

---

---

---

---

---



array → group of similar data type + contiguous memory

integers

int a

int b

int c

int d

int e

[array]

int → string X

class Main {  
 prv fun ()  
 int a=5  
}  
prv main () {

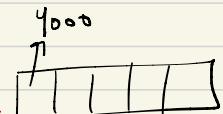
int [] arr;  
arr = new int [n];

}

temporary  
stack memory

RAM

permanent  
heap memory



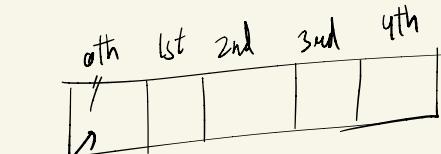
Syntax

data type + [] + array name

int [] arr; // declaration

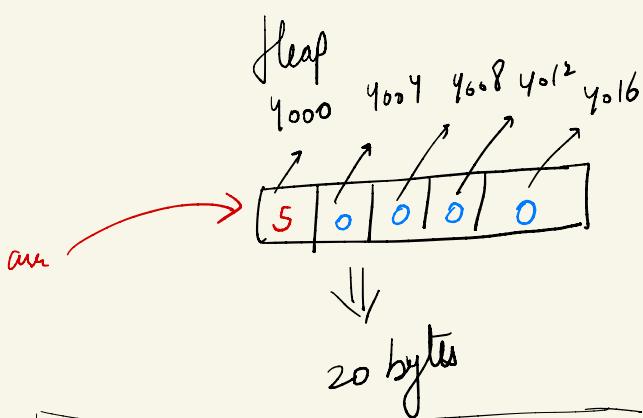
arr = new int [s];

(indexes)



arr size = n;  
Indices = [0, n-1]

int () and;  
arr = new int [s]



set

arr [idx] = value;

size

int size = arr. length;

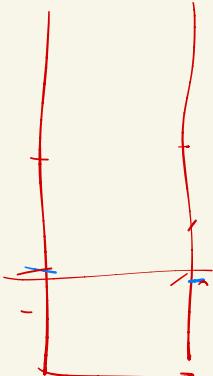
get

arr [idx]

class Main {

    psv main() {  
        1) int arr;  
        2) arr = new int [s];  
    }

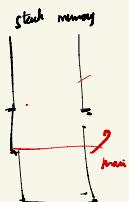
stack



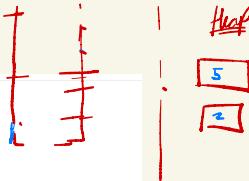
Heap

garbage collector

`arr = new int[5]; // initialization`



```
1 public class MyClass {  
2     public static void swap(int[] arr1, int[] arr2){  
3         int temp=arr1[0];  
4         arr1[0]=arr2[0];  
5         arr2[0]=temp;  
6     }  
7     public static void main(String args[]){  
8         int[] arr1={1,2,3};  
9         int[] arr2={4,5,6};  
10        swap(arr1,arr2);  
11        System.out.println(arr1[0]);  
12        System.out.println(arr2[0]);  
13    }  
14 }
```



Ques 1) Take an input  $n$ , create an array of size  $n$ .

Then Take  $n$  inputs, store it in array and print it.

Ques 2) Print the array in reverse.



$n=5$

$i=0 ; i < n$

$i = 4 \rightarrow 0$

array

19

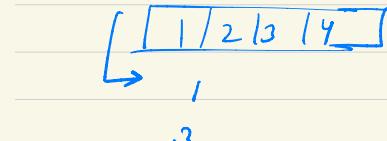
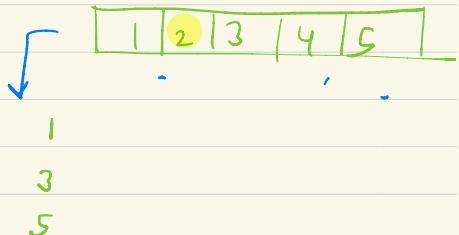
15

13

22

1

Ques 3) Print alternate elements of array



Ques 4) Print all the even numbers existing in your array and print at which index they are.

Ques 5) Get the sum of all the even numbers and all the odd numbers in your array and find their difference.

Ques 6) Find the number of elements with index = to  $n$ .

Ans 1) 0 or 1.

↓  
input this value.

