INTRODUCTION TO OOP'S

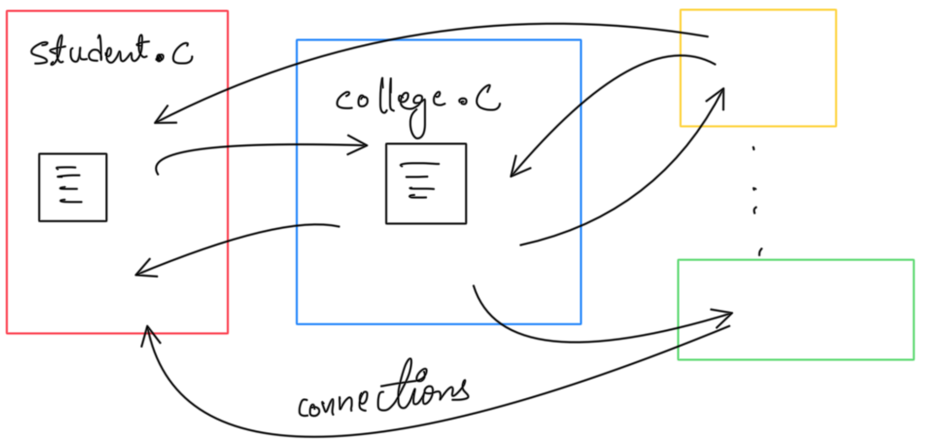
Day 26
05/08/2025

- Why we Need Object Oriented Programming
- Real World Examples
- Key concepts
  - class
  - object
  - Encapsulation / Bundling of data and methods
  - Constructor
  - Access Modifiers

Most of the Modern Softwares are Built using object Oreinted programming

Why We Need Object Oriented Programming

Before using OOP's concept programs of Software are Written In Different files.



codes are Organized in different files.

When software Data Increases or The use of Software Creates So many connection's Between other code files of a software which created complexity to understand the code & Difficulty to modify

Maintainace & Management Becomes a Big issue

To Solve this issue Object Oreinted programming is Introduced.

Suppose Take any real world Activity or concept What are the Entities we required to create a software for it.

Entities ⟶ Real world object's ⟶ Actors

Lets take a Library Management system as Example

What are the Entities Here ?

These Entities Become class

(1) Books (3) staff
(2) Students (4) orders

] Analyze the Department and list all Entities

The task performed By different Entities are considered as operations.

Example : Login / Logout

from student stand point → [checkout of Books
check In of Books] → Action
operations or Action

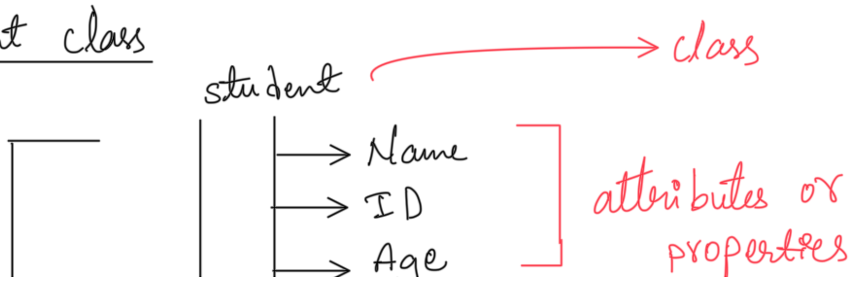
from library working staff → [add Books
Request of Books
Stocks of Books
New student registrations]

All these Actions Become Functions or Methods
and All the Entities Become classes

