Day:

Array:

collections of data type.

datatype [] variable-name = new datatype [size];

* datatype → what is type of data store in array.
* all the data type should be same.
* If array of int there will be only int data type in array.
* nos → pointing to an object that is contain int data type.

nos = new int [5]

↓

refrence variable.

* int [] nos; // declaration of array

nos is getting defined in the stack.

nos = new int [5] // initialization : actually here object are creating being in the memory

↓

heap

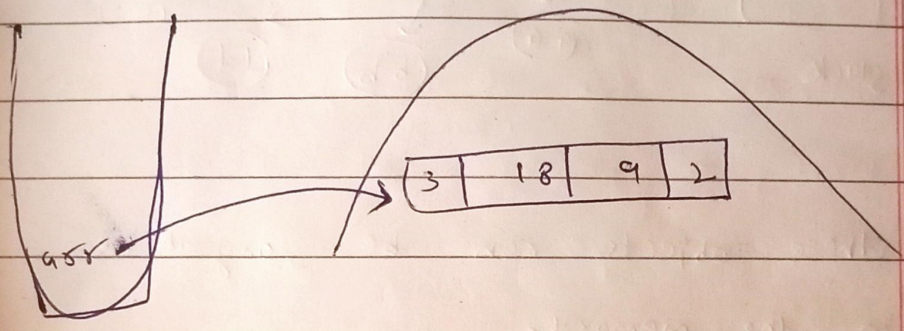
dynamic memory allocation

compile time → run time

Day: _____ Date: _____

```
int [] array = new int[5];
```

- `int []` → data type
- `array` → reference variable
- `new int[5]` → creating the object in heap memory.



stack

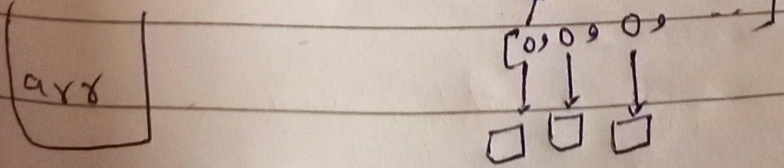
null ⇒ by default value of any refrence variable.

primitive are stored in stack

↓

int, float all data types.

but ~~non-primitive~~ object are store in heap memory.