$A = b$

$$\left[ \begin{array}{cccc|c} 0 & 1 & 2 & 3 & 4 & 5 \\ 4 & 4 & 1 & 4 & 4 & 12 \end{array} \right] \rightarrow 1$$

$$\boxed{\tan = 4}$$

$\tan > 0$  &  $\tan < 1$

0	1	2	3	4
12	13	19	16	28

$$\text{Even sum} = 12 + 16 + 28 = 56$$

$$\text{Odd sum} = 13 + 19 + 32 = 64$$

(24)

```

44
45     System.out.println("Difference of all the even and odd numbers");
46
47     int esum=0;
48     int osum=0;
49
50     for(int i=0; i<n; i++){
51         int v=arr[i];
52
53         if(v%2==0){
54             esum=esum+v;
55         } else {
56             osum=osum+v;
57         }
58     }
59
60     System.out.println("The difference is = "+(esum-osum));
61
62 }
```

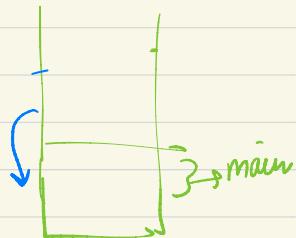
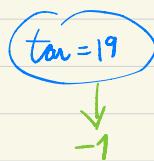
Ques 1) Take an input  $n$ , input  $n$  integers in your array  
Take another input "tar". Find at which index  
tar exist

eg.

0	1	2	3	4	5	
9	4	5	13	4	13	9

```
for (int i=0; i < n; i++) {  
    if (arr[i] == tar) {  
        return i;  
    }  
}
```

$i = 6 \times 2 + 3 + 5$



funtion → employee

↓  
(input) → work/processes → output

↓  
parameters/arguments

Ques 2 Find the maximum value in this array.

Find the minimum value in this array.

0	1	2	3	4
-10	-3	-4	-19	-5

max = -3  
↓  
Intelligent Value

```
// find maximum in array
public static int max_of_array(int[] arr, int n){
    int max=0; // this is wrong

    for(int i=0; i<n; i++){
        if(arr[i]>max){
            max=arr[i];
        }
    }

    return max;
}
```

$$n=5$$

↓

0	0	0	0	0	1
---	---	---	---	---	---

int a=5;  
int b=3;

Math.max

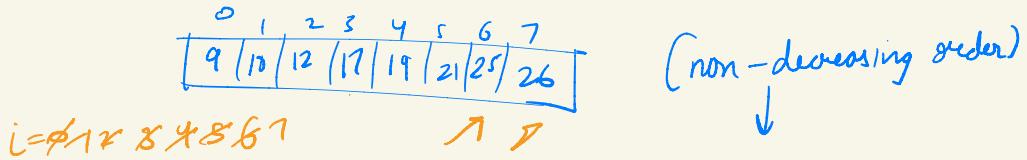
b = Math.max(a, b);

$$b = 5$$

a = Math.max(a, b);

$$a = 85$$

Ques  $\Rightarrow$  Given a sorted array, check if it contains duplicate values or not.

