Devops - Official document Important

Notes Date:

System Design

OOPS-1
OOPS-2
OOPS-3 / SOLID
LLD → AF, FM, Si, Proxy

SD - blackwasp.co.uk

Agenda

OOP

Classes and Objects (Encapsulation)
- Person
- Reference vs Instance
- Swap 1 - p1 & p2 swap via temp
- Swap 2 - p1 and p2 swap via properties
- Games in swap2 - new p1 and p2

Miscellaneous
- Static and non-static - Bank Account story
    - Static vs non-static data members
    - static funs - no "this", only static data members
    - non-static fns - this, all data members
- final - ~~object~~ variables and data members
- Object creation - steps, this, this chaining
    - space allocation - java Defaults.
    - Parsing - Non dynamic Class Creator defaults
    - Default ctor - Dynamic class Creator defaults
    - Parametrized ctor - class User defaults.

Inheritance and Polymorphism
- P, C - all cases except abstract and final.

PASSION

Notes

Date:

OOPS - 1

Encapsulation

- Classes
- Object

Class bannai jayegi

data member

function

Sath m

rhega

The Process of creating

-> Class k function m object बनाया जाता है

-> Bank Account

-> Student

function
Play
exams

Marks
datamembers

LLD

I OOPS APPLICATION

- -> OOPS Language Basic
- -> SOLID Principal
- -> 23 design patterns

Multi threading

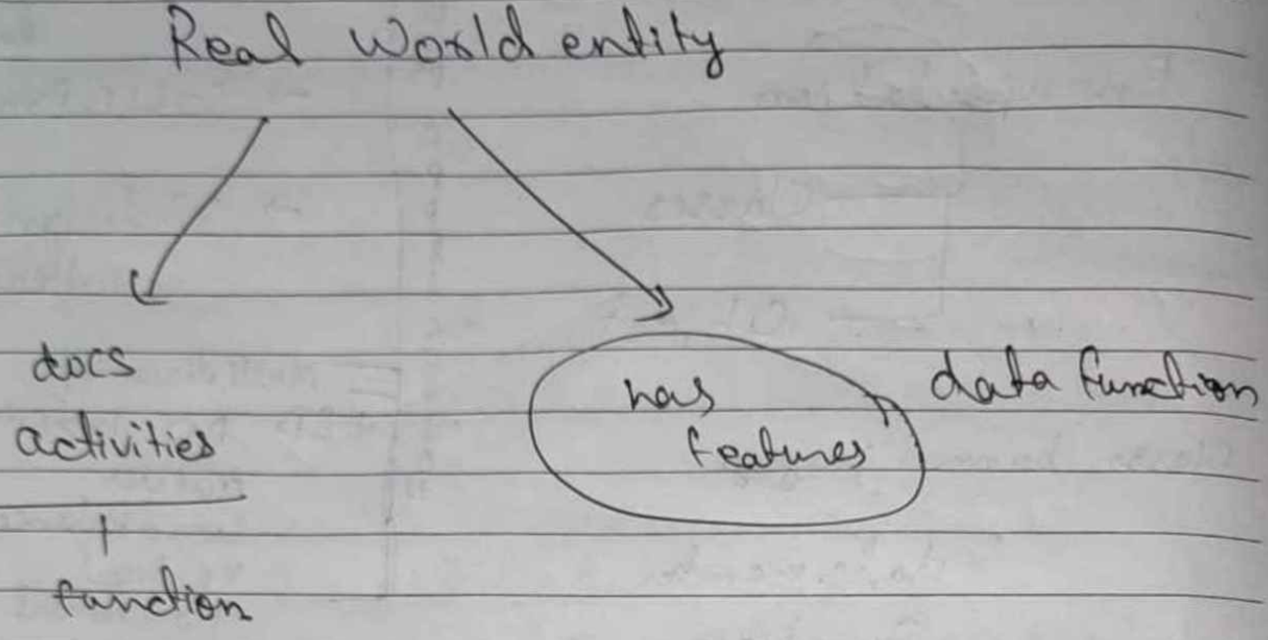
HLD DD Insta design
queues
Load Balance
vertical
Horizontal

Q. Only one class bane

Q Mario wale game
kahi p cloud, tree
every class different
object.
Memory use
fly weight design
pattern.

