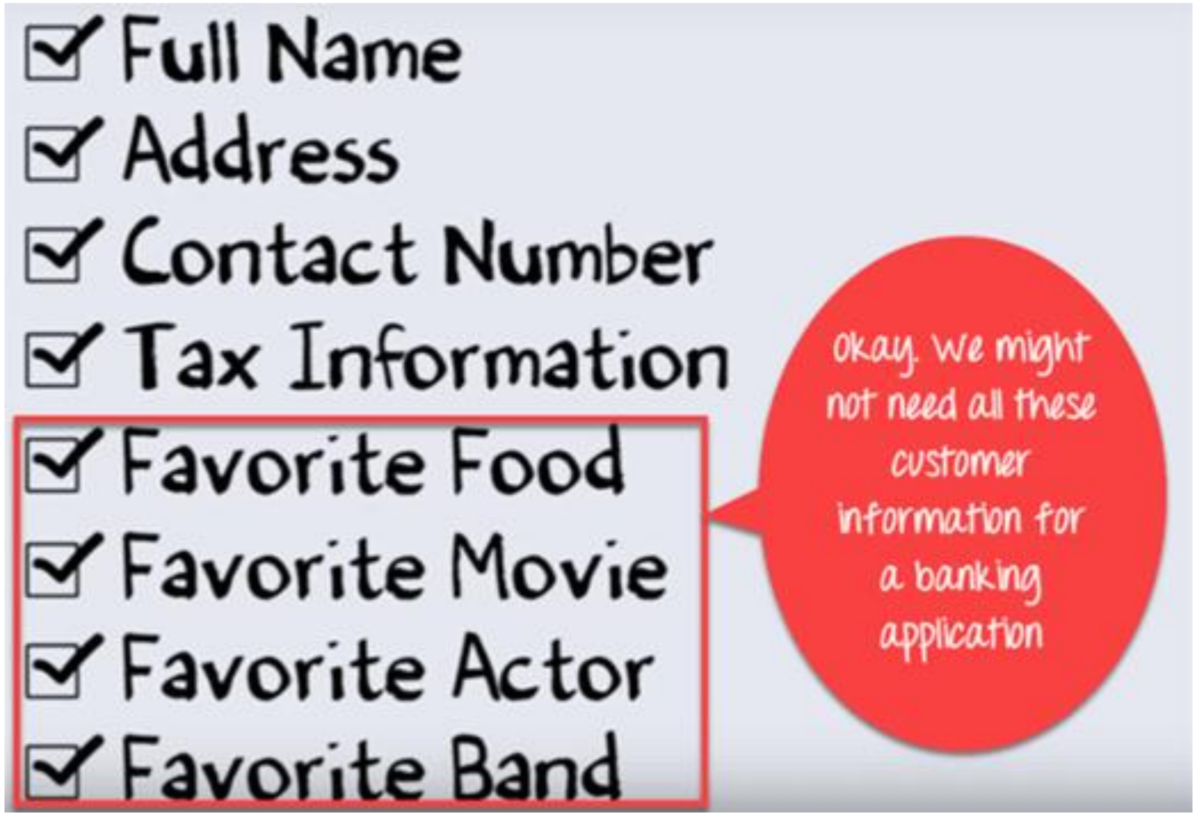
Identity	**State/Attributes**	**Behaviors**
Name of dog	*Breed*	*Bark*
	Age	*Sleep*
	Color	*Eat*

The 4 fundamental OOPS Concepts

1. Data Abstraction
2. Encapsulation
3. Polymorphism (dynamic binding)
4. Inheritance (particular case of polymorphism)

Data Abstraction

- It deals with **hiding the details** and **showing the essential things** to the user.
- In Java, abstraction is accomplished using **Abstract classes** and interfaces.
- An example of abstraction- A Banking application, where only required data fields are revealed to the user



- ✓ Full Name
- ✓ Address
- ✓ Contact Number
- ✓ Tax Information
- ✓ Favorite Food
- ✓ Favorite Movie
- ✓ Favorite Actor
- ✓ Favorite Band

Okay, we might not need all these customer information for a banking application

Encapsulation

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit.