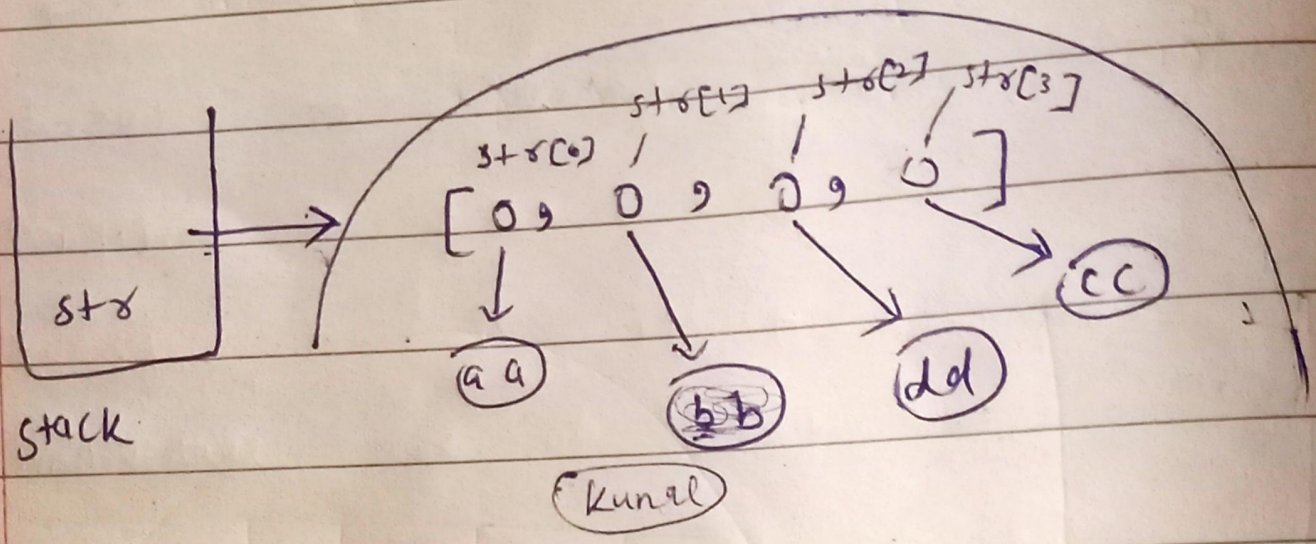


collection of refrence variable

every refrence variable is itself object.

Day:

refrence variable to the object.



these objects can be anywhere in the memory.

Array of objects.

- mutable objects → Array
- imutable → string.

Day: Date:

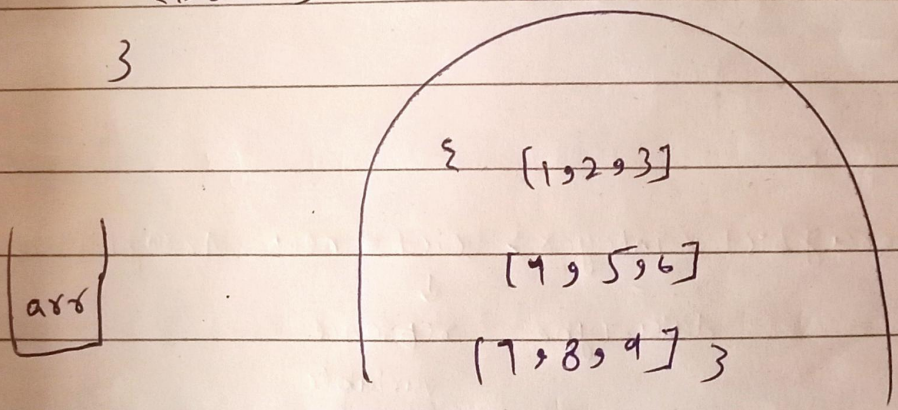
2D array

No of rows are mandatory

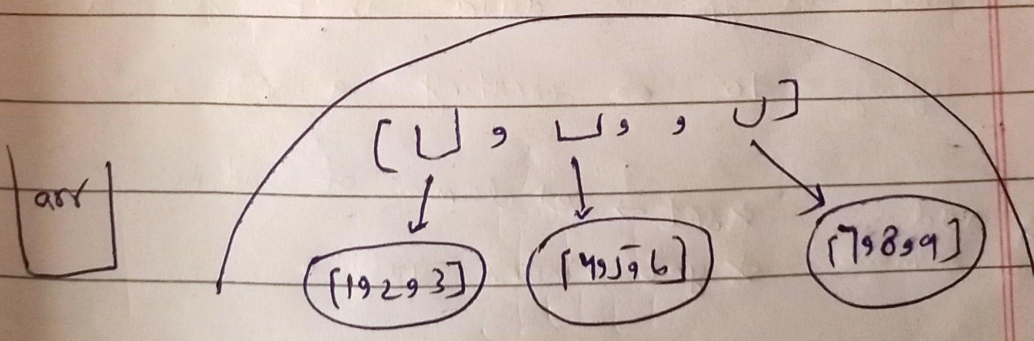
no of cols are not mandatory.

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

```
int[][] arr = {
    {1,2,3}, → 0th index
    {4,5,6}, → 1st index
    {7,8,9} → 2nd index
}
```



array of arrays.



arr[1][0] first index of the array and what is 0 index of that.

Day: _____ Date: _____

indvisual size of array can be vary.

array of every row and its length.

every row is itself array.

individual size of array is row.

what is array at every row.

what is size of array at every row.