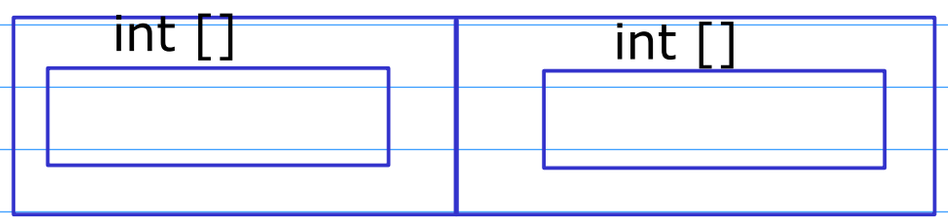
Array

int arr[]; // reference
new int[5];
arr[0]-> value

Employee [] arr = new Employee[5];

arr[0] -> employee reference

MultiDimensional Array
int arr[2][3]

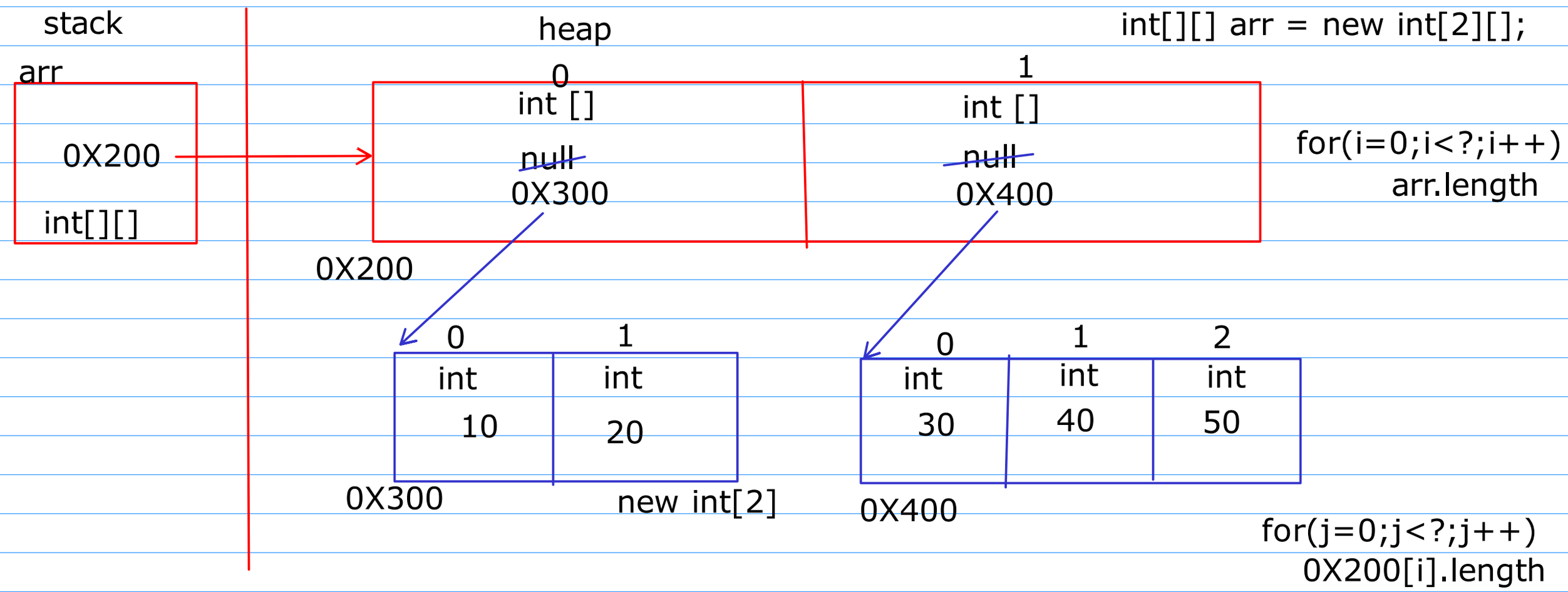


int arr[][] = new int[2][3];

array of references (int[])

