Programming Paradigm :
Style or standard way of writing a program

Without programming paradigm :
- Hard to read & understand
- Hard to test and debug
- Difficult to maintain etc.
- Less structured

OOPS is one such programming paradigm followed by languages : Java, Python, C++, C#, JS, Ruby etc.

Q. Maintain marks and names of students.

Name : [Bob, Alice, Charlie]
Marks : [92, 82, 87]

→ Maintainability
→ Data Association
→ Scalability

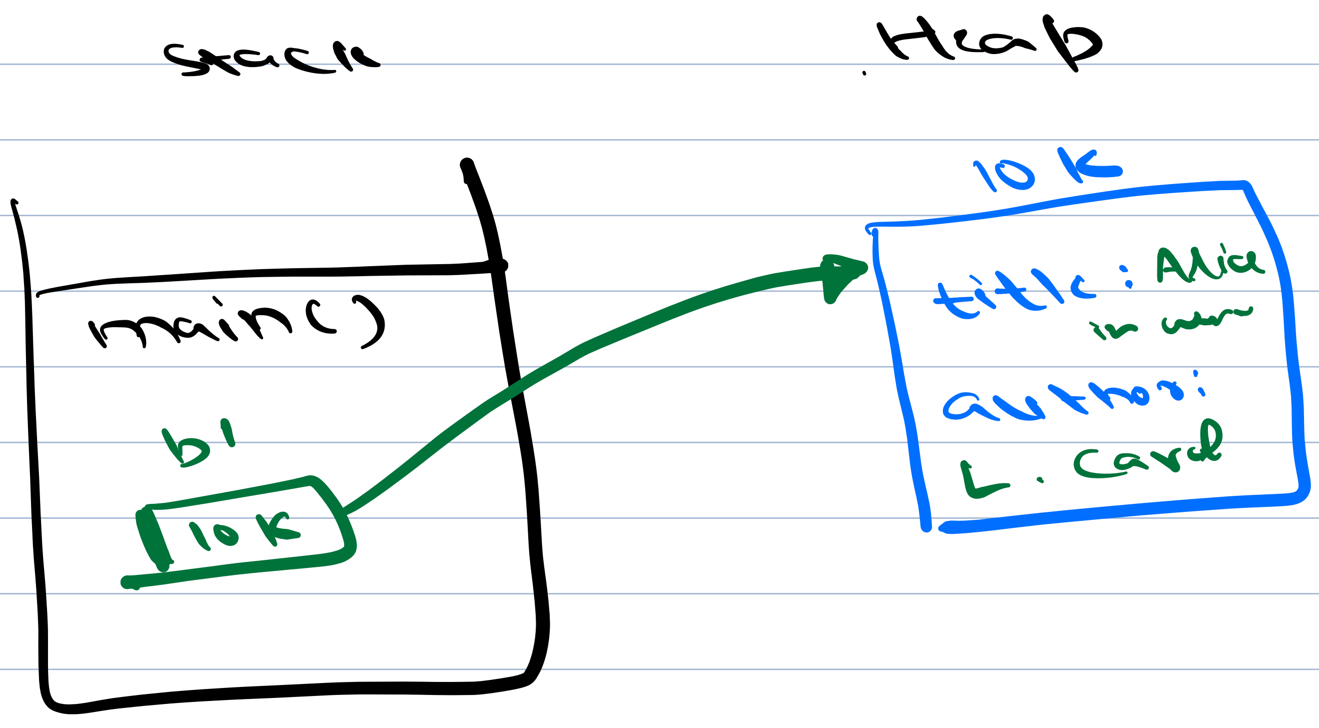
OOPs

Class : In programming, a class is a blueprint for creating objects.
It defines what attributes (properties) and methods (behaviours) the objects created from it will have.

Object : It's a real, tangible instance of class.

```
class Book <
    String title
    String author

    void read () <
        system. out. print ("Reading")
    >

    void flip Page () <
        system. out . print ("Flipping page")
    >
>
```

```
public class Main <
    public static void main ( ) <
        // create object in memory
        Book b1 = new Book()
        b1. title = "Alice in wonder"
        b1. author = "Lewis Carol"
    >
>
```



```
class Book :

    title = " "
    author =""

  def read (self) :
      return "Reading"


python_bl = Book ()


python_bl . title = "ABC"
python_bl . author = "XYZ"
```

Constructor

new Book()

