Abstract Class
- Fields
- Ctor
- non abstract methods
- abstract methods

Employee
calculateTotalSalary()

Manager Salesman

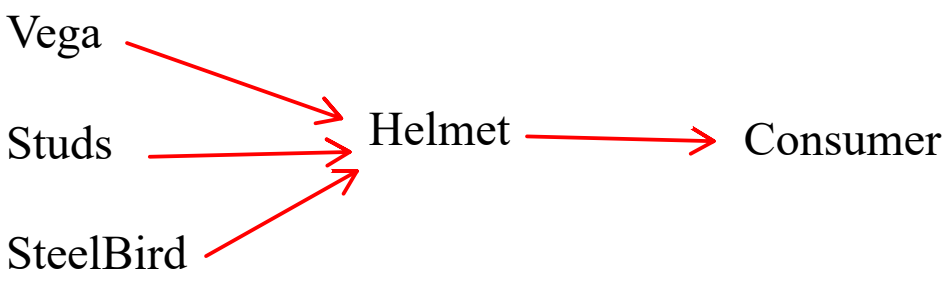
accept()
display()
calculateTotalSalary()

Fragile Base Class Problem

Interface

Interface
- It is a contract between Service Provider and a Service Consumer
- It Provides Set of Rules/Protocols
- It is also used to group unrelated types together

ISI, EU

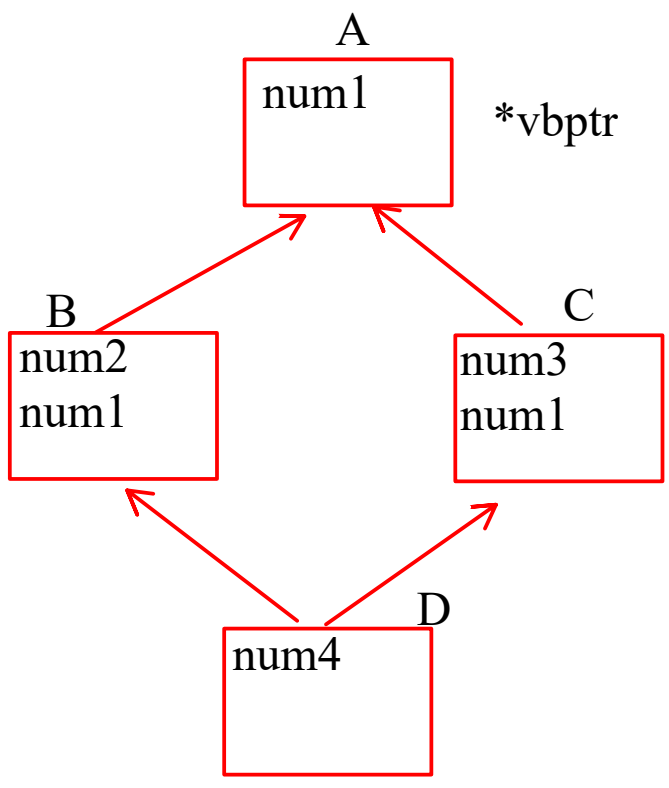


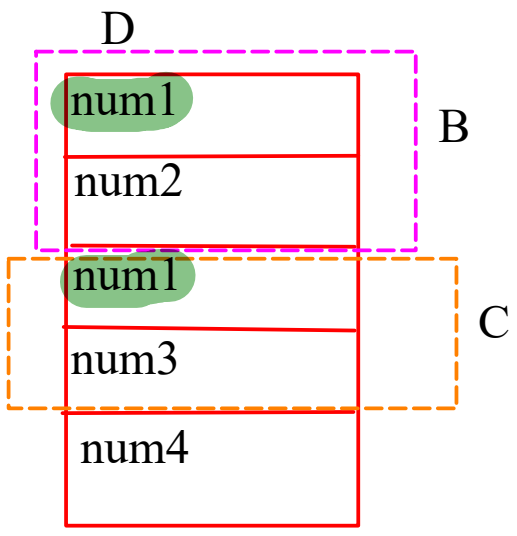
Types of inheritance

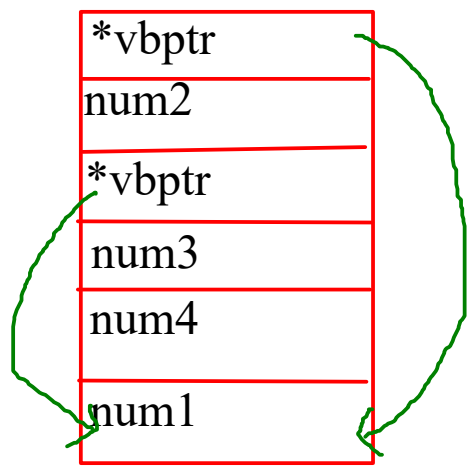
interface Sunbeam{
1.8AM lecture
2. Monday & Thursay Uniform
}

interface CDAC{
1. 75% Attendance
2. 8AM lecture
}

class Student implements Sunbeam, CDAC







class vs abstarct vs Interface

class Employee{
id
name
salary
}

class Manager extends Employee{
bonus

equals(Object obj){
if(obj instance of Manager)
Employee m = (Employee)obj;

class vs abstarct vs Interface

Class
- It consists of non abstract methods
- We can have static as well as non static fields
- We can have a ctor inside the class
- we can create object of the class

Abstract Class
- It consists of both abstract as well as non abstract methods
- We can have static as well as non static fields
- We can have a ctor inside the abstract class
- we cannot create object of the abstract class only reference is allowed

Interface
- It consists of all public abstract methods
- By default all the declared fields as public static and final
- We cannot have a ctor inside the interface
- we cannot create object of the interface only reference is allowed

Marker interface
- Empty interface is called as marker interface
- these interfaces are also called as tagging interface
- marker interfaces are used to provide extra information / metadata to the JVM
- eg -> java.lang.Cloneable is a marker interface
    -> java.io.Serializable is a marker interface

Exception Handling
- In java, A program that is in execution can generates some problem
- The problem is categorized as
    1. Error
        - A problem that is generated by RunTimeEnvironment
        - HDD Crash, Stack/Heap is full
        - Java recomends not to handle the errors, crash the program
    2. Exception
        - A problem that is generated by the instructions from the program
        - Wrong inputs given to the program represents exception
        - NullPointerException, ArithmeticException.....
        - It is recommended to handle the exceptions

Throwable
- A class that represents all the errors and exceptions
- It have 2 sub classes that refres to the errors and exceptions in java

1. Error
2. Exception

Exception Handling keywords

1. try
    - Used to check for the statements that generate exceptions or no.
2. catch
    - Used to handle the exceptions
3. throw
    - used to generate new Exceptions
4. throws
    - used to navigate/route the exceptions from current method to the caller method
5. finally
    - Used to close the resources.
    - gets executed every time irerespective of exceptions generated.

- Exception in java is further classified as

1. Checked Exception
    - Exception class and all its subclasses except RuntimeException class are all checked exceptions
    - checked exceptions are compulsary to handle
2. Unchecked Exception
- RuntimException class and all its subclasses are considered as Unchecked Exceptions
- unchecked exceptions are optional to handle