Entities ⟶ Classes
Entities Actions ⟶ Method

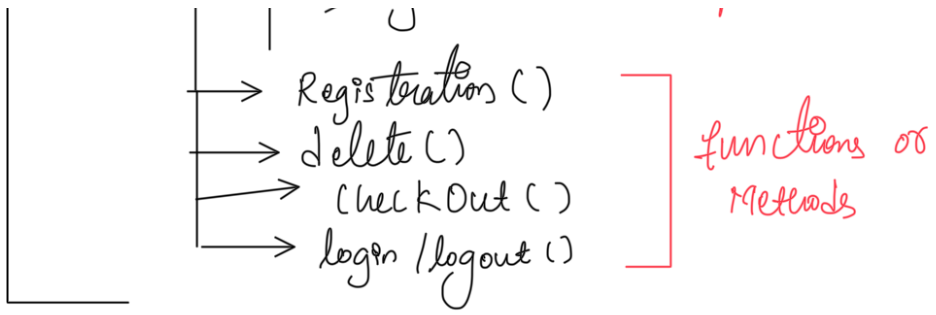
What is class ?
Encapsulation All the Data & Methods related to a particular Entity. (Its Encapulating technique.)

Example :- student class



Boundaries are

Created to
Create Safe
Softwares.

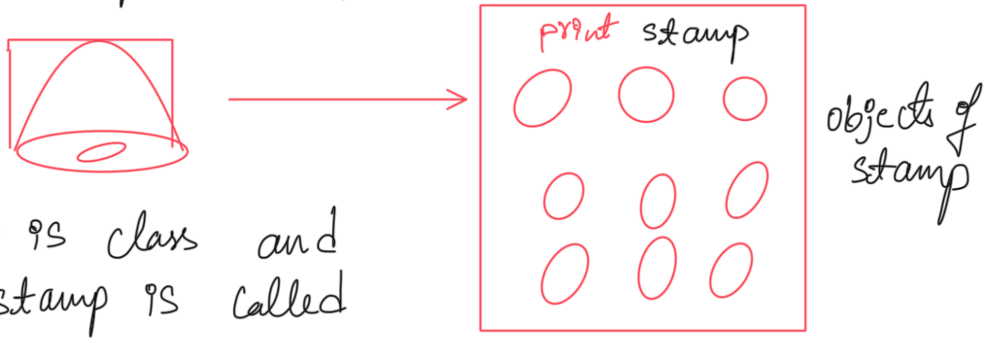


How to use This class ?

classes are used By Creating objects

lets take stamp as Example

stamp class



Here stamp is class and
points of stamp is called
object.

In Memory

student Veeresh = new student () ;

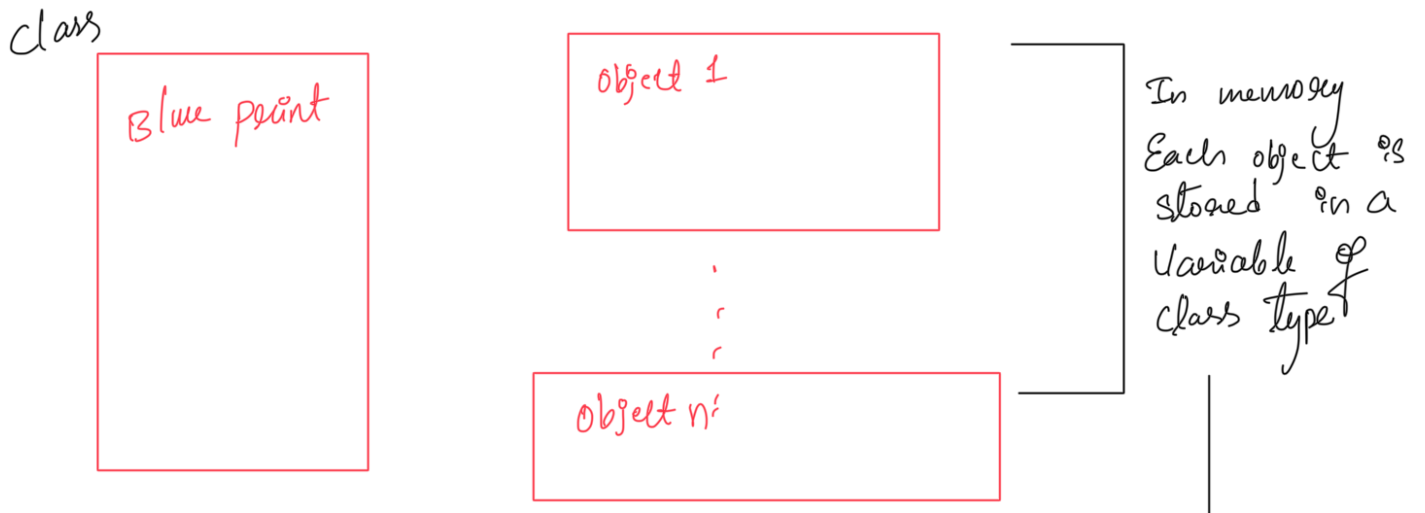
keyword used to Create object of class
Defines Data type of Variable or Instance
Variable or Instance
stored Here
object of student class
Class Name