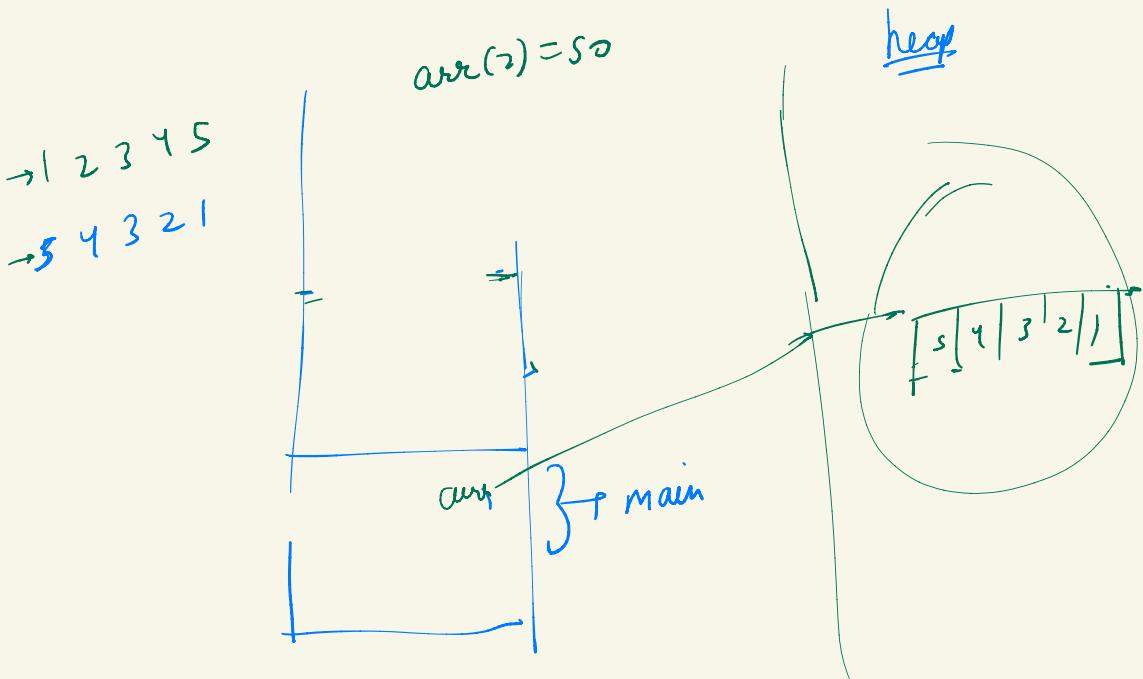


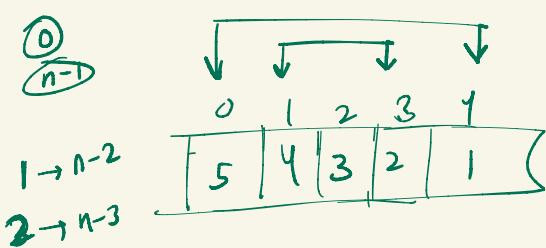
```
//find if duplicates are there in sorted array
//n-> size of array
public static boolean containsDuplicates(int[] arr, int n){
    for(int i=0; i<n; i++){
        if(arr[i]==arr[i+1]){
            return true;
        }
    }
    return false;
}
```

$i < n-1$

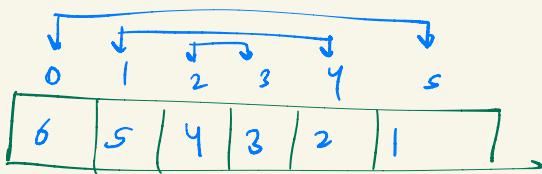
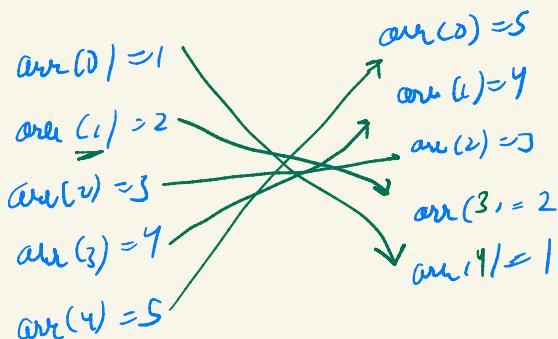
$arr(i) = 26 \quad arr(j)$   
 $arr(i+1) = \quad arr(8)$   
↓  
(arr[8])

Ques) Reverse an array. [Inplace some way]





first index  $\rightarrow$  last index  
 second index  $\rightarrow$  last second index  
 3rd  $\rightarrow$  last 3rd index  
 $\dots$   $\rightarrow$  last  $n$ th index



$$\begin{cases} 0 \rightarrow n-1(5) \\ 1 \rightarrow n-2(4) \\ 2 \rightarrow n-3(3) \end{cases}$$

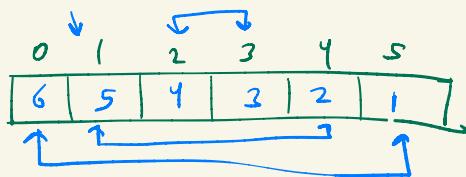
$$\begin{aligned}
 0 &\rightarrow (n-1)-0 \Rightarrow (n-1) \\
 1 &\rightarrow (n-1)-1 \Rightarrow (n-2) \\
 2 &\rightarrow (n-1)-2 \Rightarrow (n-3) \\
 3 &\rightarrow (n-1)-3 \Rightarrow (n-4)
 \end{aligned}$$

for (int i=0; i<n/2; ++)

$(i, n-1-i)$

$\boxed{i \rightarrow (n-1)-i}$

$n=6$



$i=0$

$i < 3$

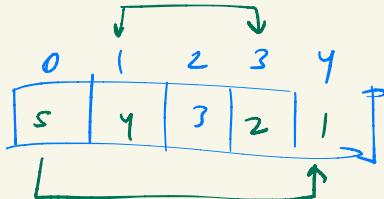
```

57
58     public static void reverse_array(int[] arr, int n){
59         for(int i=0; i<n/2; i++){
60             //swap (i) and (n-i-1);
61             int temp=arr[i];
62
63             arr[i]=arr[n-i-1];
64             arr[n-i-1]=temp;
65
66     }
  