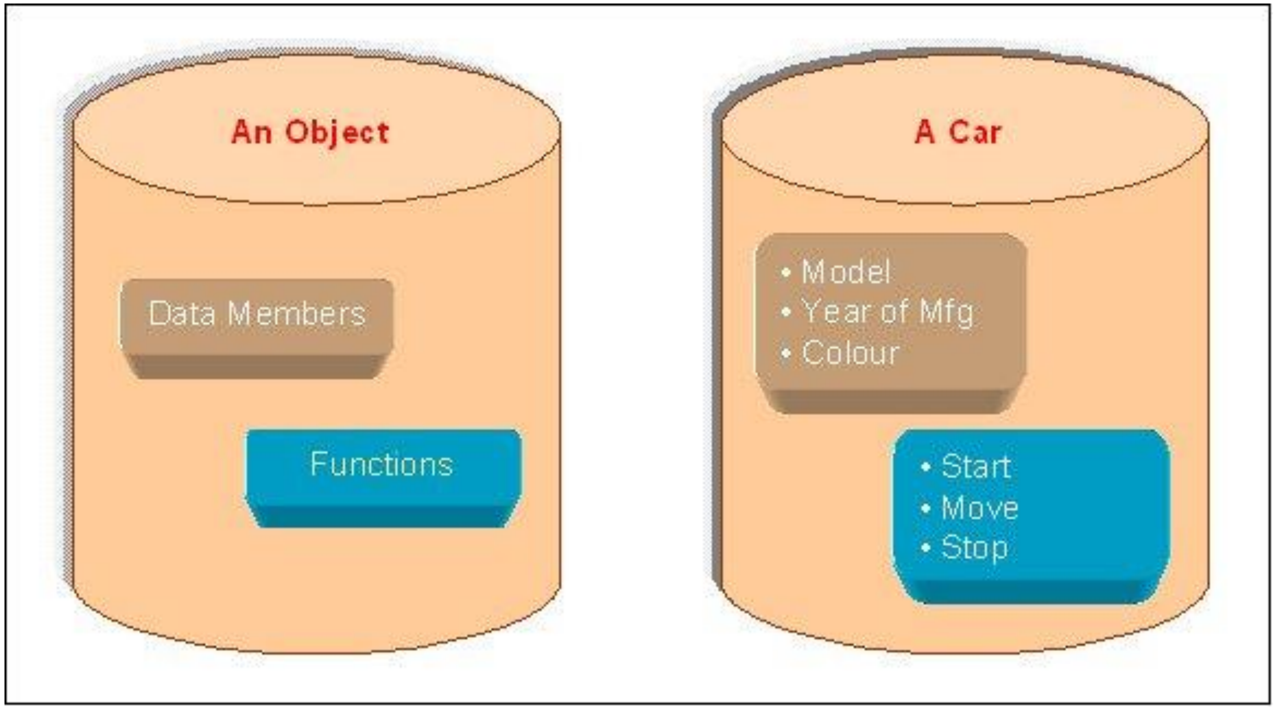
The variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class

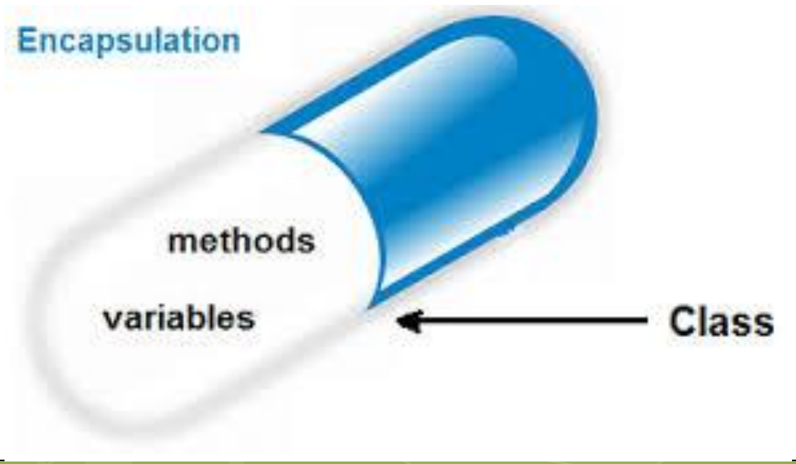
Encapsulation and classes

- Class contains:

$$\begin{tikzcd} \text{Data} \arrow[r] & \text{Member/Instance variables} \\ \text{Code} \arrow[r] & \text{Member methods} \end{tikzcd}$$

- Class encapsulates complexity


An Object
Data Members
Functions
A Car
• Model
• Year of Mfg
• Colour
• Start
• Move
• Stop