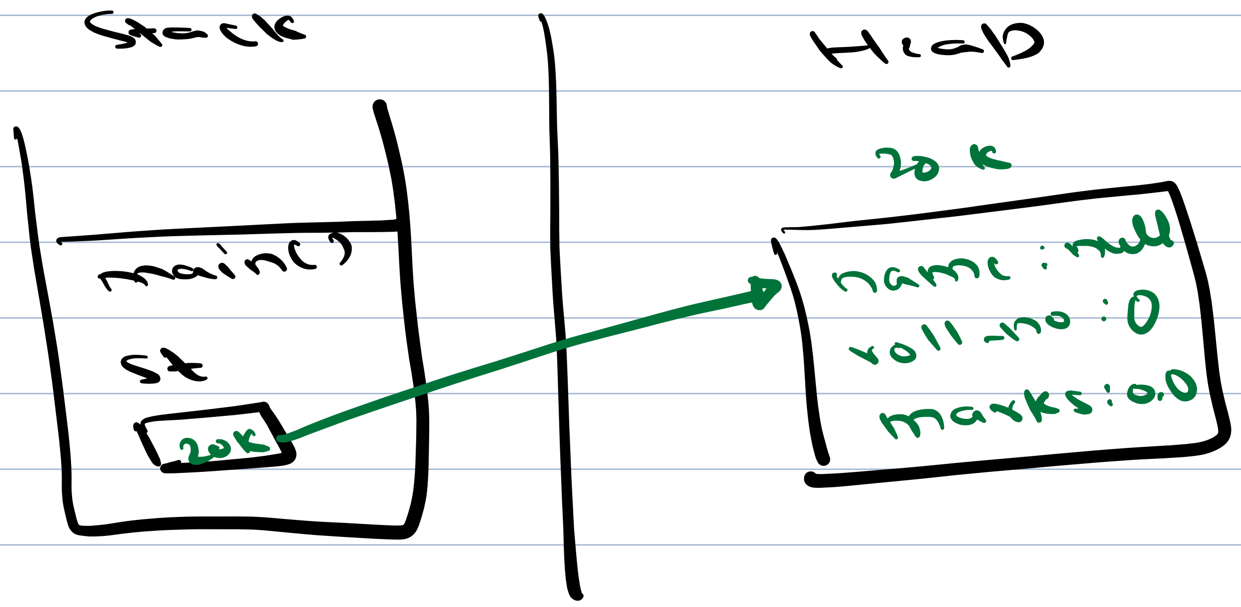
Method which creates object of the class.

Default Constructor

When you don't create your own constructor, a default constructor is created.

① create object
② initializes values of attributy in object

```
class student {
    String name
    int age
    double marks

    // How default constructor
    looks like
    student() {
        name = null
        age = 0
        marks = 0.0
    }
}
student st = new student()
```