#Ragged Arrayx

- It is a multi dimensional array where the size of the inner arrays are different



arr[0] = new int[2]

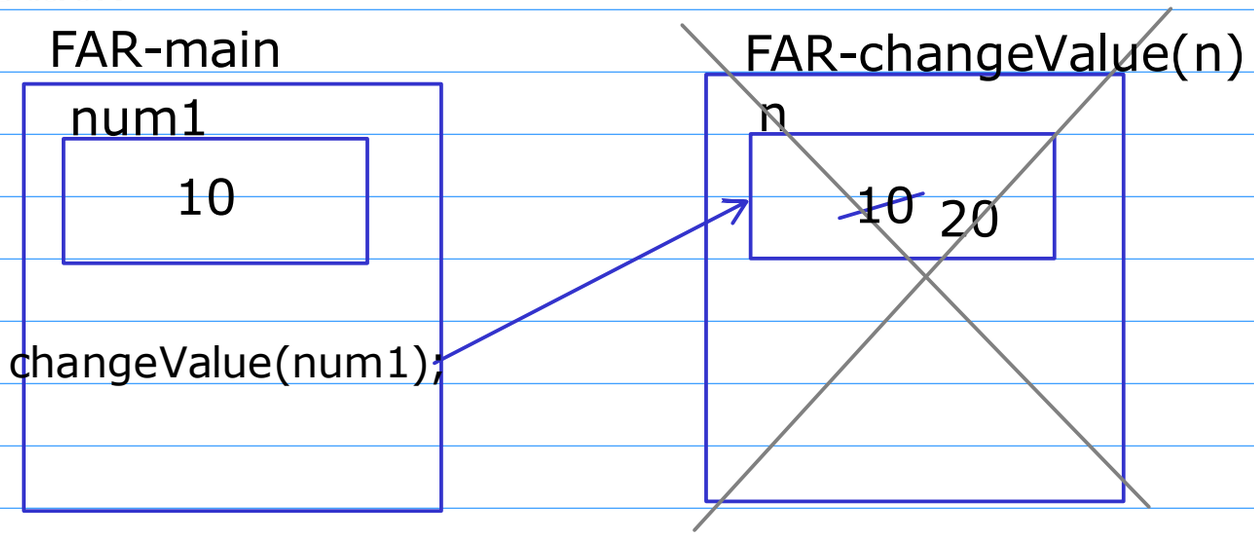
arr[1]=new int[3]

