Types of Methods

- 1. ctor
 - two types of ctor
 - 1. default/paramaterless
 - 2. parameterized

w/o any arguments

by passing arguments

Constructor Chaining

- Calling another ctor from the current ctor is called as ctor chaining

```
File Edit Source Refactor Source Navigate Search Project Run Window Heli
private int hr;
  5
       private int min;
  6
       public Time() {
  8
           this (10, 10); // this statement
  9
                        Constructor Chaining
 10
       public Time(int hr, int min) {
 11∘
            this.hr = hr;
 13
            this.min = min;
 15
       public void displayTime() {
            System.out.println("Time - " + hr + " : " + min);
 19
 20 }
 22 public class Program01 {
       public static void main(String[] args) {
```

Types of methods

- Ctor
- setter
- getter
- facilitator

- Setter

- a method used to set/change the value of individual field of the class.
- the name should start with set followed by the name of the field to manipulate
- Its industry recommended practice.
- it should accept the value only through the parameter. i.e it shlould accept only 1 paramter of the same type as that of the field

- getter

- a method used to get/read the value of individual field of the class.
- the name should start with get followed by the name of the field to read
- Its industry recommended practice.
- It should compulsary return the value of that field

Q. Why to use setter and getters?

- To make sure the valid data is entered in the fields we make the fields as private.
- to provide the access of the fields through methods.
- these methods will help to add the validation part before assigning the value to the fields.
- It will also make sure to provide the read/ write permission of the fields

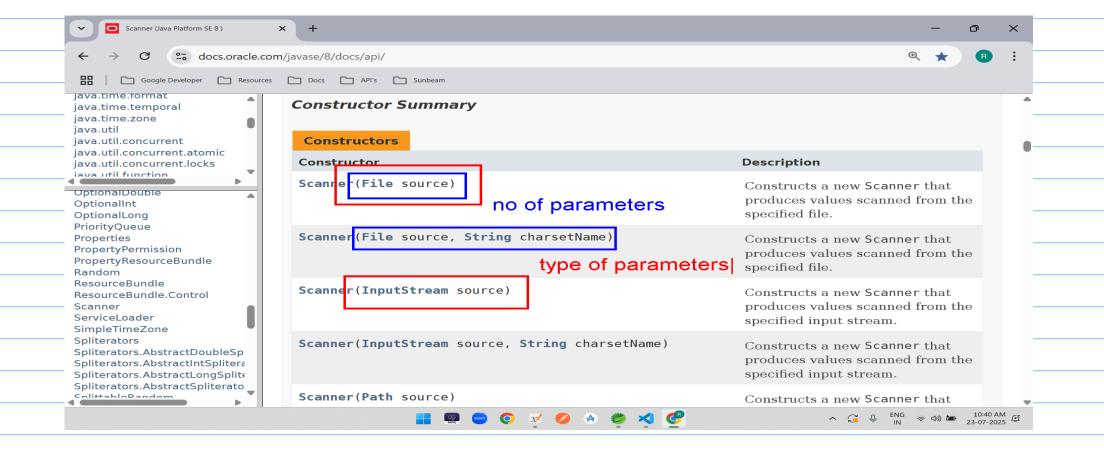
```
class Program{
    public void sort(int arr[]){
        // logic
    }

    private void swap(int n1, int n2){
    }
}

Program p1 = new Program();
p1.sort();
```

Method overloading

- It is a way of defining multiple methods with same name but different signature
- Differnt signaturfe means:
 - change no of paramaters
 - change type of parameters
 - change the order of paramaters
- It is an example of Compile Time Polymorphism



name mangling

- it a process used to assign the unique name to the methods based on their name and signature

```
9 46
       public void add(int n1, int n2)// no of paramates - 2 add i i←mangled name
28
29
                                                                                         47
           System.out.println("Addition - " + (n1 + n2));
                                                                                                public void division(int n, double d) // order of type parameters - int,doubl
 30
                                                                                         48
 31
                                                                                         49
 32
                                                                                         50
                                                                                                    System.out.println("Division - " + (n / d));
 33
       public void add(int n1, int n2, int n3) // no of paramates - 3 add | | |
                                                                                         51
 34
                                                                                         52
           System.out.println("Addition - " + (n1 + n2));
 35
                                                                                                public void division(double(n, int(d))// order of type paramaters - double , if
                                                                                         53
 36
                                                                                         54
 37
                                                                                         55
                                                                                                    System.out.println("Division - " + (n / d)); division d
       public void sqaure(int n1) // type of n1 - int square i
 38
                                                                                         56
 39
                                                                                         57
 40
           System.out.println("Square - " + (n1 * n1));
                                                                                                public void division (double (d) int (n)// order of type paramaters - double
                                                                                         <sub>4</sub>58
 41
                                                                                         59
                                                                                                    System.out.println("Division - " + (n / d)); division d i
 42
                                                                                         60
       public void sqaure (double n1) // type of n1 - double square d
 43
                                                                                         61
 44
                                                                                         62 }
                                                                                                  Method overloading is not possible by change in the paramater names
           System.out.println("Square - " + (n1 * n1));
 45
                                                                                         64 public It can be done only by changing order of type of parametrs
 46
       public void division(int n, double d) // order of type parame
48
                                                                                                public static void main(String[] args) {
```