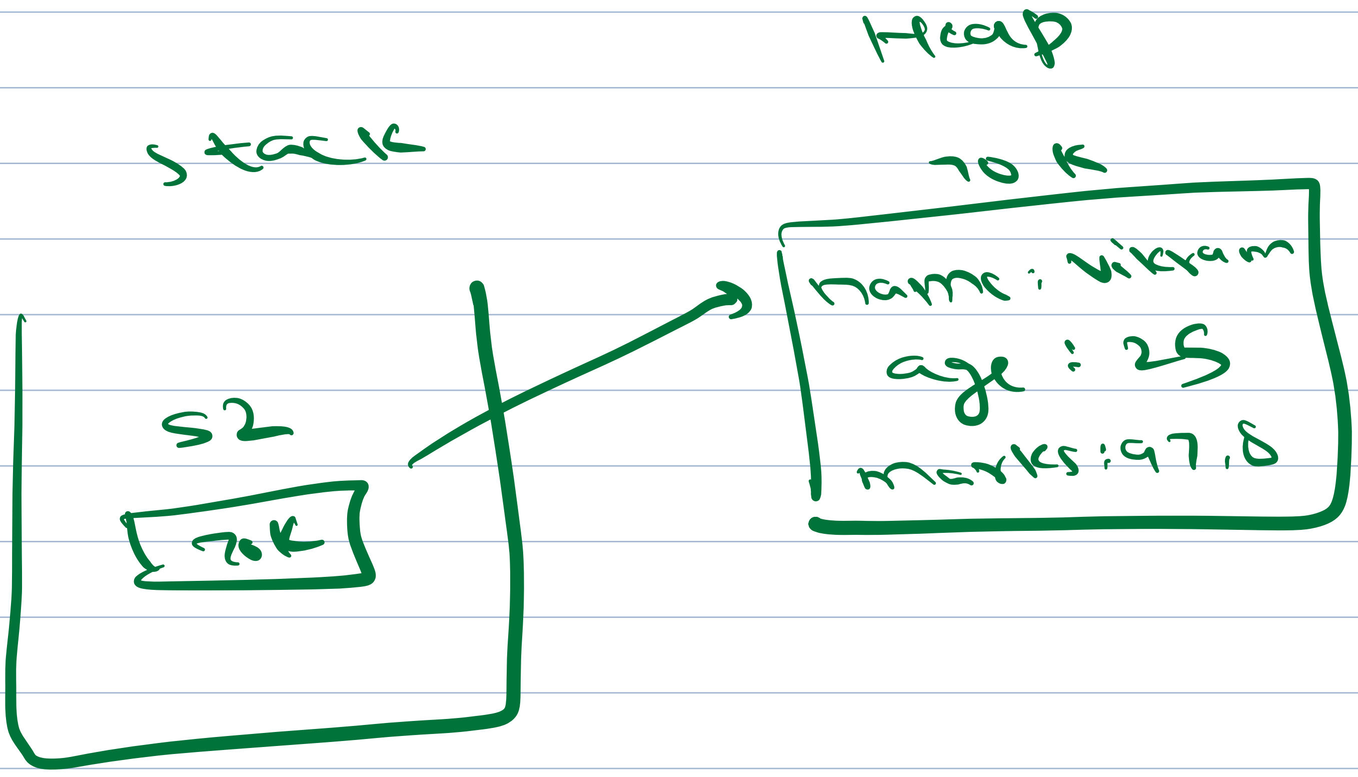


1. Constructor name is same as class name
2. If we create our own constructor, no default constructor is created
3. Default constructor does not take any arguments
4. Its public i.e. can accessed from anywhere.

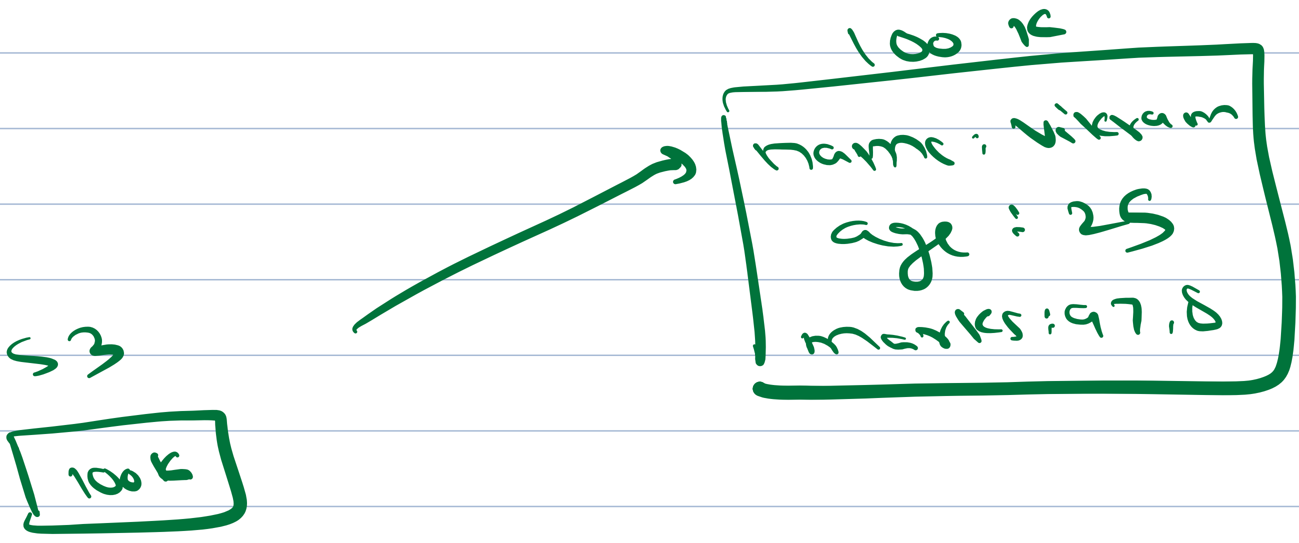
Manual constructor

1. Non-Parametrized constructor

```
class Student {
    String name
    int    age
    double marks

        Student() {
            name = "Vikram"
             age = 25
            marks = 97.8
        }
}

Student s2 = new Student()
```