0X300[0]=10
0X200[0][0]=10
arr[0][0]=10

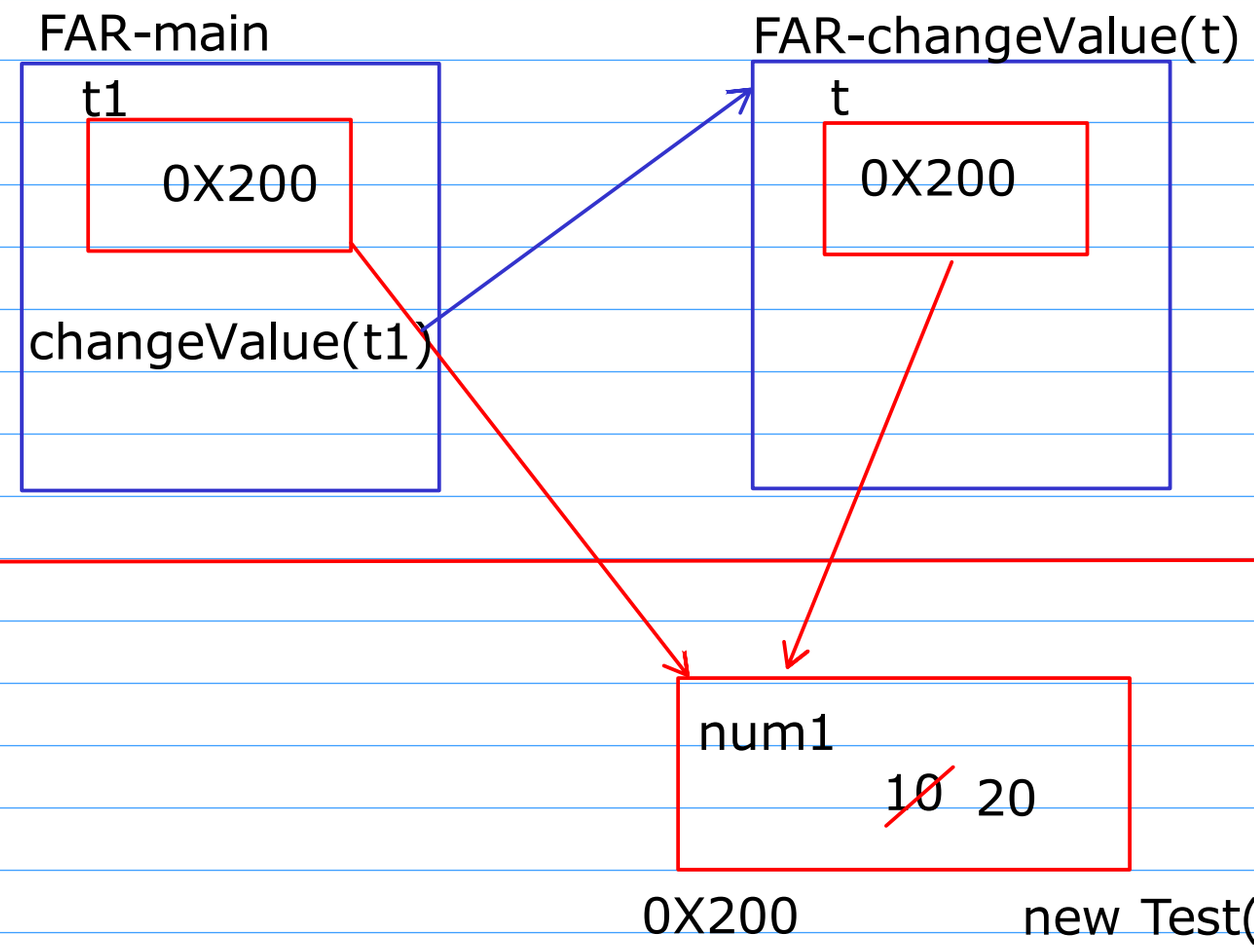
Modular batch -> java(100),dsa(150),genai(200)

Student[][] studentList = new Student[n][];
studentList[0] -> new Student[2]; // java
studentList[1] -> new Student[5]; // dsa
studentList[2] -> new Student[3]; // gen ai

Stack



stack



Heap

1. field Initializer
2. Object Initializer
3. Ctor

Rules
- Field- Object initializer-> ctor

#Final
- in java we can declared
  - Variable
    - We can declare the local variables as final.
    - These final variables once initialized we cannot change the value inside it (cannot be assigned)
    - however in java we ca n declare the variable as final and then later we can initialize it as per the requirement
  - Field
    - We can declare the field as final
    - These fields needs to be initialized in the Field initializer or the object initializer or ctor.
  - Method
  - Class
  - (method and class as final will see after inheritance)

```
class Circle{

double PI = 3.14


}
```

```
class DbUtil{
Connection connection;
PreparedStatemt
{
Sql
connection = .....
}

}
```

```
class BankAcount{
int accno;
String name;
double balance;

{
accno = logic to increment automatically
}

BankAccount(String name,double balance)
}
```

```
public void method(){
      final int num1=10;
}
```

stack

num1

10

0X200

```
changeValue(int n){

}
```

n

20

a1

accno 1001

name Anil

balance 10000

a2

accno 1002

name Mukesh

balance 20000

```
a1.accno = 1003;
a1.balance - 15000;
```

constant -> final -> immutable