Q Contra
Every Soldier
1-2 type enemy
Fly-weight
design
Story type
Question will
come.

PASSION



(m Structure help krta Encapsulation hota but Interitance
OOP's K phle ka language m variable declare idhar udhar rehta tha
confusion create hota.
OOP's - Better Structure provide krta real world
m lana k liya.

Person

int age
S name

Code
Class

```
importjava.io.*
import java.util.*;

class Main {
  Static class Person {
    int age;
    String name;
```

Instance = Object

Notes

Date: Mandatory nhi h, (Good Practiceh)

Object Ko refer Krtah

```
void SayHi(){
    Sytem.out.println(this.name + "[" + this.age+
        "] sayshi.");
    }
}
```

IOException {

Exception,

NumberFormat,

```
public static void main(String[] args) throw

Person p1 = new Person();
 p1.age = 10;
 p1.name = ("A");
 p1.SayHi();

Person p2 = new Person();
 p2.age = 20;
 p2.name = "B";
 p2.SayHi();

Person p3 = p2;
 p3.SayHi();

 p2.name = "C";
 p2.SayHi();
 p3.SayHI();

 }
}
```

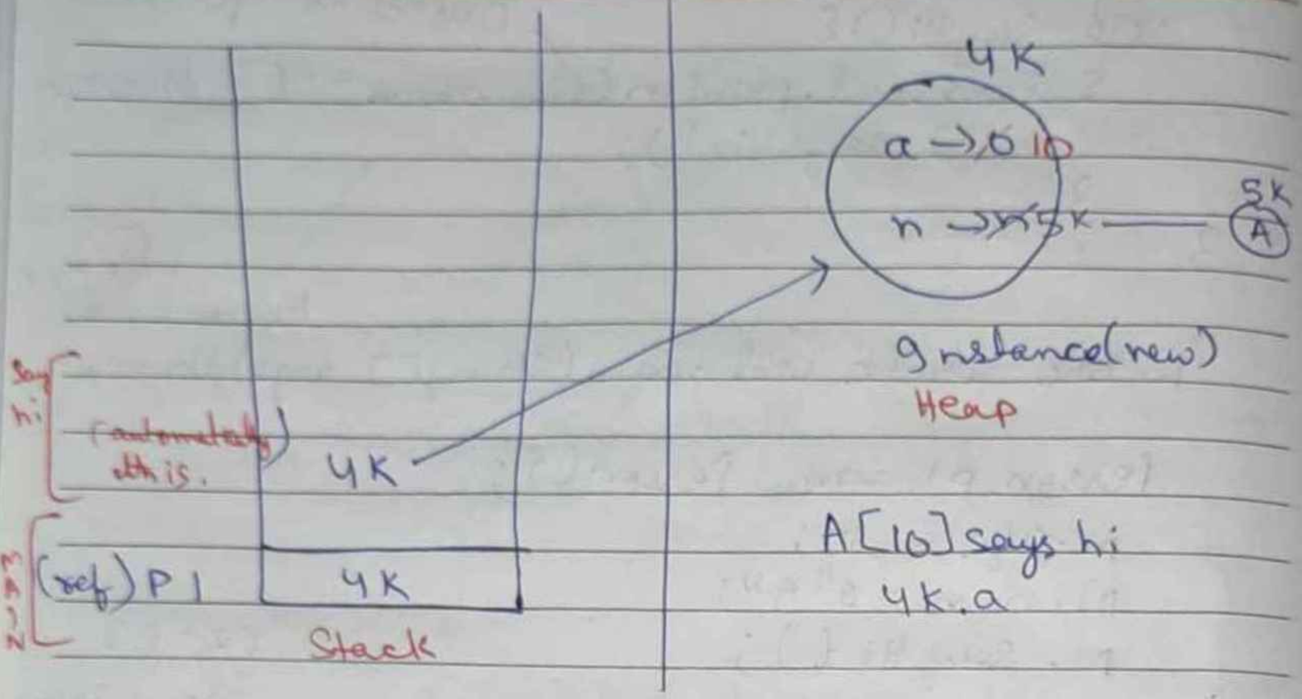
P2 = P3

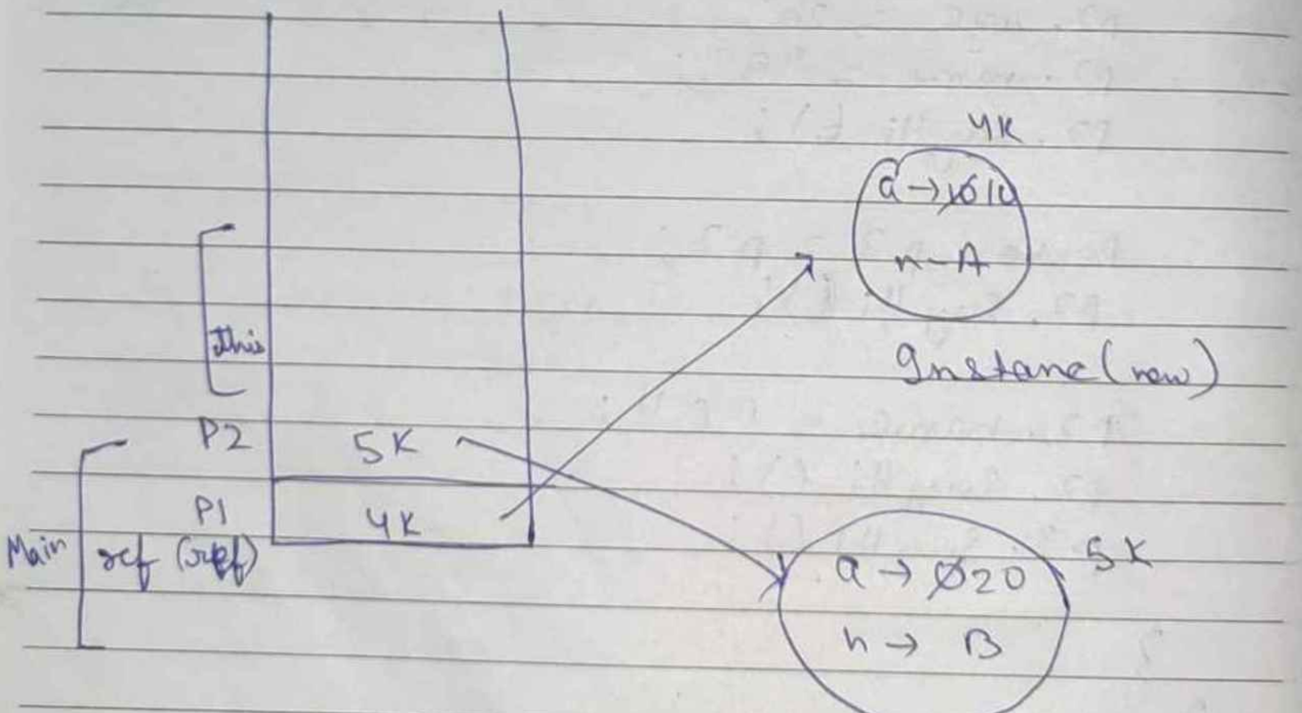
Dono Same h memory m

PASSION

Gnstance variable Age, name

Notes Date: □ □ □





P1 ek object ka reference h

P2 P3 Same object ka reference h.

this → Address h → refrence

why program start from main function? → Main ~~j~~entry point h designated

Scope m shta h

How does the program start executing?

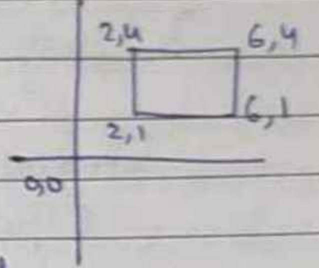
Java c — Compile kr deya

java Main . Class
  └ isse find krtah.