```

$i=0, \quad n-1$

$\frac{S}{2} + 2$

$n=5$



$i=0, \quad n-1 \rightarrow 0$

$i=1, \quad n-1-1 \rightarrow 1$

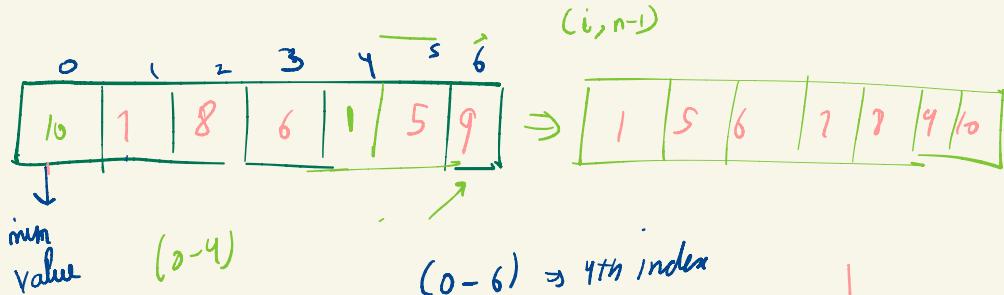
int  $a=5$   
int  $b=6$

$swap(a, b) \rightarrow b=5$   
 $a=6$

{  
 int temp = a;      ( $temp = 5$ )  
 a = b              ( $a = 6$ )  
 b = temp            ( $b = 6$ )

$a=b \times$   
 $b=a$

# 1) Selection Sort



```

4 public static void swap(int[] arr, int i, int j) {
5     int temp=arr[i];
6
7     arr[i]=arr[j];
8     arr[j]=temp;
9 }
10
11 public static void sort_array(int[] arr, int n){
12     for(int i=0; i<n; i++){
13         int min_idx=i;
14
15         for(int j=i+1; j<n; j++){
16             if(arr[j]<arr[min_idx]){
17                 min_idx=j;
18             }
19         }
20
21         swap(arr,i,min_idx);
22     }
23 }
```

$i \geq 0$

$\downarrow$

$1, 5, 6, 7, 7, 9, 10$

$\downarrow$

$\text{min\_idx} = \emptyset \leq 3 \leq y$

$\downarrow$

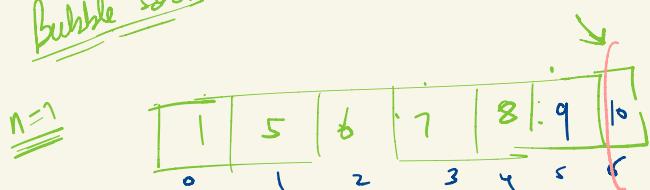
$\text{arr}(\text{min\_idx})$

$\downarrow$

$\text{min\_min}$

$i < n-1$

## Bubble Sort



$0\text{th it} \rightarrow n-2 (< n-1)$

$1\text{st it} \rightarrow n-3 (< n-2)$

$i_{\text{th}} < n-1-i$

$\downarrow$

0th iteration  $\rightarrow (5, 6)$  ( $i_{\text{th}} = n-3$ )

1st iteration  $\rightarrow (4, 5)$

2nd iter  $\rightarrow (3, 4)$

3rd  $\rightarrow (2, 3)$

4th  $\rightarrow (1, 2)$

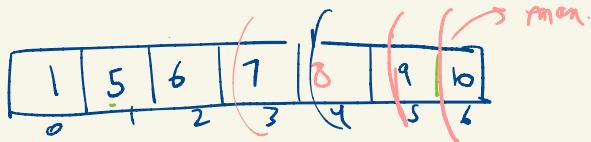
5th  $\rightarrow (0, 1)$

n-1

$i = 0$	$\rightarrow n-2$	$(i_{\text{th}} < n-1)$
$i = 1$	$\rightarrow n-3$	$(i_{\text{th}} < n-2)$
$i = 2$	$\rightarrow n-4$	$(i_{\text{th}} < n-3)$
$i = 3$	$\rightarrow n-5$	$(i_{\text{th}} < n-4)$
$i = 4$	$\rightarrow n-6$	$(i_{\text{th}} < n-5)$
$i = 5$	$\rightarrow n-7$	$(i_{\text{th}} < n-6)$