Encapsulation
methods
variables
Class

Inheritance

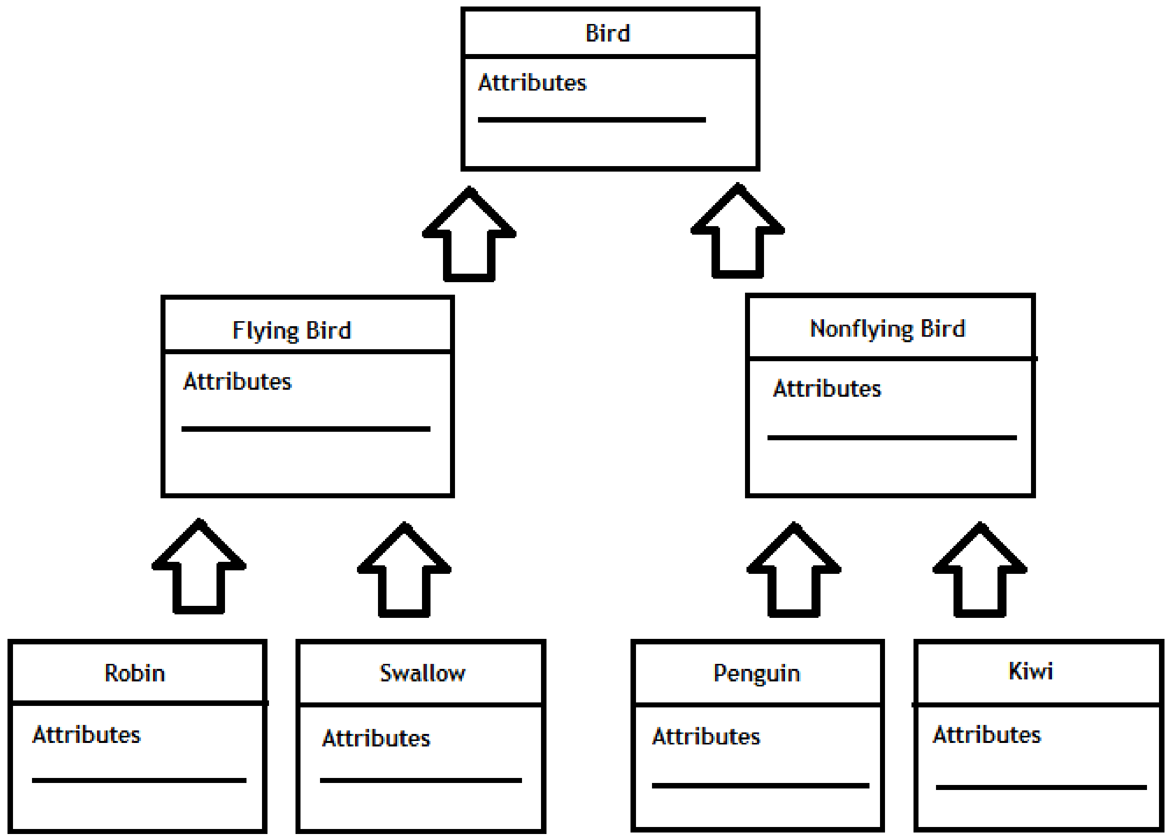
- A mechanism in which one class acquires the property of another class.
  - **Example, a child inherits the traits of his/her parents.**

With inheritance, the fields and methods of the existing class can be reused.

- An inherited class is called a **subclass** of its parent class or super class.

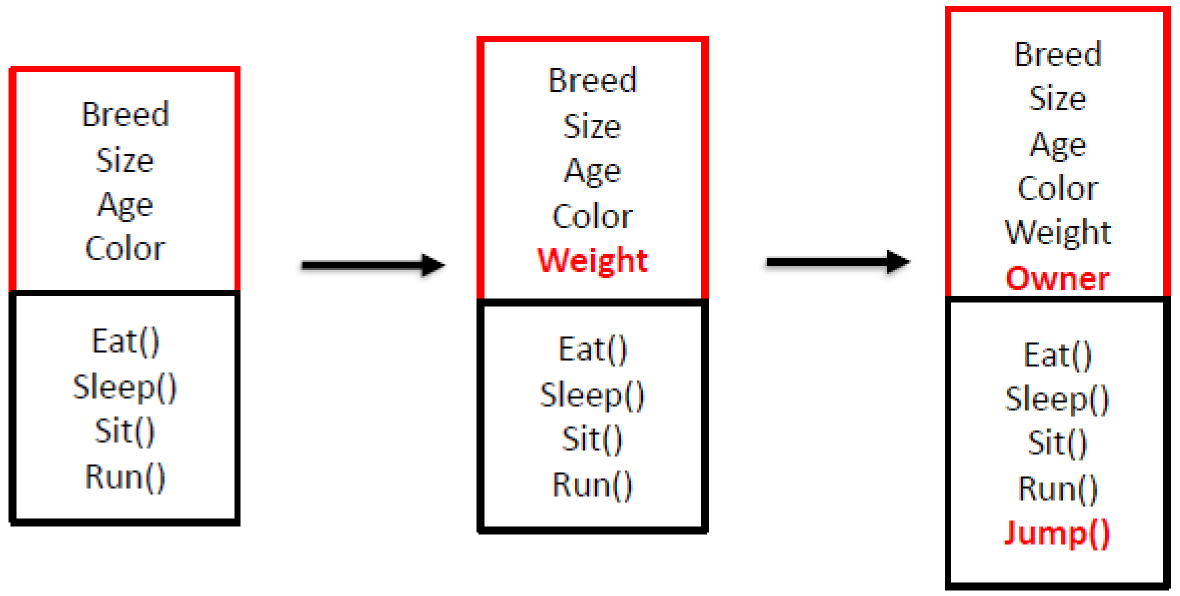
- It can be of different types- Single, Multiple, Multilevel, Hierarchical and Hybrid