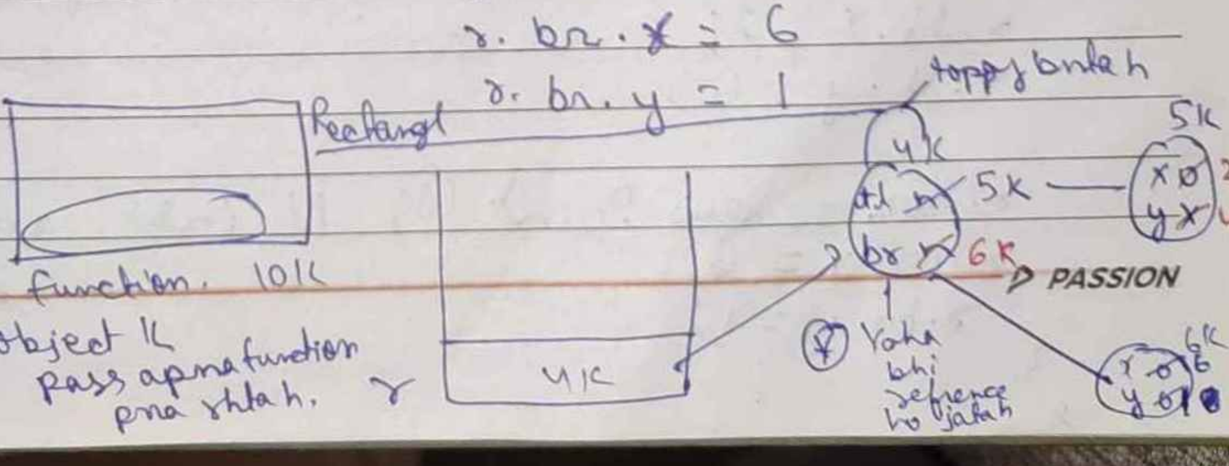
Eclipse m puchta h Two multiple main function ho to.
  └ Puchta kaunsfile run kre

Command k waqt hum khud rkhtah.

Point	Rectangle
i x	Point tl (top left)
i y	Point br (bottom right)



```
Rectangl r = newrectangle
  r. tl = new point()
  r. tl .x = 2
  r. tl . y = 4

  r. br = new point
  r. br . ~~x~~ = 6
  r. br . y = 1
```



kn object k pass apna function pna shta h.

(*) Yaha bhi refrence ho jatah

Static Global - nearly Same baath

C++ Stack m ho object ban jatah.

Notes

Date: □□□

Code

```
import java.util.*;

class Main{

  static class Point {

    int x;
    int y;

  }

  Static class Rectangle {
        point tl;
        point br;
  }

  public static void main (String [] args) throws NumberFormatException,

  Rectangle r = new Rectangle (); // instance, ref

    r.tl = new Point (); // inst, ref
    r.tl.x = 2;
    r.tl.y = 6;

    r.br = new Point (); // inst, ref
    r.br.x = 6;
     r.br.y = 1;
  }
}
```

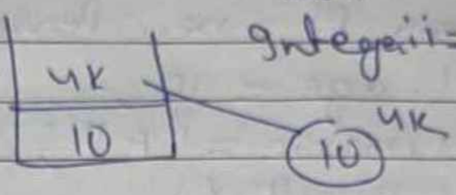
PASSION

Frontend - HLD system design
backend - LLD

Object = Instance

Primitive
int a

int i = 10 — faster
Integer i =



main function k baad such whi shla
Garbage Collector ata sab ko collect krleta h.

Register ⇒ CPU Register is One of a small set of data holding set of data holding places that are part of the computer processor. A register is ~~one of a small set of data holding~~ may hold an instruction, a storage ~~class~~ address or any kind of data.

IOException {

* ek class k under bohot sare class bana skte h.

int a, int b, - normal primitive data type. (variable).