$\downarrow$

$n-1-5$



$i=28486$

```

25 // bubble sort =====
26
27 public static void bubble_sort(int[] arr, int n){
28
29     for(int i=0; i<n-1; i++){ // more than one iterations will be required
30
31         for(int idx=0; idx<n-1-i; idx++){
32             if(arr[idx]>arr[idx+1]){
33                 swap(arr, idx, idx+1);
34             }
35         }
36     }
37 }
38 }
39 }
40 }
```

$i \leq n-1-i$        $i \leq 1$   
 $i \leq n-5$        $i \leq 2$

arr[i]

6  
 4 5 3 8 6 1  
 ↑  
 $i \leq n-1$

$\text{arr}[i-1] < \text{arr}[i] > \text{arr}[i+1]$   
 $\text{arr}[i-1] < \text{arr}[i]$   
 $\text{arr}[i+1] < \text{arr}[i]$

## Break Statement

```
for (int i=1; i<=10; i++) {
    if (i==5)
        break;
}
```

$i=1 \leftarrow$   
 $i=2 \leftarrow$   
 $i=3 \leftarrow$   
 $i=4 \leftarrow$   
 $i=5 \leftarrow$

Continue Statement  $\rightarrow$  skips the current iteration -

```
for(int i=1; i<=10; i++){
    if(i==7){
        continue;
    }
    System.out.println(i);
}
```

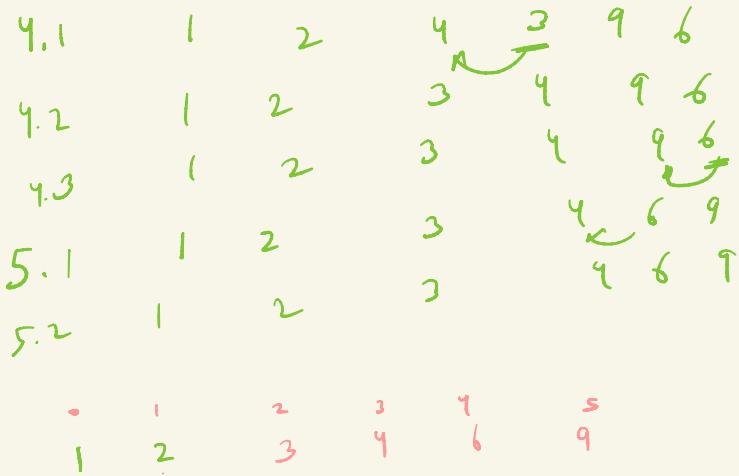
$i=x8$

1  
 2  
 3  
 4  
 5  
 6  
 8  
 9  
 10

## Insertion Sort

Iterations = n - 1

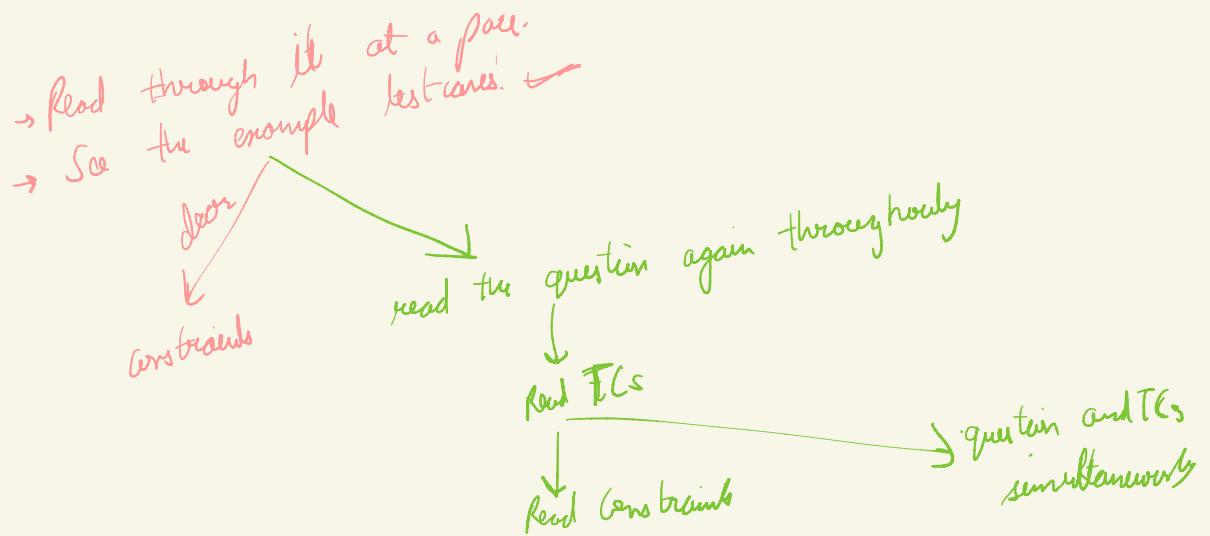
	0	1	2	3	4	5
	9	2	4	1	3	6
1.1	2	9	4	1	3	6
2.1	2	9	9	1	3	6
2.2	2	9	9	1	3	6
3.1	2	9	9	1	9	6
3.2	2	1	9	9	3	6
3.3	1	2	9	9	3	6