Here Variable "Veeresh" is a Variable of type "Student class" which stores the address of object of Student named "Veeresh"

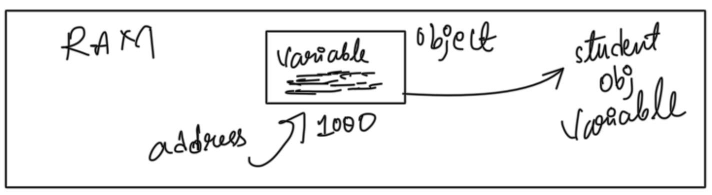
We created a New Data type

class is Nothing But creating custom Data type

→ Like this we can Create a "N" Number of Instances or objects with the help of class

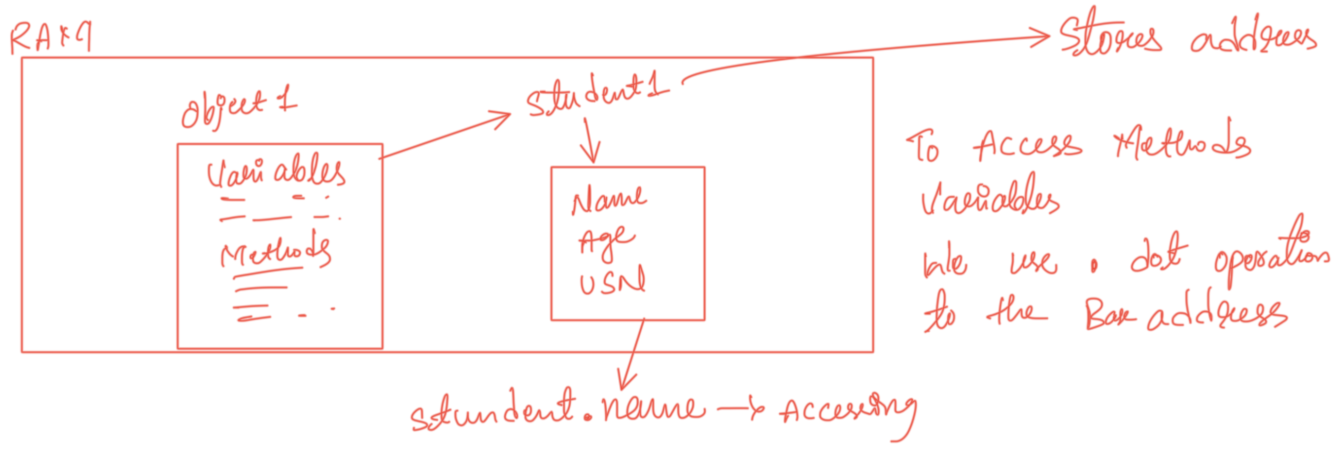


Each Variable stores address of respective objects



classtype Variable = new ClassName ( ) ;

