STUDYMATE

OOPS

Basics

OOPS refers to using objects in programming.

A class is a user defined blueprint from which objects are created.

Object is an instance of a class.

Object is referance of content / properties of it is stored in stack memory.

And the actual data gets stored in Heap memory.

eg. Creating class.

```
                    ClassName
public class Student{
    String name;
    int rollnumber;
}
```

* Creating objects

```
public class StudentInfo {
    public static void main (String[] args){
ClassName obj = new ClassName()
        Student S1 = new Student();
        Student S2 = new Student();
```

referance

default null 4 0 resp

```
        System.out.println(S1.name+" "+S1.rollnumber);
```

giving values

```
        S2.name = "xyz";
        S2.rollnumber = 7;
    }
```

★ Access Modifiers :-
① Public ② Private ③ default.

Access modifiers restricts controls the properties that have in the class.
Just add modifier before variable or property.
private int rollnumber;

① default :- They are accessible within same package.

② Private :- They are accessible only within same class.

③ Public :- Available to all packages within same project.

If you are using file within different package then you have to import the file.

★ Getters and setters.

Create function

Getter

```
public int getRollNumber(){
    return rollnumber;
}
```

```
objName.getRollNumber();
```

Setter

```
public void setRollNumber(int num){
    rollNumber = num;
}
```

```
obName.setRollNumber(100);
```

* 'this' Keyword : this holds reference of current objects.

Used to differentiate between local variable and object variable.

```
public void setRollNumber(int rollNumber){
    this.rollNumber = rollNumber;
}
```

local variable.

object variable.

* Constructor : You can think it as function, but can be called when the object has being created. It can called only ones in any objects lifetime.

```
Student s1 = new Student();
```

constructor.

Constructor does not have return type.
Name of constructor should be exactly same as class name.

```
public Student(String name, int rollNumber){
    this.name = name;
    this.rollNumber = rollNumber;
}
```

* 'final' Keyword :- No one can change the value of variable need to specify value.

```
final double conversionfactor = 0.95;

final private int rollNumber;

public Student (String name, int rollNuber){
        this.name = name;
        this.rollNumber = rollNuber;
}
```

- final variable can be initialized only once & cannot be modified further.
- We can initialize final variable inside constructor.
- final keyword is non-access modifier.

* Static keyword

static ——— belongs to class
shared by all objects

```
static int numStudents;
```

To access static variable no nee of object.

```
student.numStudents;
```

STUDYMATE

Date :
Page :

* Static functions :

You cannot access non static variable / function inside static function.
this or super keyword cannot be used inside a static fun.
non static variable / property is object dependent.

* Why main is public static ?

public -> So anyone can access it.
static -> No need to create object of the class because is first thing which runs first.