$i = 1, 2, 3, 4, 5,$

```
public static void insertion_sort(int[] arr, int n) {
    for(int i=1; i<n; i++){ // we need to insert arr[i] at correct position
        for(int j=1; j>0; j--){ // asking j-1 everytime
            if(arr[j-1]>arr[j]){
                swap(arr,j-1,j);
            } else {
                break;
            }
        }
    }
}
```

5 3 4 2 1



0 1 2 3 4 5  
3 6 9 3 3 9

(0, 3)  
(0, 4)

(2, 5)  
(3, 4)

```
public static int getLuckyPairs(int[] arr, int n){  
    int count = 0; int i=0; i<n; i++
```

```
        for(int i=0; i<n; i++){  
            for(int j=i+1; j<n; j++){  
                if(arr[i]==arr[j])  
                    count++;  
            }  
        }
```

```
    return count;  
}
```

Ans Given an array, find first occurrence and last occurrence of given target value.

0	1	2	3	4	5	6	7	8	9	10	11
1	3	4	3	4	5	5	3	7	3	4	5

for  $i=3$ ,  $l_0 = 1$ ,  $h_0 = 9$   $\rightarrow$  single for loop

$i = 7 + 2 \times 2 = 11$   $l_0 = 8, h_0 = 9$

```

int lo=-1;
int fo=-1;

for(int i=0; i<n; i++){
    if(arr[i]==tar){
        if(fo== -1){
            fo=i;
        }
        lo=i;
    }
}

```

```

System.out.println("first occurrence is: "+fo);
System.out.println("last occurrence is: "+lo);

```

Ques Find if the given array is mountain array or not

→ 2, 21, 45, 50, 33, 29, 25, 19 ✓

(i) 1, 81, 79, 50, 40 ✓

(i=1) 1, 2, 7, 9, 11, 5 ✓  
1 2 3 4 5 → X

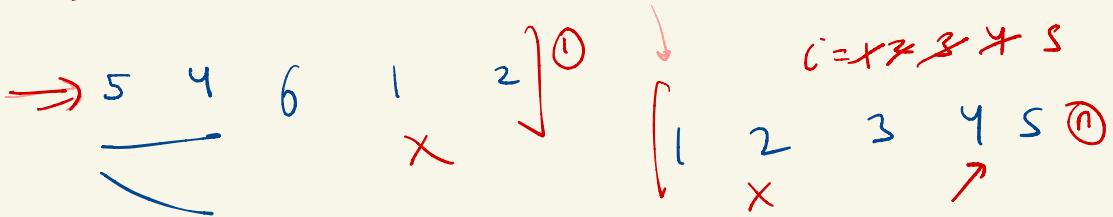
| 4 3 8 2 → X 5 4 3 2 1 → X

✓ arr < arr(i) < - - arr(i) > arr(i+1) > arr(i+2)  
- - - arr(n-1)

5 4 6 2 1 → false

i = X ≠ 3

[2 9 8 5 3] ?

