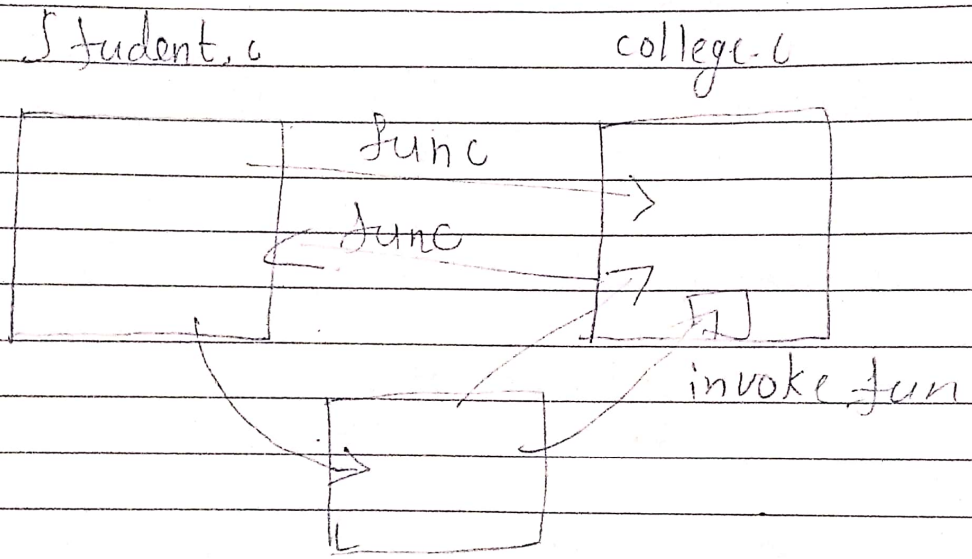
Object Oriented Programming

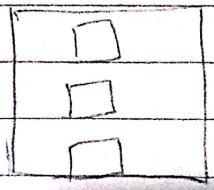
- Why we need OOPs
- Real world examples
- Key concepts
  - Class
  - Object
  - Encapsulation / Bundling of data & metho
  - Constructor
  - Access Modifier
    - public
    - private
    - protected

1) Why we need OOPs



- Struggle to refactore a code

project.c



Library Management System

class eg:
- Books
- Students
- staff
- orders (checkin/checkout)

→ Entities / objects / Actors

Operations / Function / Methods
- Login / Logout
- checkout, checkin
- Add books.
- Request Maintain
- Register students
- Fines

→ Actions

class: Blueprint of objects

Student
- Name, Id, Age {Attributes / properties}

Functions / methods
- Register(), Delete(), checkOut()
- Login, Logout

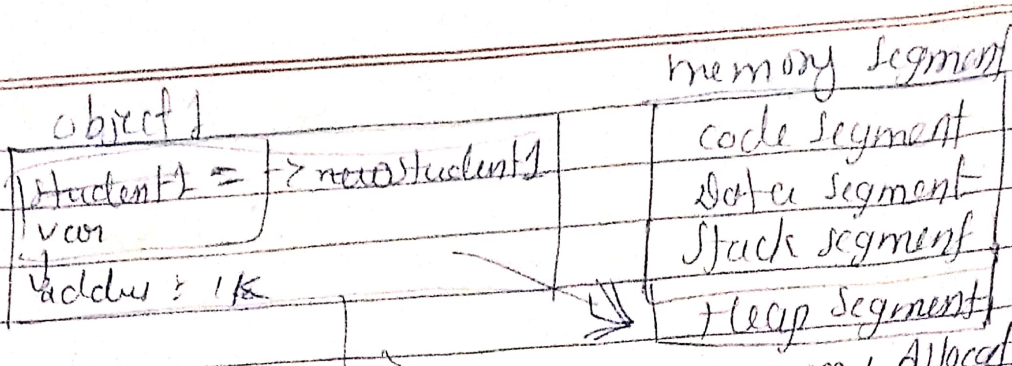
create objects of class to use classes functions

object:

In java

student1 new = new Student()
- new → instance of class
- Student() → class.