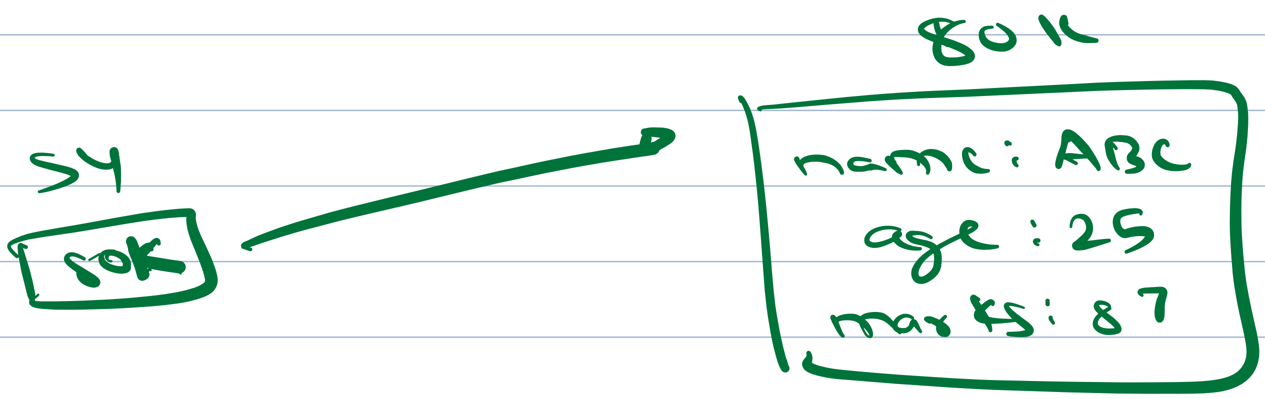
Student s3 = new Student()



② Paramatrized constructor

```
class Student <
|   String name
|   int    age
|   double marks
|
|   Student(String sname, int sage, int smarks)
|   |   name = sname
|   |   age = sage
|   |   marks = smarks
|   ,
,
```

Student s4 = new Student("ABC", 25, 87)



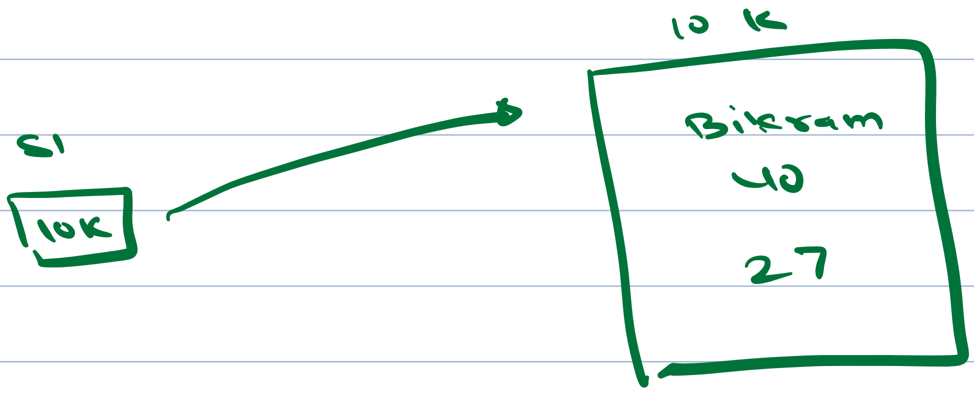
This Keyword (in Java)

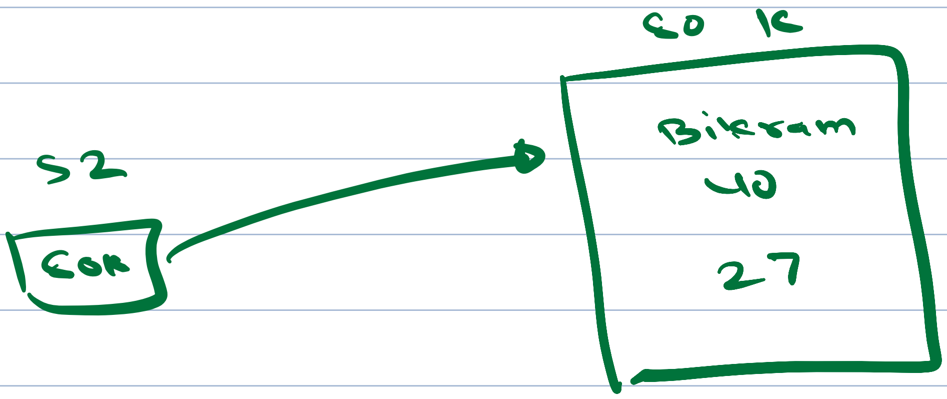
```
class student {
    string name
    int age
    double marks

    student(string name, int age, int marks)
        this.name = name
        this.age = age
        this.marks = marks
    }
}
```