Array

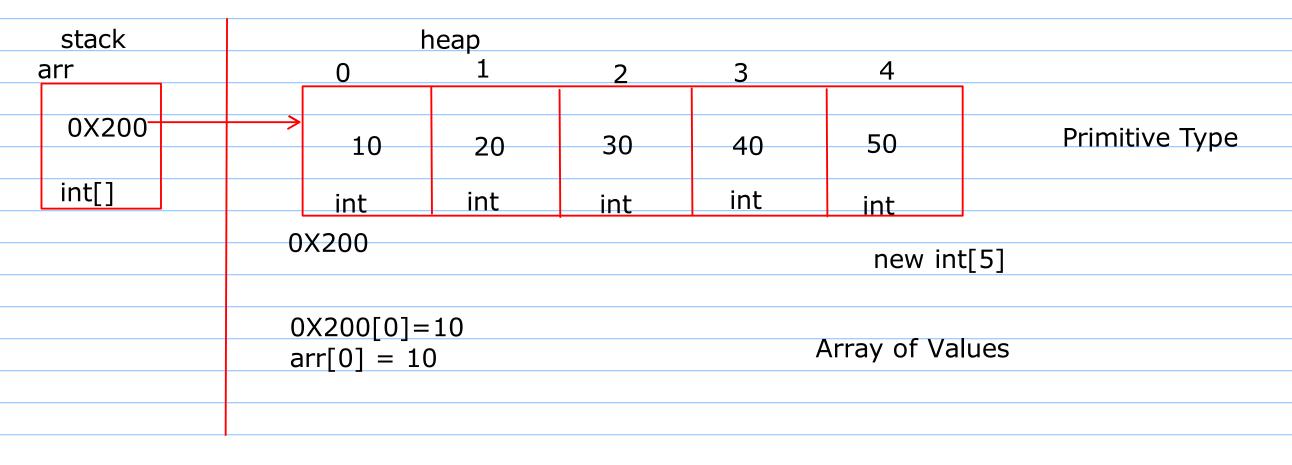
- It is a data structure that is used to store smiliar type of elements in contigious memory loaction
- It is of fixed size
- array provides index based operation
- In java array is a reference type

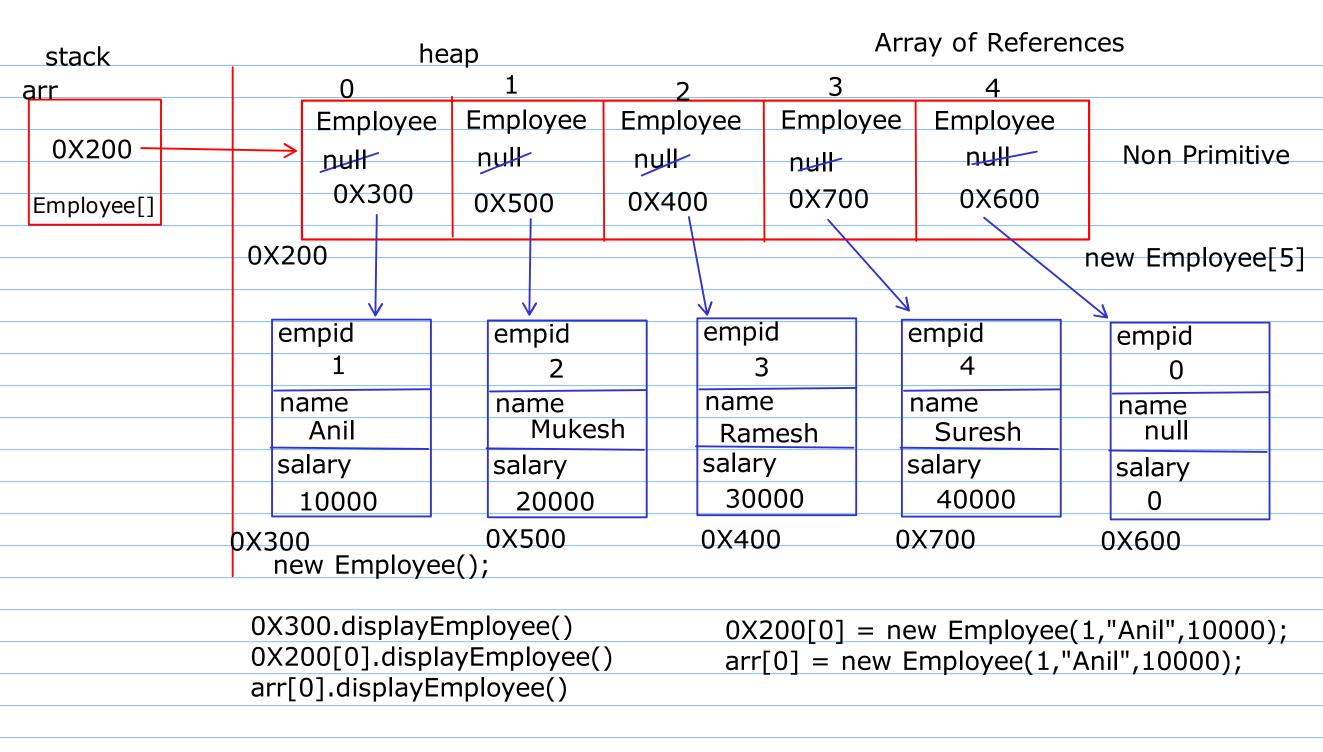
	0	1	2	3	4	
int arr[5];						
	10	20	30	40	50	20 6.4
		20		. 0		20 bytes
	int	int	int	int	int	
0X200						
arr						
<u></u>						