In py: student1 = Student()



Dynamic memory Allocation

cleaning: Garbage Collector

Encapsulation:

Acess only with function

Constructor

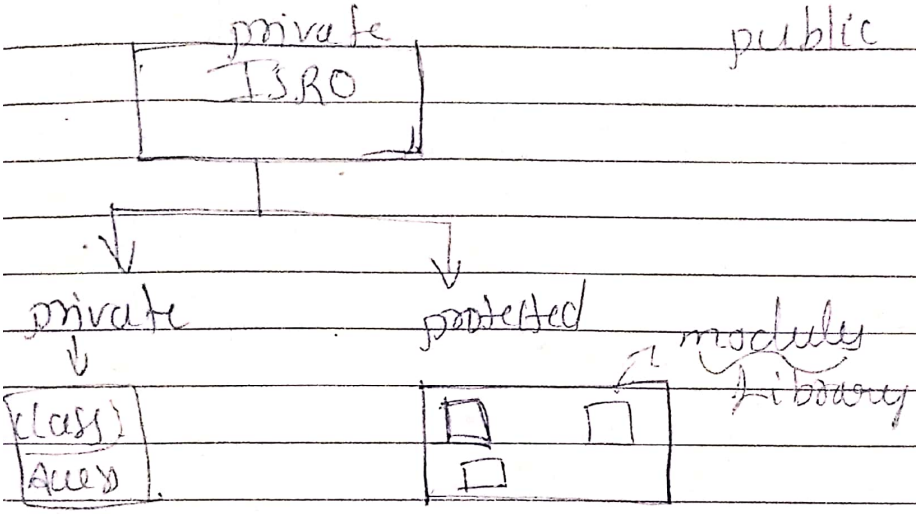
```
Class Student {
    public String Name;   & Attributes
    public int age;         variable

public String getName() {
        return this.Name; }

public int getAge() {
        return this.age; }

public Student() {  // constructor
    this.Name = "";
    this.age = 0 ; }

public Student(String Name, int age) {
    this.Name = Name;  // parameterized
    this.age = age ; } } constructor
```

RAM

Name = RAM
Age = 50

code Segment	class
Heap	objects

Access Modifiers
- public
- private
- protected



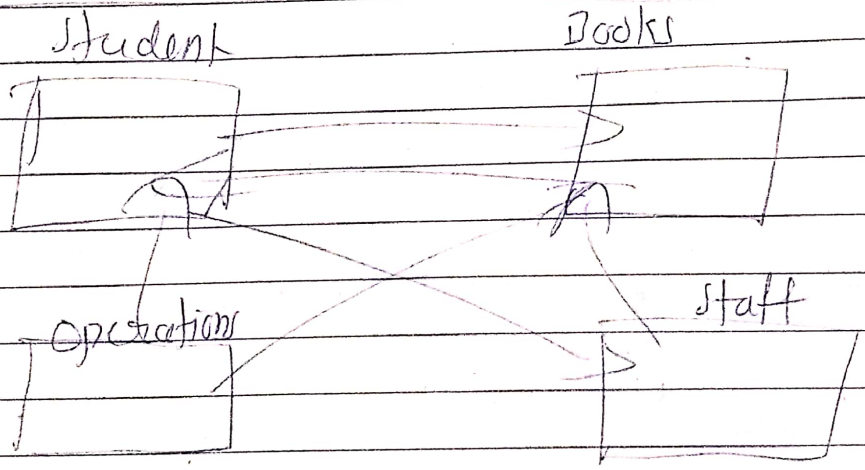
Default

class Student
{
}

c++/ private
java / public/private
py / public
(No keywords)

	C++	Py	Java
public	public	public (no keyword)	private
private	private		

	C++	Py	Java
public	public	No keyword class	private public
private	private	(--)	private
protected	protected	(_) prefix underscore	protected
Default	private	No keyword	private



-> Declaration
-> Defining
-> creating Objects