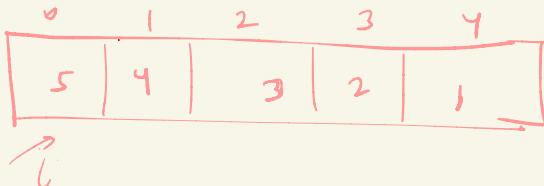


```
for(i=1; i<n; i++){
    if(arr[i-1]>=arr[i]){
        break;
    }
}
```

→ declaration  
while ( condition ) {  
 // work  
 // → inc/dec  
 ↓

{ for (int i=0; i<n; i++)  
 {  
 int i=0;  
 while (i<n) {  
 i++  
 }  
 }  
}

Two pointers



$i = \cancel{x} + 3$   
 $j = 4 - 2 + 1$

reverse

```
public static void reverse(int[] arr, int n){
    int i=0;
    int j=n-1;

    while(i<=j){
        // swap i and j

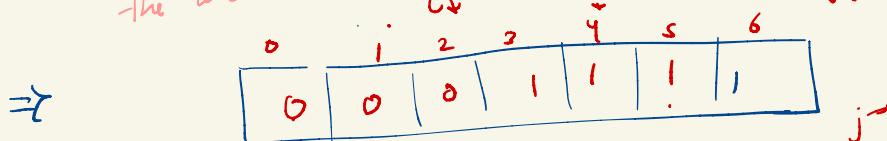
        int temp=arr[i];

        arr[i]=arr[j];
        arr[j]=temp;

        i++;
        j--;
    }
}
```

$i <= j$

Ans Given an array containing only 0's and 1's. Sort the array.



$i = \cancel{x}^0$   
 $j = \cancel{0}x^23^4$

```
// (0, i) -> zeros
// (i+1, j-1) -> ones
// (j, n-1) -> undefined
public static void sort_zero_one(int[] arr, int n){
    int i=-1;
    int j=0;

    while(j<n){
        if(arr[j]==0){
            i++; // increasing zeros' area
            swap(arr,i,j);
            j++;
        } else { // increasing ones' area
            j++;
        }
    }
}
```

ans

value



areas

$(0, i) \rightarrow \text{zeros}$

$(i+1, j-1) \rightarrow \text{ones}$

$(j, n-1) \rightarrow \text{undefined}$

$\text{if } (\text{arr}[i] == 1) \{$   
 $\quad j++;$

$(0, i)$

$(i+1, j-1)$      $(j, n-1)$

$(0, i) \rightarrow \text{zeros}$

$(i+1, j-1)$

$(j, n-1) \rightarrow \text{undefined}$

$\text{zeros} = 5$

$\text{ones} = 4$

$\text{twos} = 1$

$\text{zeros}$

$i$

$j$

$0$

$n-1$

Ques

Given an array containing 0's, 1's and 2's. Sort the array in one traversal.

0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	1	1	1	1	2	2

$p_1$

$p_3$

unknown  
(0, 1, 2)

$p_2$

$p_3$

$n-1$

0's

1's

2's

$(0 - p_1) \rightarrow \text{zeros}$

$(p_1 + 1, p_2 - 1) \rightarrow \text{ones}$

$(p_2, p_3) \Rightarrow \text{unknown}$

$(p_3 + 1, n-1) \rightarrow 2's$

$(10, 10)$

$(p_2, p_3)$   
 $p_2 > p_3$

swap( $p_2, p_3$ )

```
// (0, p1) -> zeros
// (p1+1, p2-1) -> ones
// (p2, p3) -> unknown
// (p3+1, n-1) -> twos
public static void sort_zero_one_two(int[] arr, int n){
    int p1=-1;
    int p2=0;
    int p3=n-1;

    while(p2<=p3){ // unknown area alive
        if(arr[p2]==0){
            p1++; // increase zero area
            swap(arr, p1, p2); // swapping 0 and 1
            p2++;
        } else if(arr[p2]==2){
            swap(arr, p2, p3);
            p3--;
        } else {
            p2++;
        }
    }
}
```

and 1

mn area

if ( $\text{arr}(p_2) == 2$ ) {  
 swap( $\text{arr}, p_2, p_3$ )  
 $p_3--$   
 ↓  
 increase  
 2's area}

## Intersection of two arrays

$a = \{1, 1, 2, 3, 3\}$   
 $b = \{1, 1, 2, 2, 3\}$

$$a(i) = 1$$

$$b \rightarrow$$

$$a(i) == b(j)$$

✓

Count ++

$$\text{Common} = \{1, 2, 3\} \Rightarrow 3$$

$a \rightarrow \underline{\text{skip}}$

$\curvearrowright$

$a[i] == a[i-1] \Leftarrow \text{skip}$

$i = 1, 2, 3, 4, 5$

```
// the intersection of two arrays.
public static int NumberOfElementsInIntersection(int a[], int b[], int n, int m) {
    int count=0;
    Arrays.sort(a);
    for(int i=0; i<n; i++){
        // search a[i];
        if(i>0 && a[i]==a[i-1]){
            continue;
        }
        for(int j=0; j<m; j++){
            if(a[i]==b[j]){
                count++;
                break;
            }
        }
    }
    return count;
}
```

