Date
Page

INTRODUCTION TO ARRAYS AND ARRAYLIST IN JAVA.

★ Why do we need Arrays?

- Arrays let you store multiple values of same type (like integer, string, etc) in a single variable.
- Each element has index, allowing direct acces
- stored in continous memory ~~bloot~~ blocks, making them memory efficient?

★ Array :- Array is a data structure used to store a collection of data of same datatypes in continous memory.

★ Syantax :-

- 1st way :-

```
datatype[] variableName = new datatype[size].
```

Eg:-
```
int [] Rollno = new int [3];
       Roll no [0] = 11;
       Rollno [1] = 21;
       Rollno [2] = 31;
```

- 2nd way :-
  datatype [] variable_name = {asign value}.

  eg:- int[] rollNo = {11, 21, 31};

Notes:-

- $$\overbrace{\underbrace{\text{int [ ]}}_{\text{datatypes}}\ \underbrace{\text{arr}}_{\text{reference variable}}}^{\text{at compile time}} = \overbrace{\underbrace{\text{new int [size]}}_{\text{creating object in heap memory}}}^{\text{runtime}};$$

  i.e. this is called dynamic memory allocation

- Dynamic Memory Allocation :- It is the process of allocating memory at runtime (while the program is running) instead of compile time.

- Internal working of an array :-

- int [] arr :- // decleration of array
  an arr getting defined in stack

- arr = new int [5] :- // Initialisation
  ~~acto~~ actual memory allocation happens.
  Object being created in heap memory.

- new :- it is used to create an object.
  it will create object in heap memory of array size 5.

Internal Repersentation of an array :-

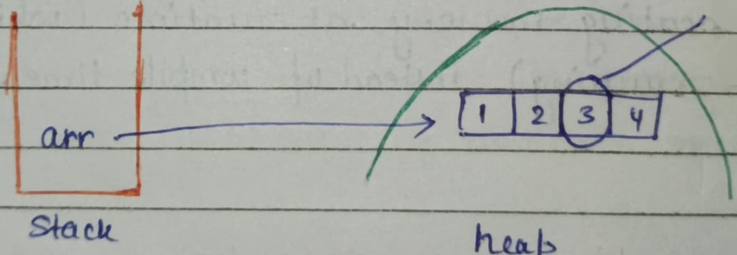
- In java, memory allocation tottaly depends on JVM whether it will be continous or not!

Reasons :-

- Objects stored in heap Memory
- In JLS(Java language Specification) it is mentioned that heap object are not continous.
- Dynamic memory allocation.

Hence; array object in java may not be continous .(it depends on JVM).

Diagram :-



Notes :-

- Array of primitives in Java are stored in contiguous memory (in most cases)

- Array of object In Jav ~~store~~ references contiguously, but not objects.

Date
Page

⇒ Index of an arrays :—

index :-	0	1	2	3	4	5
arr →	3	8	9	10	53	93

arr [0] = 3

arr [1] = 8

to change the value of certain index.

arr [2] = 19

0	1	2	3	4	5
3	8	19	10	53	93

↳ it is changed from 9 to 19

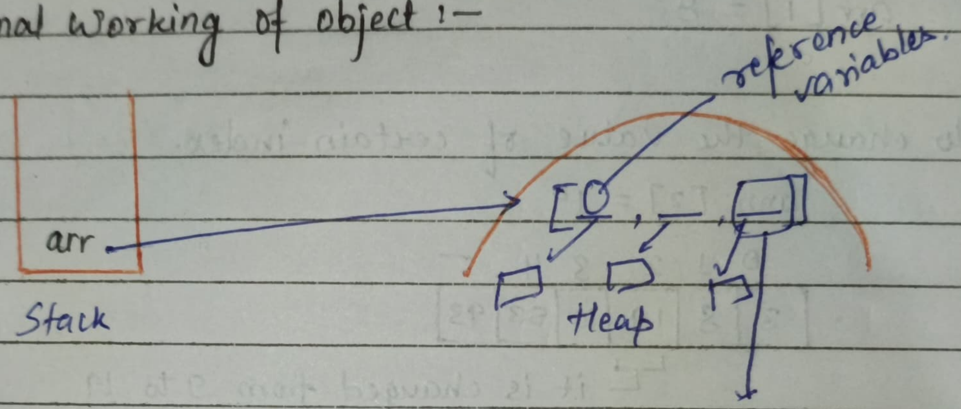
Notes :- If we don't provide value in array, internally by defaut it stored [0, 0, 0, 0, 0] for above size of array.

