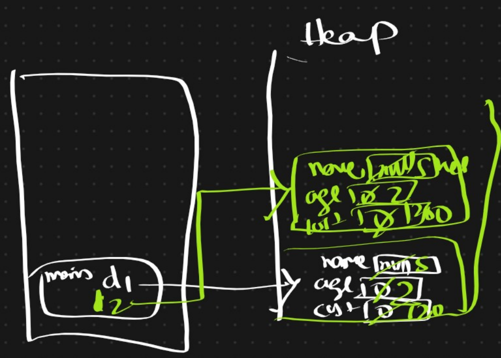
=> Constructor :-

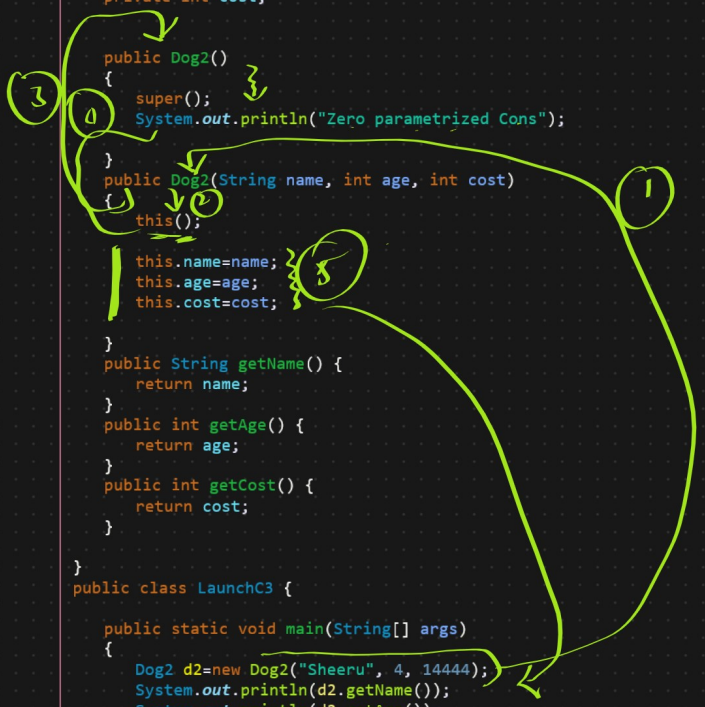
- => class and object => ✓
- => variables, methods => ✓
- => method overloading => ✓
- => Encapsulation :- private members
  - setters
  - getters

```java
class Dog1
{
   private String name;
   private int age;
   private int cost;
   public Dog1()
   {
      System.out.println("Zero parametrized Cons");

   }
   public Dog1(String name, int age, int cost)
   {
      this.name=name;
      this.age=age;
      this.cost=cost;
   }
   public String getName() {
      return name;
   }
   public int getAge() {
      return age;
   }
   public int getCost() {
      return cost;
   }

}
public class LaunchC2 {

   public static void main(String[] args)
   {
      Dog1 d1=new Dog1();
      System.out.println(d1.getName());
      System.out.println(d1.getAge());
      System.out.println(d1.getCost());


      Dog1 d2=new Dog1("Sheeru", 2, 12000);
      System.out.println(d2.getName());
      System.out.println(d2.getAge());
      System.out.println(d2.getCost());

      //d1.disp();



   }

}
```