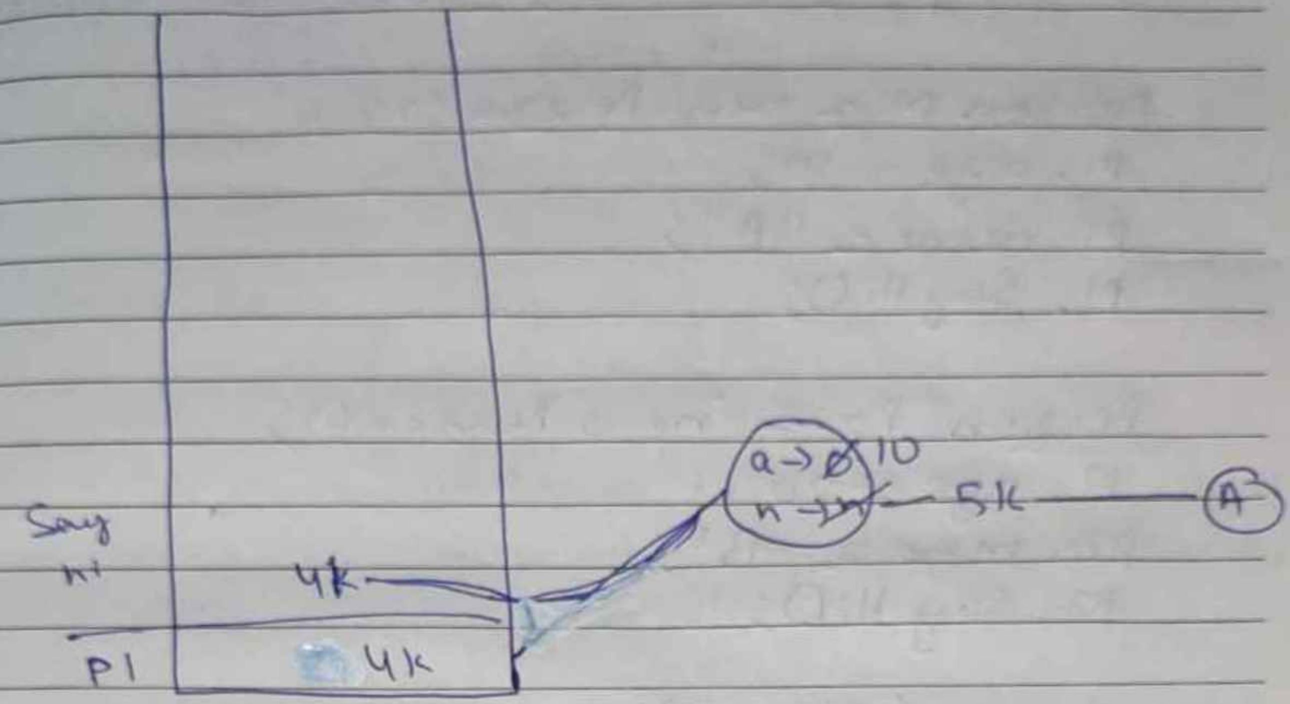
new - word - reference

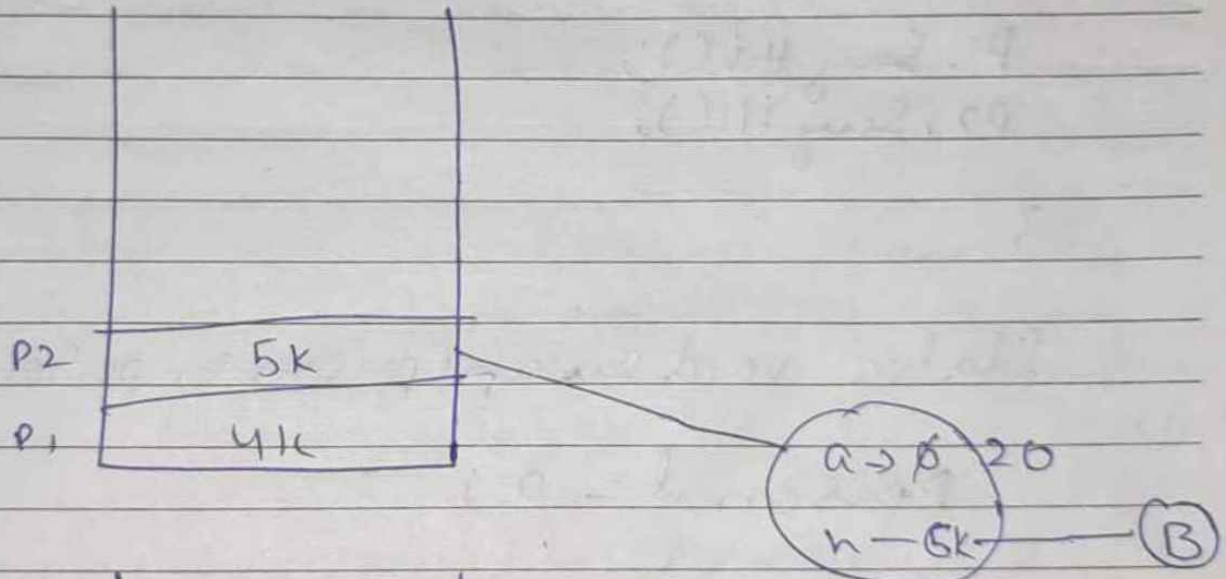
Understanding SWAP

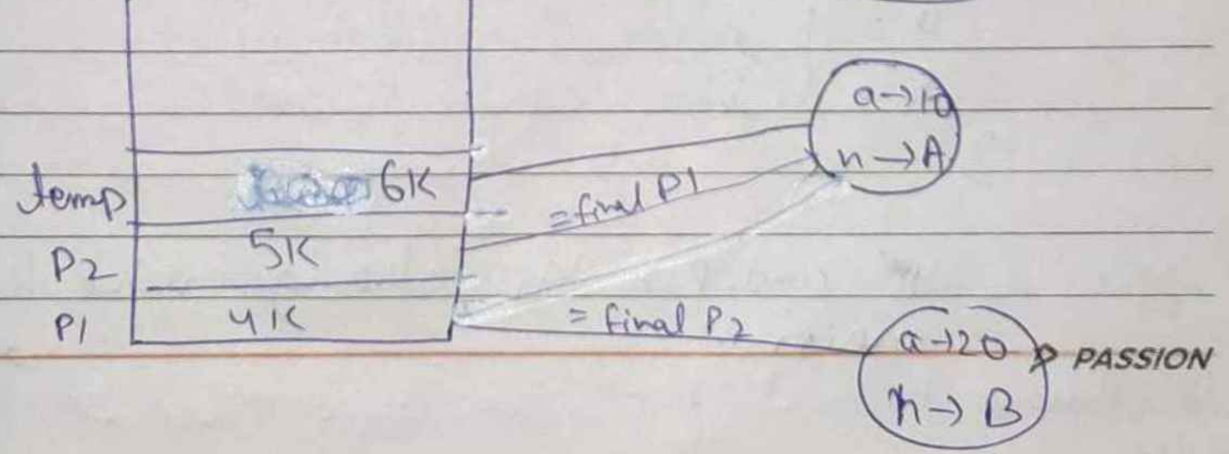
```
Person P1 = new Person();
    P1.age = 10;
    P1.name = "A";
    P1.SayHi();

  Person P2 = new Person();
     P2.age = 20;
     P2.name = "B";
     P2.SayHi();

  Person temp = P1;
     P1 = P2;
    P2 = temp;

  P1.SayHi();
  P2.SayHi();
```

Output:-

```
A[10]
B[20]

A[20]
B[10]
```


Say
n1
4k
P1
4k
a → ~~0~~ 10
n → ~~n~~ — 5k
A


P2
5k
P1
4k
a → ~~0~~ 20
n — 6k
B


temp
6K
P2
5K
P1
4K
= final P1
= final P2
a → 10
n → A
a → 20
n → B

If swap function available what happened?

```
Person p1 = new Person();
 p1. age = 10;
 p1. name = "A";
 p1. SayHi();

Person P2 = new Person();
 p2. age = 20;
 p2. name = "B";
 p2. SayHi();

 Swap (P1, p2);

  P1. SayHi();
  P2, SayHi();

}

Static void swap(person a, person b){

  Person t = a;

   a = b;
   b = t;
  }
```

p1 and P2 ko change nhi krskta maim m

String - Immutable hota h Refrences - mutable hota h

Swap function chalega $a = 5k$, $b = 4k$ but
(a, b) alag refrences tha
wo effect nhi krenga p1 and P2 ko

Dependency Injection hoga
In Software engineering
It is a design pattern in which an object receives other objects that it depends on.

*

```
Staticvoid swap (Person a, Person b){
    int tage = a.age;
    a.age = b.age;
    b.age = tage;

String tname = a.name;
    a.name = b.name
    b.name = a.name;
}
```

heap m change krwe rha h

Av Swap hoga

Arrey ki value swap hoti h pura nhi hoti h

Class swap krtda hota h

Call by reference Stack m change hota h

Only Call by value works in Java

Static and non-static

Static Data Member Class ko belong krta h,

```
import java.io.*;
import java.util.*;

class Main {

static class Bank Account {

  Int anam;

  int money;

  Static int roi;

  void announce() {

    System.out.point ln(this anamt ", "+
                this.money + ", " + this.roi);

public Static

Bank Account a1 = new Bank Account();
```

