Java Arrays

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

Basically array is collection of object

-How to declare

Datatype[] VariableName = new Datatype[size];

Or directly

Datatype[]VariableName = {1,2,3,4,5,6}

1. Datatype represents what is the type of data inside the array
2. All the type of data in the array should be same eg: you cannot do this 1st element will be int, 2nd element will be Boolean, 3rd element will be String…….. you cannot mix and match

int[] rollNo; // declaration of array .object is getting defined it the stack

rollNo = new int[5]; // actually here object is being created inside the memory (heap)

-definition of array says it the continuous data

-In C++, array is continuous

-But in java, array may not be continuous → because it depends on JVM (because heap is not continuous)

New Keyword:

It used to create an object in heap memory.

```java
//int array
int[] arr = new int[5];
System.out.println(arr[0]);  // 0


//String array
String[] str = new String[6];
System.out.println(str[4]);  //null
```

By default, the element of array is “0” for int and “null” for string and “0.0” for float.

```java
// take input
for (int i = 0; i < arr.length; i++) {
    arr[i] = sc.nextInt();
}

// print output



for (int j = 0; j < arr.length; j++) {
    System.out.print(arr[j]);
}

 // for each loop
for(int num: arr) {  // for every element in array, print the element
    System.out.print(num);  // here num represents the element of the array
}
```

Arrsys are mutable and String are immutable

```java
   int[] arrs = {1,2,3,4,5};
     System.out.println(Arrays.toString(arrs));
     change(arrs);
     System.out.println(Arrays.toString(arrs));
 }

public static void change(int[] arrs) {
     arrs[3]=23;
}
```

2D Arrays

A multidimensional array is an array of arrays.

Multidimensional arrays are useful when you want to store data as a tabular form, like a table with rows and columns.

```
public class MultiDimension {
    public static void main(String[] args) {
        /*
            1 2 3
            4 5 6
            7 8 9
        */

        int[][] arr = new int[5][];  //adding no. of row is mandatory,
column is not

        int[][] arr2 = {
                        {1,2,3},
                        {4,5,6},
                        {7,8,9}
                        };


        int[][] arr3 = {
                {1,2,3},   // 0th index
                {4,5},   // 1st index
                {7,8,9,10}   // 2nd index
        };

        for (int row = 0; row < arr3.length; row++ ) {
             for (int col = 0; col < arr3[row].length; col++){
                  System.out.print(arr3[row][col]+" ");
             }
             System.out.println();
        }



        // input
        Scanner sc = new Scanner(System.in);
        int[][] arr4 = new int[4][3];
        for (int row = 0; row < arr4.length; row++) {
            // for each column in every row
            for (int col = 0; col< arr4[row].length; col++) {
                arr4[row][col]=sc.nextInt();
            }
        }
```

```java
        // output
        for (int row = 0; row < arr4.length; row++) {
            for (int col = 0; col < arr4[row].length; col++) {
                System.out.print(arr4[row][col]+" ");
            }
            System.out.println();
        }




        for (int row = 0; row < arr4.length; row++) {
            System.out.println(Arrays.toString(arr4[row]));
        }

        System.out.println(" ");
        System.out.println(" ");


        // enhanced for-each loop
        for (int[] a : arr4) {
            System.out.println(Arrays.toString(a));
        }
    }
}
```

Arraylist

Arraylist is like a array, but there is no size limit. We can add or remove the element anytime

-Add Items

For example, to add elements to the `ArrayList`, use the `add()` method:

-Access an Item

To access an element in the Arraylist, use the get(index) method and refer to the index number:

-Change an Item

To modify an element, use the set(index, value) method and refer to the index number:

-Check an item

To check an item present in arraylist use .contain(item)

-Remove an Item

To remove an element, use the remove(index) method and refer to the index number:

To remove all the elements in the `ArrayList`, use the clear() method

-ArrayList Size

To find out how many elements an ArrayList have, use the size() method:

-Sort ArrayList

Collection.sort()

1. Size is actually fixed internally.

2. If arraylist fills by 50% it will create the new arraylist of double the size, the old element will get copy in new arraylist and old arraylist gets deleted.

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.Scanner;

public class ArrayListExample {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>(10);
        list.add(32);
        list.add(87);
        list.add(32);
        list.add(36);
        list.add(378);
        list.add(34);
        list.add(97);
        list.add(36);
        list.add(39);
        list.add(33);
        list.add(32);
        list.add(32);
        list.add(11);
        list.add(22);
        list.add(64);

        // output
        System.out.println(list);

        // to sort the array
        Collections.sort(list);
        System.out.println(list);

        // to get the size
        System.out.println(list.size());

        // check wheather the list contain
        System.out.println(list.contains(39));

        // set the value element
        list.set(0, 100);
        System.out.println(list);


        // remove
        list.remove(5);
        System.out.println(list);


        // get the list till index
        for (int i =0 ; i < 5; i++) {
            System.out.print(list.get(i)+" ");  //pass  index here;
list[index] syntax will not work here
```

```
        }

        System.out.println();
        System.out.println(list);



        //  input
        Scanner sc = new Scanner(System.in);
        ArrayList<String> list1 = new ArrayList<>(5);
        for (int i = 0; i < 5; i++) {
            list1.add(sc.next());
        }

        System.out.println(list1);



        // getting item at any index
        for (int i = 0; i < 5; i++) {
            System.out.print(list1.get(i));
        }
        System.out.println(list1);
    }
}
```

Multi-Arraylist

Arraylist of an Arraylist

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;

public class MultiArrayList {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        ArrayList<ArrayList<Integer>> list = new ArrayList<>();

        //  initialisation
        for (int i = 0; i < 4 ; i++) {
            list.add(new ArrayList<>());
        }

//          add elements
        for (int i = 0;  i < 4; i++) {
            for (int j = 0; j < 4; j++){
                list.get(i).add(sc.nextInt());
            }
        }
        System.out.println(list);
        System.out.println();




        int n = 3;

        // Here aList is an ArrayList of ArrayLists
        ArrayList<ArrayList<Integer>> aList = new
                                    ArrayList<ArrayList<Integer>>(n);

        // Create n lists one by one and append to the
        // master list (ArrayList of ArrayList)
        ArrayList<Integer> a1 = new ArrayList<Integer>();
        a1.add(1);
        a1.add(2);
        aList.add(a1);

        ArrayList<Integer> a2 = new ArrayList<Integer>();
        a2.add(5);
        aList.add(a2);

        ArrayList<Integer> a3 = new ArrayList<Integer>();
        a3.add(10);
        a3.add(20);
        a3.add(30);
        aList.add(a3);
```

```
        for(int i = 0; i < aList.size(); i++){
            for (int j = 0 ; j < aList.get(i).size(); j++){
                System.out.print(aList.get(i).get(j)+ " ");
            }
            System.out.println();
        }
    }
}
```