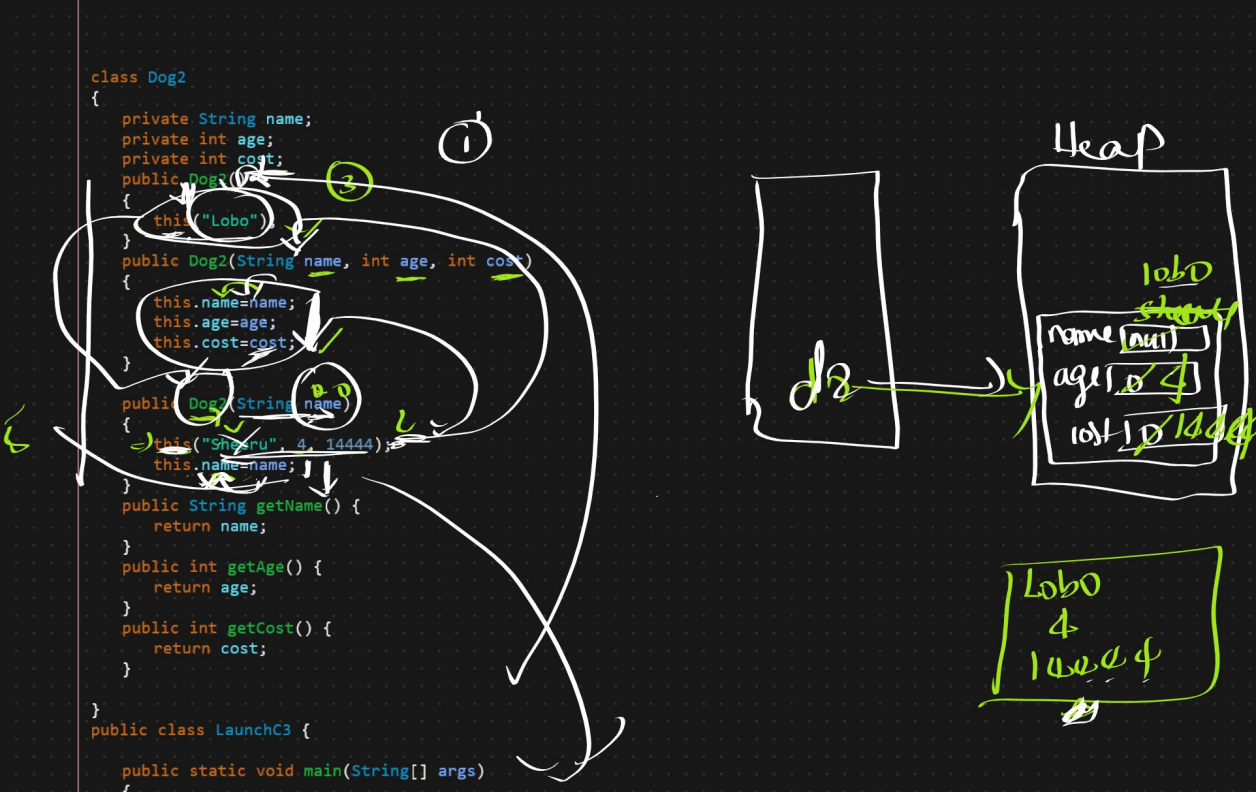


sw
2
12000

```java
class Dog2
{
    private String name;
    private int age;
    private int cost;

    public Dog2()
    {
        super();
        System.out.println("Zero parametrized Cons");

    }
    public Dog2(String name, int age, int cost)
    {
        this();

        this.name=name;
        this.age=age;
        this.cost=cost;

    }
    public String getName() {
        return name;
    }
    public int getAge() {
        return age;
    }
    public int getCost() {
        return cost;
    }

}
public class LaunchC3 {

    public static void main(String[] args)
    {
        Dog2 d2=new Dog2("Sheeru", 4, 14444);
        System.out.println(d2.getName());
        System.out.println(d2.getAge());
        System.out.println(d2.getCost());



    }
```



```java
class Dog2
{
    private String name;
    private int age;
    private int cost;
    public Dog2()
    {
        this("Lobo");
    }
    public Dog2(String name, int age, int cost)
    {
        this.name=name;
        this.age=age;
        this.cost=cost;
    }
    public Dog2(String name)
    {
        this("Sheeru", 4, 14444);
        this.name=name;
    }
    public String getName() {
        return name;
    }
    public int getAge() {
        return age;
    }
    public int getCost() {
        return cost;
    }

}
public class LaunchC3 {

    public static void main(String[] args)
    {
        //Dog2 d2=new Dog2("Sheeru", 4, 14444);
        Dog2 d2=new Dog2();
        System.out.println(d2.getName());
        System.out.println(d2.getAge());
        System.out.println(d2.getCost());


    }
```



=> Constructor & setter, getters &

=> Fields <=> Data members <=> instance var
<=> object var

=> Static keyword

=> static => class (inner class)

=> & variable ->
=> & block ->
=> & method ->

=>

=> Methodoverloading => Y
Main method => Y Y

Yes =>

All fundamentals of Java is over Y

=> From main method -> execution
!),
first one X false