String Array :-

Syantax :-

String[] arr = new String [size]

Internal Working of object :-



Here each element of string array itself an object & will be stored in different part of heap memory

★ Primitives (int, char, etc) are stroed in stack.

★ All other objects are stored in heap memory.

Notes :-

- In an array, we can change the object hence, they are mutable.
- String are immutable.

Input / Output in Array :-

Input :- Using for loop.

Syantax :-

```
psrm {
    Scanner input = new scanner (system.in);
    int[] arr = new int [6];                      // declaration &
                                                     initialisation of
                                                     array
    for (int i=0 ; i< arr.length ; i++ ){
        arr[i] = input.nextInt();                 // use for loop for
                                                     taking input of
    }                                                array
```

Output 1 → Using foreach. loop.

$$\text{for } (\underbrace{\text{int}}_{\text{datatypes}}\ \underbrace{\text{num}}_{\text{variables}} : \underbrace{\text{arr}}_{\text{arrayname.}})\{$$

```
        sout (num +"  ");
    }
```

} → for each loop
- enhanced version of ~~for loop~~
- used to iterate over array in simple & readable way.

Output 2 → toString → Method.

```
~~sout(Arrays.Str~~
    sout ( Arrays.toString (array name);
}
```

→ tostring () internally uses for loop & gives the output.

2D Array :-

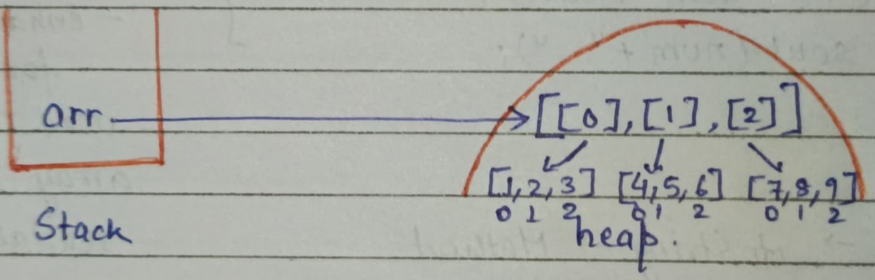
- Syntax :-
  - int [][] arr = new int [size][] — not mandatory

    row ↓ column ↓ ; └ mandatory to give size of row.

or

- int [][] arr = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9}} →

  1 2 3
  4 5 6
  7 8 9



arr [0] = [1, 2, 3]

arr [0] [2] = 3

★ datatypes [][] variable_name :- • declaration of variable.
• declared in stack during compile time.

★ variable_name = new datatype [row_size] [column_size];
- new object will be created/initialized in heap memory during runtime.

Arraylist :-

- Arraylist is a resizable array implementation of list interface in java.util package.
- It provides a dynamic array in Java.
- It is slower than standard.

★ Syntax:-

```
Arraylist <Integer> list = New Arraylist <> ( );
              ↳                            (
         Wrapperclass                        initialcapacity
        /Buit in object
```

★ Integral Working of Array list :

-
- Size is fixed internally
- So, when Arraylist get filled by some amount
  - It will make an arraylist of bigger ~~size~~ size (double) of initial arraylist.
  - Old elements are ~~coppie~~ copied In new arraylist
  - Old ones are deleted.