this → current object we're building

this.name → object's attribut

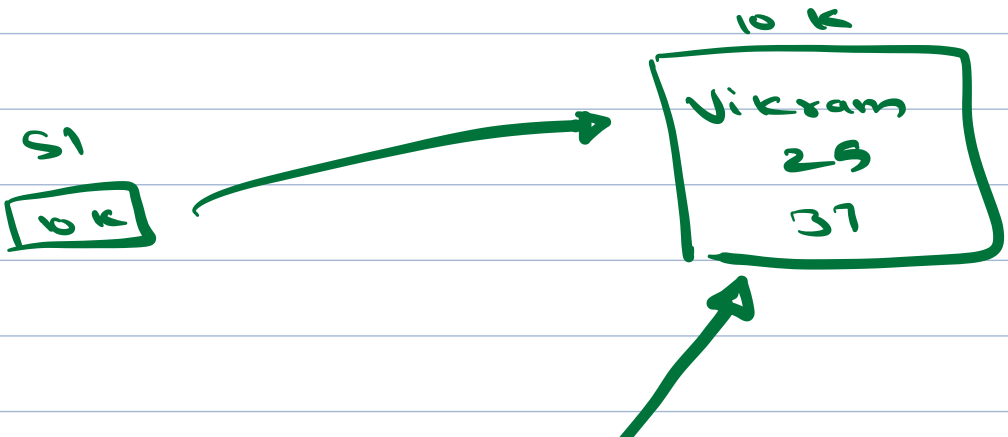
Shallow copy vs Deep copy



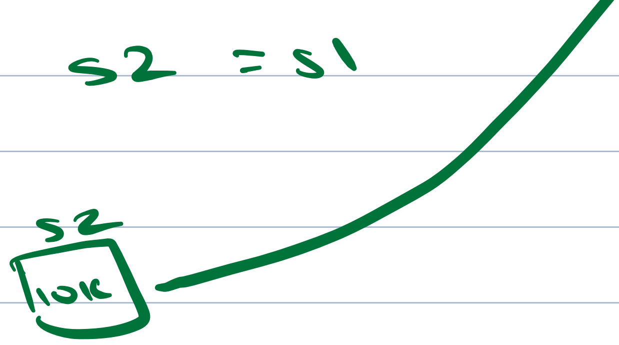


Q. Student s2 = s1 ? ✗

① Student s1 = new Student ("Vikram", 25, 37)



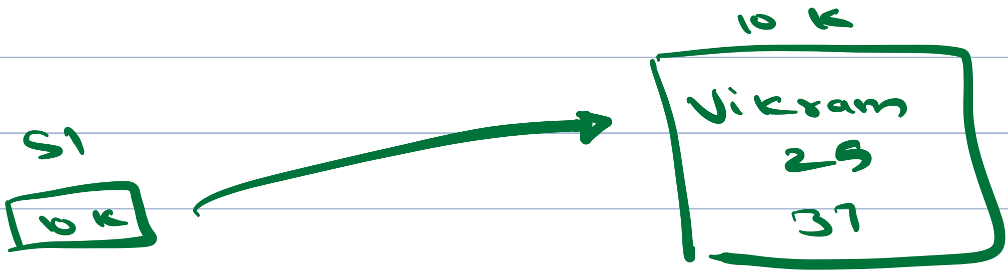
② Student s2 = s1



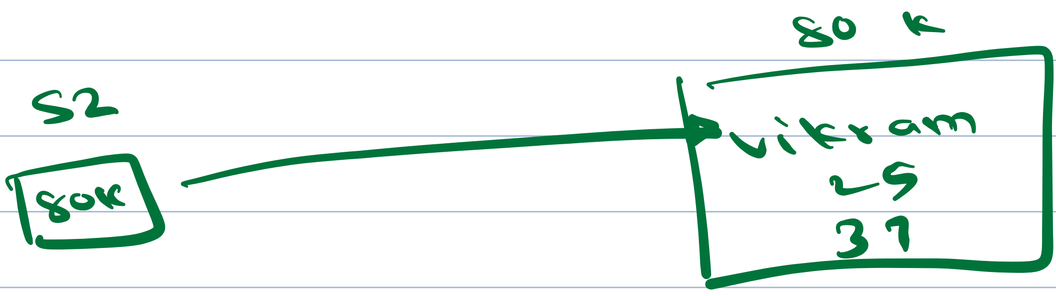
Shallow copy (No new object is created)

Deep copy

① Student s1 = new Student("Vikram", 25, 37)



② Student s2 = new Student("Vikram", 25, 37)



```
Student s2 = new Student
        (s1.name, s1.age, s1.marks)
```

10:21