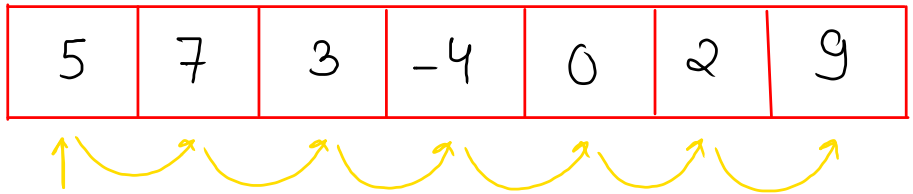
→ Loop (for each loop)

arr =



Syntax

```
for ( data_type  var_name : arr_name) {
      // statement
}
```

ex:-

```
for ( int i : arr) {
     Syso ( i );
}
```

Note:-

1) for each loop only traverse from start to end
2) it always jump by +1
3) here, we dont have indexes

ArrayList Printing

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    ArrayList<Integer> arr = new ArrayList<>();
    int n = scn.nextInt();
    for (int i = 0; i < n; i++) {
        int val = scn.nextInt();
        arr.add(val);
    }

    // print using for loop
    for (int i = 0; i < n; i++) {
        System.out.print(arr.get(i) + " ");
    }
    System.out.println();
    // print using for each loop
    for (Integer i : arr) {
        System.out.print(i + " ");
    }
}
```

for loop

for each loop

Code

Ex :-

```
public static void main(String[] args) {
    String[] arr = { "abc", "xyz", "fgh", "uyt" };
    for (String str : arr) {
        System.out.println(str);
    }
}
```

output

```
abc
xyz
fgh
uyt
```

→ Inbuilt function

To sort an array :- Arrays.sort(arr);

To sort an arraylist :- Collections.sort(arr);

↑ing order :- Collections.sort(arr);

↓ing order :-

Collections.sort(arr, Collections.reverseOrder());

→ Reverse :- Collections.reverse(arr);

ArrayList reverse printing

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    ArrayList<Integer> arr = new ArrayList<>();
    int n = scn.nextInt();
    for (int i = 0; i < n; i++) {
        int val = scn.nextInt();
        arr.add(val);
    }

    printReverse(arr);
}
public static void printReverse(ArrayList<Integer> arr) {

    for (int i = arr.size() - 1; i >= 0; i--) {
        System.out.print( arr.get(i) + " " );
    }
    System.out.println();

    Collections.reverse(arr);
    for (Integer i : arr) {
        System.out.print(i + " ");
    }
}
```

ArrayList with if-else

- First Declare an ArrayList arr.
- Then take T as an Integer input.

no. of test cases

Format for next T Lines : $(case, x(optional))$

- **case 1:** Print the size of the ArrayList in a separate line.
- **case 2:** Print and Remove element from the last index of the ArrayList.
- **case 3:** Print x and Add x in last index of the ArrayList.
- **case 4:** Print and Remove an element from the starting (index = 0) of the ArraList.
- **case 5:** Print x and Add x at beginning (index = 0) of the ArrayList.
- **case 6:** Print all the elements from left to right that are there inside the ArrayList.

Note : In **case** $2, 4, 6$ when arr is **empty** the move is invalid, so print "**invalid-move** all lowercase".

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    ArrayList<Integer> arr = new ArrayList<>();
    int t = scn.nextInt();
    for (int i = 0; i < t; i++) {
        int n = scn.nextInt();
        if ( n == 1 ) {
            printSize(arr);
        } else if ( n == 2 ) {
            removeLastIndex(arr);
        } else if ( n == 3 ) {
            int x = scn.nextInt();
            addAtLastIndex(arr, x);
        } else if ( n == 4 ) {
            removeFirstIndex(arr);
        } else if ( n == 5 ) {
            int x = scn.nextInt();
            addElementAtBeginning(arr, x);
        } else if ( n == 6 ) {
            print(arr);
        }
    }
}
```

1)
```java
public static void printSize(ArrayList<Integer> arr) {
    System.out.println(arr.size());
}
```

2)
```java
public static void removeLastIndex(ArrayList<Integer> arr) {
    if ( arr.size() == 0 ) {
        System.out.println("invalid-move");
        return;
    }
    int val = arr.get(arr.size() - 1);
    arr.remove(arr.size() - 1);
    System.out.println(val);
}
```

3)
```java
public static void addAtLastIndex(ArrayList<Integer> arr, int x) {
    arr.add(x);
    System.out.println(x);
}
```

4)
```java
public static void removeFirstIndex(ArrayList<Integer> arr) {
    if ( arr.size() == 0 ) {
        System.out.println("invalid-move");
        return;
    }
    int val = arr.get(0);
    arr.remove(0);
    System.out.println(val);
}
```

5)
```java
public static void addElementAtBeginning(ArrayList<Integer> arr, int x) {
    arr.add(0, x);
    System.out.println(x);
}
```

6)
```java
public static void print(ArrayList<Integer> arr) {
    if ( arr.size() == 0 ) {
        System.out.println("invalid-move");
        return;
    }

    for (Integer i : arr) {
        System.out.print(i + " ");
    }
    System.out.println();
}
```

Merge two sorted arrays 7

$A = [\ \overset{0}{2}, \overset{1}{3}, \overset{2}{7}, \overset{3}{9}\ ]$

$B = [\ \overset{0}{3}, \overset{1}{6}, \overset{2}{8}, \overset{3}{9}\ ]$

$ans = [\ 2, 3, 3, 6, 7, 8, 9, 9\ ]$

dry run

$A = [\ 2^{0}, 3^{1}, 7^{2}, 9^{3}\ ]$ ← i

$B = [\ 3^{0}, 6^{1}, 8^{2}, 9^{3}\ ]$ ← j

ans = [2 , 3 , 3 , 6 , 7 , 8 , 9 , 9]

→ How to remove duplicate elements

ans = [2 , 3 , 7 , 8 , 9]

0 1 2 3 4 5 6 7

i

```
while ( i < n ) {
    if ( arr.get(i) != arr.get(i+1) ){
        i++ ;
    } else {
        remove(i);
    }
}
```

1)

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] A = new int[n];
    for (int i = 0; i < n; i++) {
       A[i] = scn.nextInt();
    }

    int m = scn.nextInt();
    int[] B = new int[m];
    for (int i = 0; i < m; i++) {
       B[i] = scn.nextInt();
    }

    mergeArrays(A, B, n, m);
}
```

2)

```java
public static void mergeArrays(int[] A, int[] B, int n, int m) {
    ArrayList<Integer> ans = new ArrayList<>();
    int i = 0;
    int j = 0;
    while ( i < n && j < m ) {
        if ( A[i] < B[j] ) {
            ans.add(A[i]);
            i++;
        } else {
            ans.add(B[j]);
            j++;
        }
    }
    // adding remaining elements
    while ( i < n ) {
        ans.add(A[i]);
        i++;
    }
    while ( j < m ) {
        ans.add(B[j]);
        j++;
    }

// remove duplicate
int idx = 0;
while ( idx < ans.size() - 1 ) {
    if ( ans.get(idx) == ans.get(idx + 1) ) {
        ans.remove(idx);
    } else {
        idx++;
    }
}

// print
for (Integer x : ans) {
    System.out.print(x + " ");
}
```