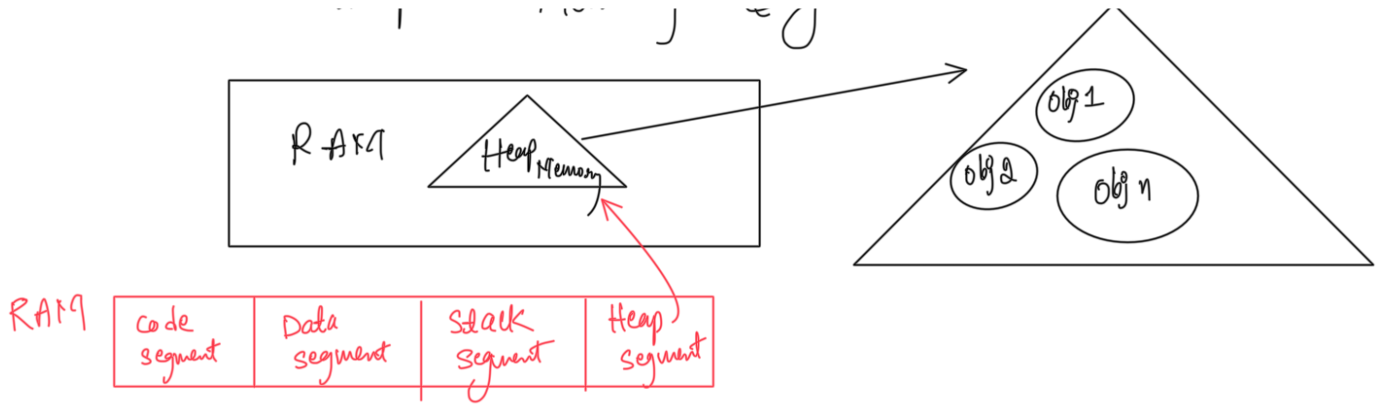
→ stores the reference address of object of classtype data



To Access Methods Variables We use . dot operation to the Box address

Objects are allocated dynamically (Dynamic Memory Allocation)

So the Memory for objects are Allocated In RAM In Heap Memory Segment.



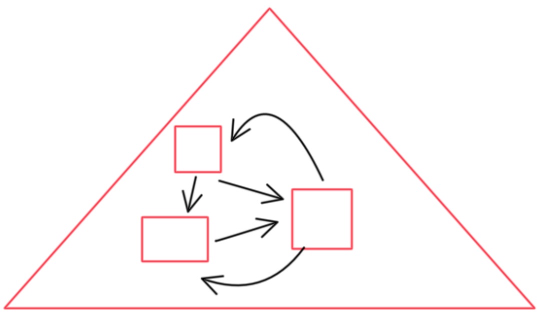
Now we learned about classes and objects
we also learned about creation of Instances

But How objects are deleted (deletion of unused or objects which dont Have reference)

deltion Handled By garbage Collector

Garbage collector find the un used objects and cleans from the Memory.

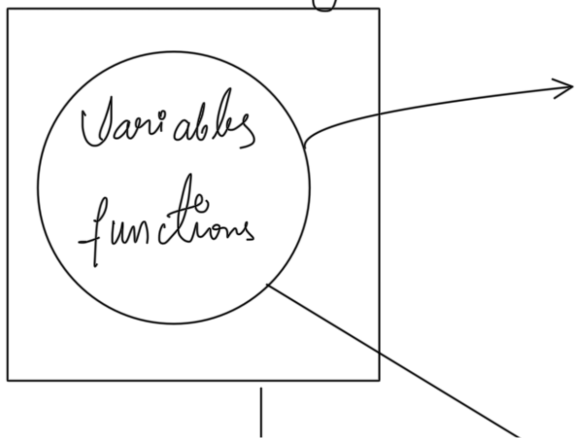
if we Have address inside Heap Memory we can Access it



objects are accessed only if we have address of it

Encapsulation ⟶ Bundling

student class



Bundling of Methods & variables of class object that cant be Accessed without permission