$O(n * m)$

$a = \{1, 2, 2, 3, 4, 5, 6\}$   
 $b = \{5, 6, 3, 4, 7, 2, 3\}$

$\{2, 3, 4, 5, 6\}$

count = 0, 1, 2, 3, 4

$\begin{cases} & \text{if } (a(i) > b(j)) \\ & j++ \end{cases}$

$a = \{1, 2, 2, 4, 11, 14, 19, 19, 100, 120\}$

$i = 0, 1, 2, 3, 4, 5$

$b = \{2, 11, 12, 13, 14, 21, 25, 100, 109\}$

$j = 0, 1, 2$

$a(i) = 100, 120$

$\begin{cases} & \text{if } (a(i) < b(j)) \\ & i++ \end{cases}$

$b(j) = 2, 12, 13, 14, 21, 25, 109$

$\begin{cases} & \text{if } (a(i) == b(j)) \\ & \text{count}++ \\ & i++ \\ & j++ \end{cases}$

3

```

// the intersection of two arrays.
public static int NumberOfElementsInIntersection(int a[], int b[], int n, int m) {
    int count=0;

    Arrays.sort(a);
    Arrays.sort(b);

    int i=0;
    int j=0;

    while(i<n && j<m){
        if(i>0 && a[i]==a[i-1]){
            i++;
            continue;
        }

        if(a[i]==b[j]){
            count++;
            i++;
            j++;
        } else if(a[i]<b[j]){
            i++;
        } else {
            j++;
        }
    }
}

```

$O(m+n)$

$a = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15\}$   
 $b = \{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16\}$   
 $\text{ans} = \boxed{1 | 2 | 2 | 4 | 11 | 11 | 12 | 13 | 14 | 19 | 21 | 22 | 120 | 130 | 140 | 150}$

$$a(i) = 4 + 11 + 12 + 19 + 120$$

$$b(j) = 2 + 4 + 12 + 13 + 14$$

$$-10^9 \rightarrow 10^9 \rightarrow \text{int}$$

$$-10^{18} \rightarrow 10^{18} \rightarrow \text{long}$$

## Two Sum

$$\text{tar} = 9$$

$$[-1, 11, -2, 5, 13, 9, 4, -4, 15] \Rightarrow 3$$

count =  $i \times 3$

sort  $\Rightarrow$   $[-4, -2, -1, 4, 5, 9, 11, 13, 15]$

tar = 9

$$\text{arr}(i) = 14$$

$$\text{arr}(j) = 9$$

$$\text{sum} = \frac{\text{arr}(i)}{\text{arr}(j)} + \text{arr}(j) = 14 + 9 = 23$$

$\text{if } (\text{sum} > \text{tar})$   
 $j--$

$i++ \rightarrow \text{sum increase}$   
 $j-- \rightarrow \text{sum decrease}$

```
// count of distinct pair with given sum
public static int countPairs(int[] nums, int target){
    int n=nums.length;

    int i=0;
    int j=n-1;
    int count=0;

    while(i<j){
        int sum=nums[i] + nums[j];

        if(sum==target){
            count++;
            i++;
            j--;
        } else if(sum<target){
            i++;
        } else {
            j--;
        }
    }

    return count;
}
```

$$[12, 3, 6, 1, 6, 9] \quad \text{tar} = 24$$

required sum =  $24 - 6 = 18$

$$[1, 3, 6, 6, 9, 12]$$

$i = k+1$   
 $j = n-1$

$$\text{want} = \frac{1}{2} X^2$$

$$\text{arr}(i) + \text{arr}(j) + \text{arr}(k) = \text{tar}$$

$$\text{arr}(i) + \text{arr}(j) = \frac{\text{tar} - \text{arr}(k)}{2}$$

$(i < j)$

$$\Rightarrow \boxed{\text{arr}(i) + \text{arr}(j) = T}$$

where  
 $T = \text{tar} - \text{arr}(k)$

$$\boxed{\text{arr}(i) + \text{arr}(j) + \text{arr}(k) + \text{arr}(l) = \text{tar}}$$

$$\boxed{\text{arr}(i) + \text{arr}(j) + \text{arr}(k) = T}$$

$T = \text{tar} - \text{arr}(l)$ .