$$0X200[0] = 10$$
 $0X200[1] = 20$ $arr[0] = 10$ $arr[1]=20$

Array in JAVA

int [] arr = new int[5];

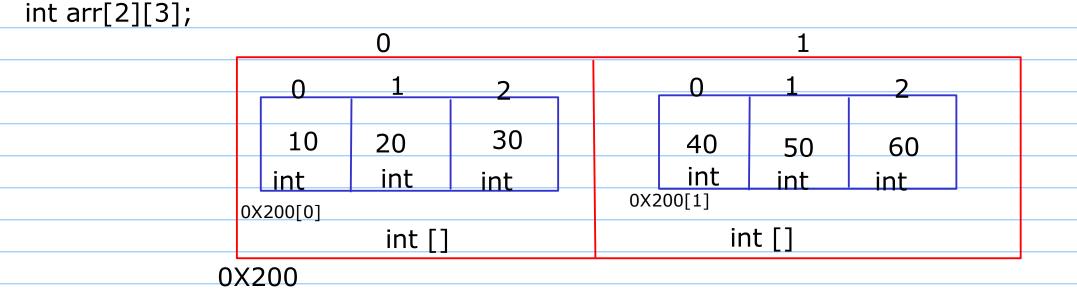




Types of Array

- 1. Single Dimensional Array
- 2. Multi Dimensional Array
- 3. Ragged Array

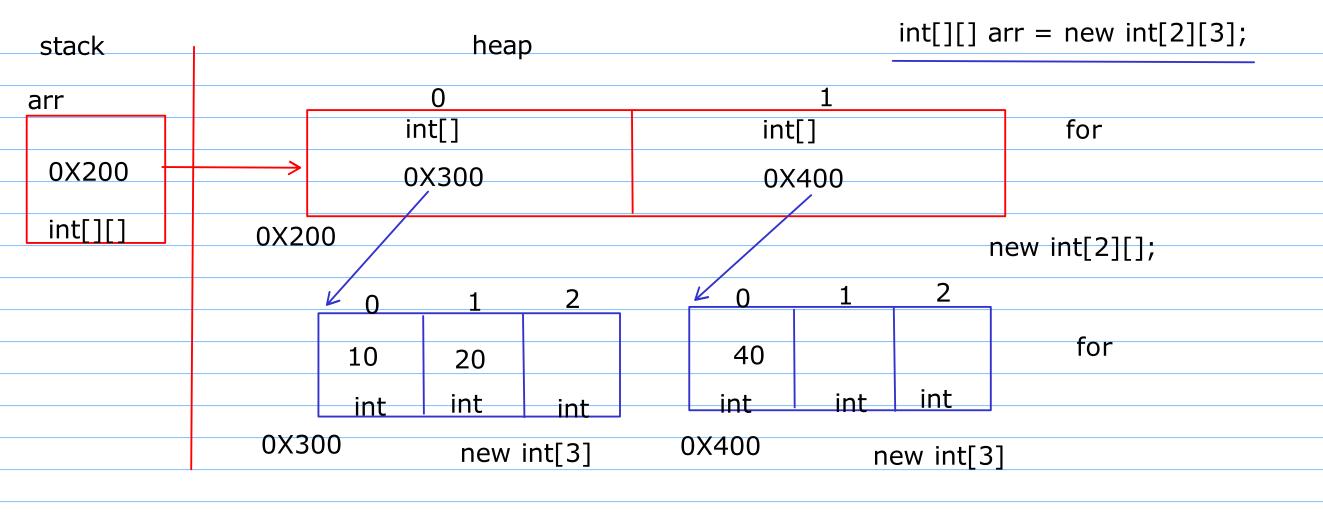
int arr[2]; // single dimensional array
int arr[][]; // multi dimensional array (2d)
int arr[][][]; // multi dimensional array (3d)



0X200 -> Base address of outer array 0X200[0] -> base address of inner array at index 0 0X200[1] -> base address of inner array at index 1

arr

0X200[0][0] = 10arr[0][0] = 10 0X200[1][0] = 40 arr[1][0] = 40



$$0X300[0] = 10$$

 $0X200[0][0] = 10$
 $arr[0][0] = 10$

$$0X400[0] = 40$$

 $0X200[1][0] = 40$
 $arr[1][0] = 40$