Real World example of inheritance



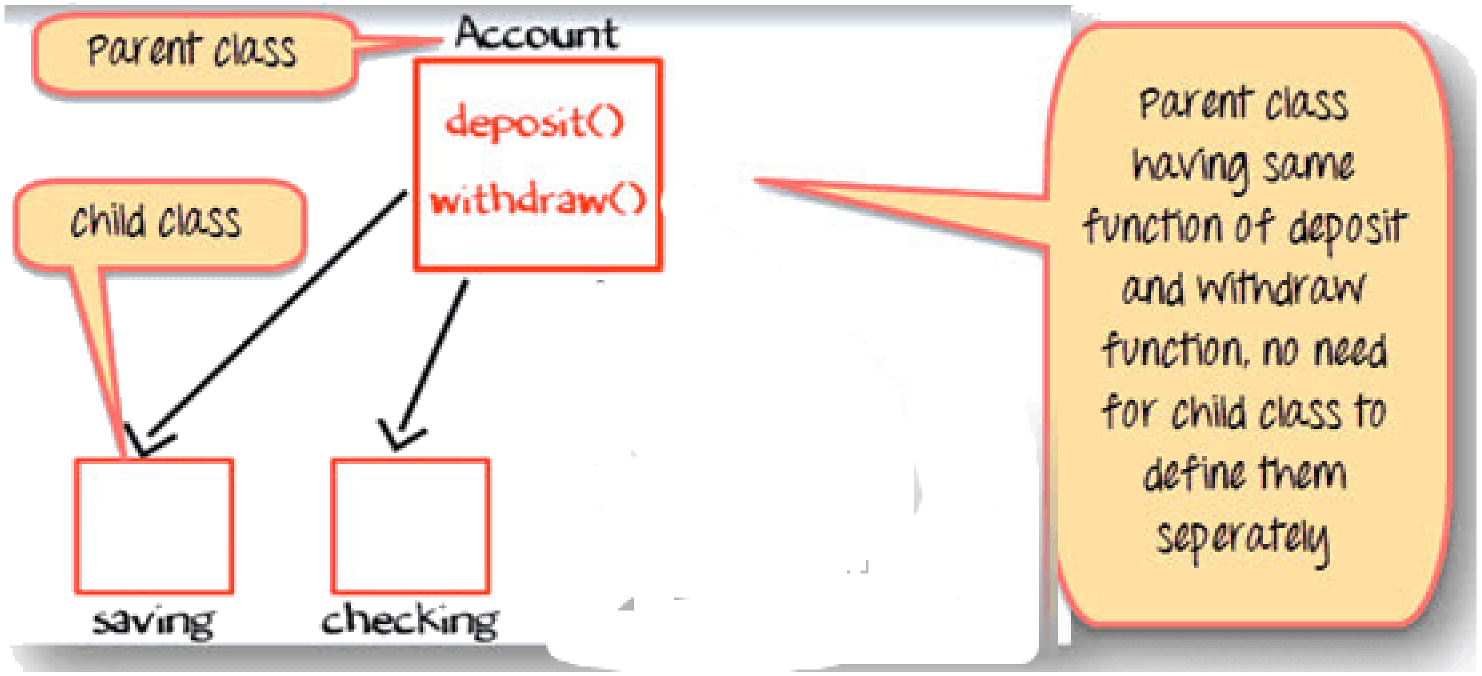
Another example of Inheritance:

One class acquires properties of another class



Supports hierarchical classification

Real World Example in a banking system


Parent class
Account
deposit()
withdraw()
Child class
saving
checking
Parent class having same function of deposit and withdraw function, no need for child class to define them seperately

Polymorphism

- It is a OOP concept where one name can have many forms, i.e. variable, function or object can take on multiple forms.
- In a real world example, a person at a same time can have different characteristic.
- Example- Calculate the Area-