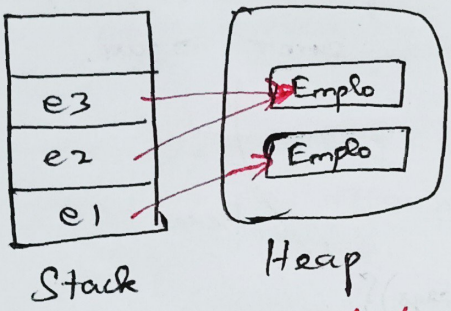
DAY-9

Singleton Class:

Whenever we call constructor new obj will be created.

How do we create obj

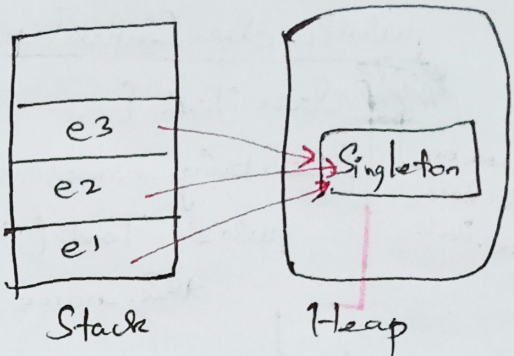
```
classname ref = new Constructor();
Emplo e1 = new Emplo();
Emplo e2 = new Emplo();
Emplo e3 = e2
```



Memory allocated by JVM on RAM

Singleton

Class has only one instance created in heap
No other way to create another.



Memory allocated by JVM on RAM

Eg:

```
class Singleton {
    private static Singleton instance;        // Private to prevent external access
    private Singleton() Constructor           // Prevent direct object creation outside the class
    public static Singleton getInstance(){    // Return the instance, creating it if needed
        if(instance == null){
            instance = new Singleton();
        }
        return instance;
    }
}
```

Why we use:

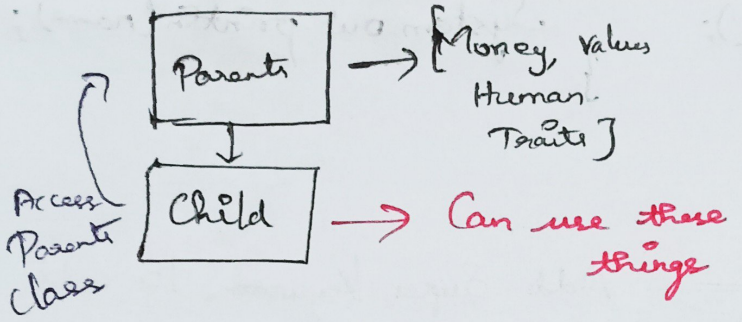
Save memory
Avoid unnecessary obj creation
Provide global point of access for a shared object.

Principle of OOPS Concepts:

Inheritance:

A class that can use the properties of the other class.

Child class can have some values.



Eg:

```
public class Box {
    double l;
    double h;
    double w;
Box Box() {
    this.h = -1;
    this.l = -1;
    this.w = -1;
}
Cube Box(double side){
    this.h = side;
    this.l = side;
    this.w = side;
}
public void info(){
    System.out.println("Run");
}
}
```

Main function();

```
public class Main {
    public static void main
        (String[] args){
        Box box = new Box(side:4);
        System.out.println(box.h+
            " "+box.l+" "+box.w);
    }
}
```

O/p: 4.0 4.0 4.0

Depends on Box( ) value:

If we give three value, it create box.

We can also extend the class

```
public class Boxweight extends Box{
    double weight;
    public Boxweight(){
        this.weight = -1;
    }
```

O/p: -1

Super Keyword: From child class we can access parent class

Eg:

```
class Animal{
    Animal () {
        System.out.println("Animal");   ①
    }
}
class Dog extends Animal {
    Dog () {
        super("Monkey");
        System.out.println("Dog");   ②
    }
}
public class MainClass{
    public static void main (String [] args){
        Dog d1 = new Dog();
    }
}
```

Access Animal

First call child class

⇒

```
Animal (String name) {
    System.out.println(name);
}
```

Parameter value

→ Add Super Keyword to call the Parent parameter value.

When we also add fun to the parent class, we can call in child class by

super.____ function() name;

o/p: Animal
Dog

Ex: Call a Person class Assign name & employee extends person call employee name.

```
class Person{
    String name;
    Person (String name){
        this.name = name;
    } S.o.p ("Person" + this.name);
}
class employee extends Person{
    employee (String name) {
        super (name);
    } S.o.p ("Employee" + super.name);
}
public class Main{
    public static void main (String [] args) {
        employee e1 = new employee ("Priya");
        System.out.println (e1.name);
    }
}
```

Parent Class

Child Class

o/p: Priya

o/p:
Person Priya
Employee Priya
Priya

Print all class

We created obj for child class