Encapsulation — Together Bundling In a Memory

Bundling of Data + Functions

<u>constructor</u> → used to set values to dataset of object

```
class student {

    public string Name;
    public string Age;          // variables or attributes
                                   or properties

  public string getName() {
        return this.Name;       // function

  }
  public student () {
      this.Name = " ";
      this.Age = 0;
  }
            or
  public student (string Name, int Age) {
       this.Name = Name;
       this.Age = Age;
  }


}
```

constructor

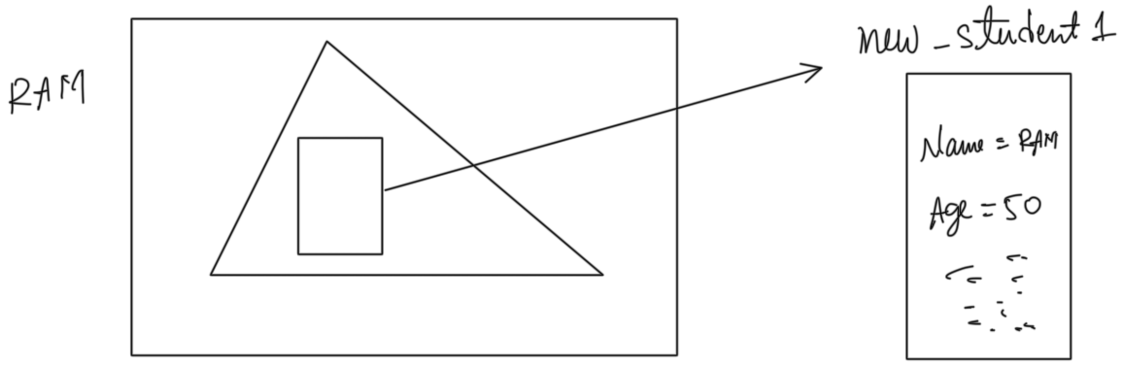
Example for polymorphism

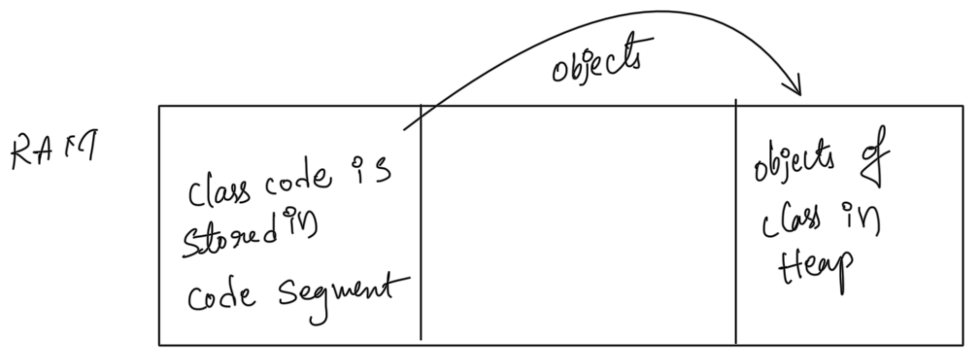
Same function Name different Behaviours

Creating obj Reserving space for it

student new_student1 = new student ("Ram", 50); Value assigned

constructor Invoked





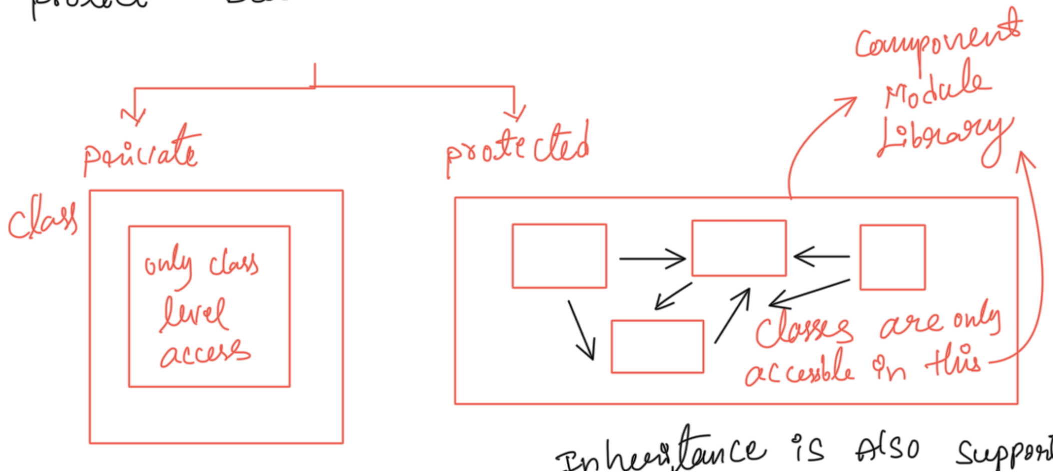
compilation

$$\begin{tikzcd} \text{HLL} \arrow[d] \\ \text{Lowlevel} \arrow[d] \\ \text{Binary} \end{tikzcd}$$

code always resides in code Segment
objects in Heap Memory

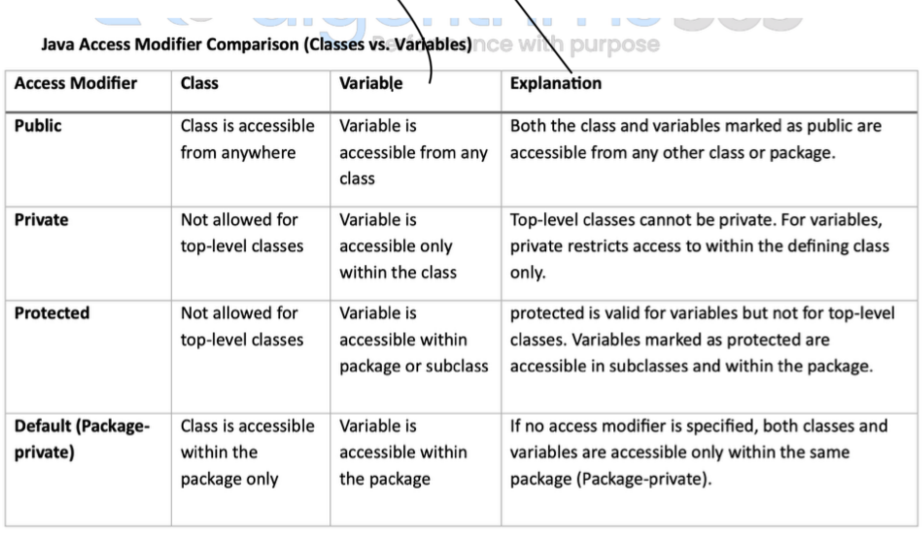
Access Modifiers
- → public → Any one can Access
- → private → There are Restriction for Access
- → protected → More Restrictions for Access

used to protect Data



Inheritance is Also supported

Access Modifiers Work on Variable level & class level



Java Access Modifier Comparison (Classes vs. Variables)

Access Modifier	Class	Variable	Explanation
Public	Class is accessible from anywhere	Variable is accessible from any class	Both the class and variables marked as public are accessible from any other class or package.
Private	Not allowed for top-level classes	Variable is accessible only within the class	Top-level classes cannot be private. For variables, private restricts access to within the defining class only.
Protected	Not allowed for top-level classes	Variable is accessible within package or subclass	protected is valid for variables but not for top-level classes. Variables marked as protected are accessible in subclasses and within the package.
Default (Package-private)	Class is accessible within the package only	Variable is accessible within the package	If no access modifier is specified, both classes and variables are accessible only within the same package (Package-private).

Default Access Specifier in Java

$\uparrow$

(package private)

High Security softwares are Built on OOP's Concept.

Access Modifiers $\longrightarrow$ Security guards