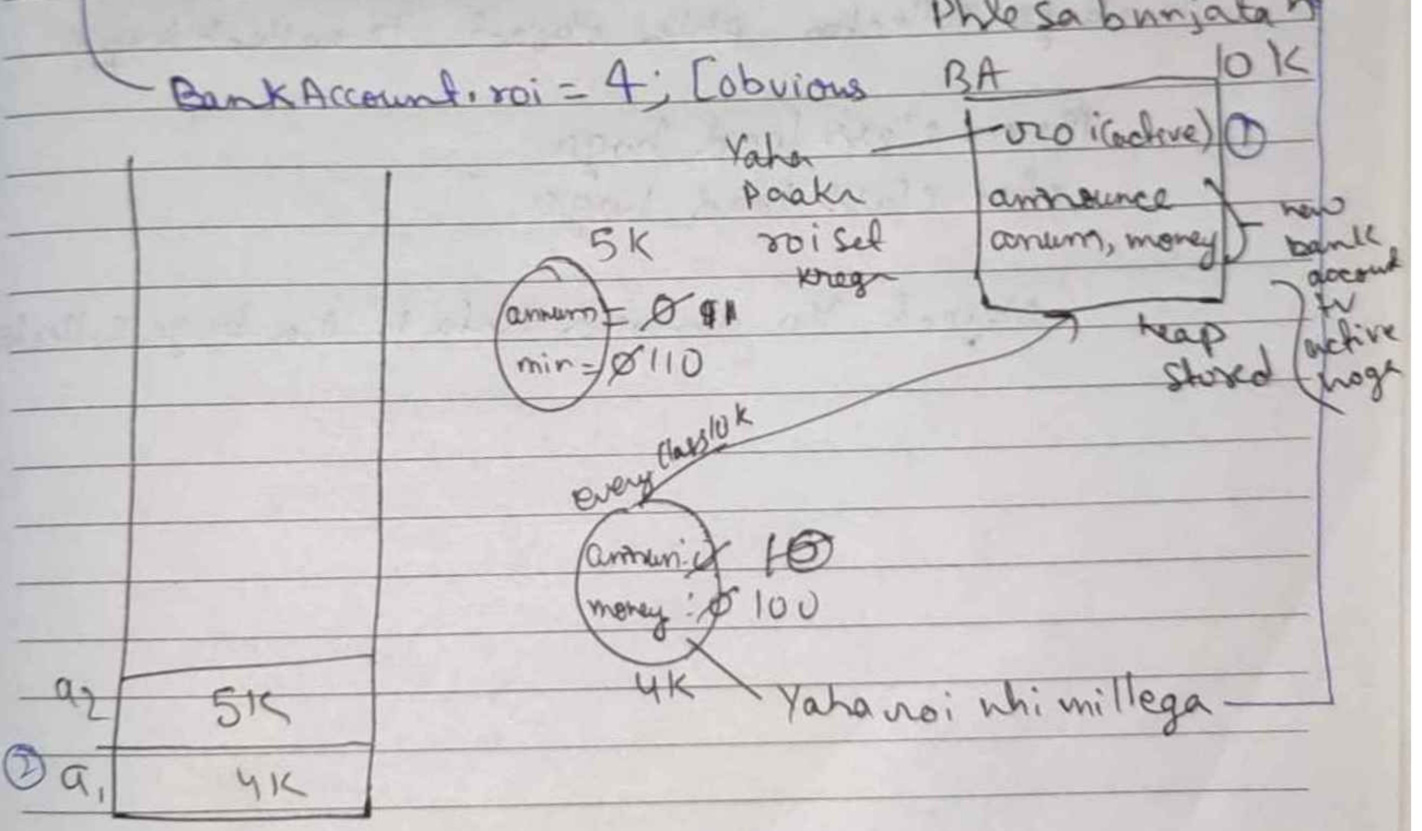
a1. anum = 10;
a1. money = 100;
a1. roi = 4;   worst

a1. announce ();

Bankaccount a2 = new Bankaccount();
a2. anum = 11;
a2. money = 110;
a2. roi = 5;   worst

a2. announce ();
a1. announce ();

BankAccount. roi = 4; [obvious



Static data member ko object se change kr skte h ?

Haa kr skta h

But Bad Practice h

a2. roi = 5 (Worst)
Bank account. roi = 4 [Obvious]

static se change krenge.

Garbage collector phle object to collect kregge

Main class load hoga
other class load hoga

object ko unload krta h Garbage Collector