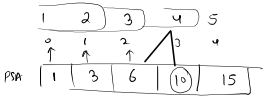
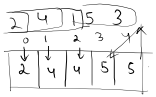
Previous

PSA:



arr[i] + psa[i-1]

1. Greatest till me.



max (pa[i-1] , arr[i])

2. Print Prefix b/w L and R.

n=5 → [4 2 1 3 5]
0 1 2 3 4

l = 1
r = 3

4 6 7 10 15

6 7 10

3. Print freq of alphabets

"a b c c c b a d p"

1. Making arr of size 26

a-2
b-2
c-3
d-1
p-1

a -2
b-2
c-3
d-1
p-1

4. a b a a c c c c b

a - 3
b - 2
c - 4 → freq ↑

c

maxf = 0 → 3 → 4
mfc = 'a' → 'c'

3 > maxf

4 > maxf

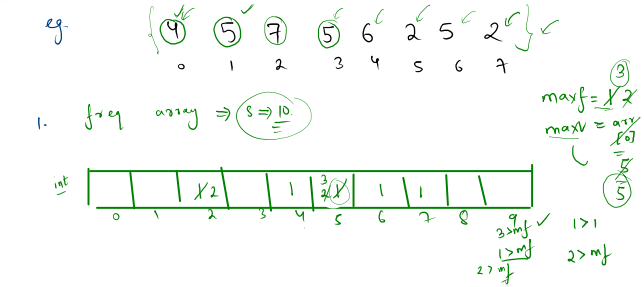
4 > maxf 4 > 4

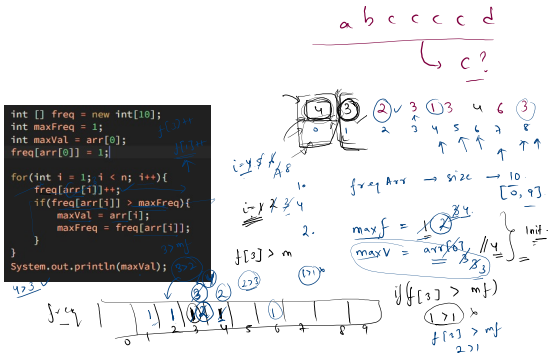
2 > maxf

2 > 4

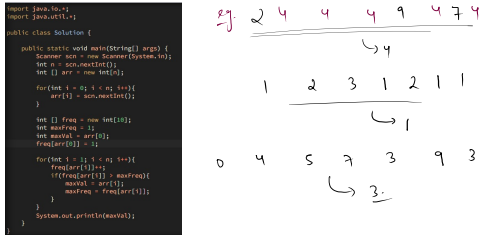
Int with Maximum Freq

Given an array that contains only one-digit integers from 0-9. Print the digit which occurs in the array the maximum number of times. You must take a constant extra space and also you have to complete the question in the single traversal.

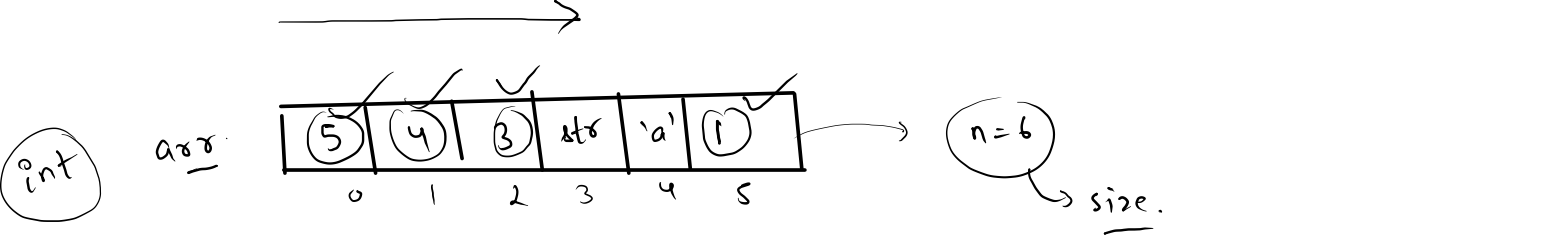




```
int [] freq = new int[10];
int maxFreq = 1;
int maxVal = arr[0];
freq[arr[0]] = 1;

for(int i = 1; i < n; i++){
    freq[arr[i]]++;
    if(freq[arr[i]] > maxFreq){
        maxVal = arr[i];
        maxFreq = freq[arr[i]];
    }
}
System.out.println(maxVal);
```

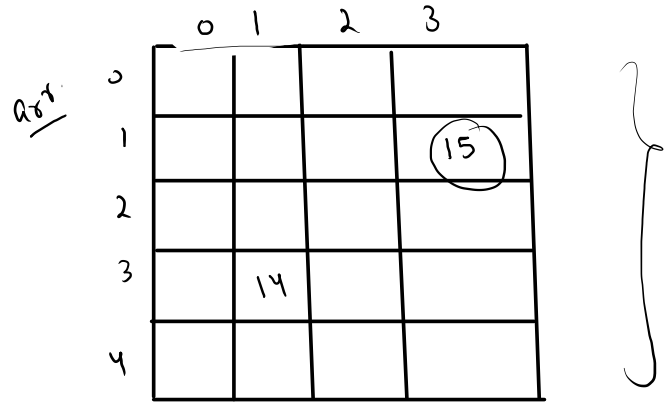


```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] arr = new int[n];

        for(int i = 0; i < n; i++){
            arr[i] = scn.nextInt();
        }

        int [] freq = new int[10];
        int maxFreq = 1;
        int maxVal = arr[0];
        freq[arr[0]] = 1;

        for(int i = 1; i < n; i++){
            freq[arr[i]]++;
            if(freq[arr[i]] > maxFreq){
                maxVal = arr[i];
                maxFreq = freq[arr[i]];
            }
        }
        System.out.println(maxVal);
    }
}
```



2D. → matrix.

1D → n.



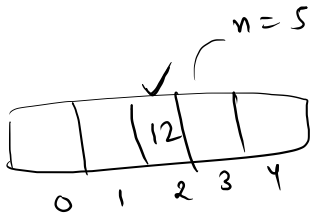
n x m

5 x 4.

arr[1][3] = 15;

Syso (arr[1][3]);

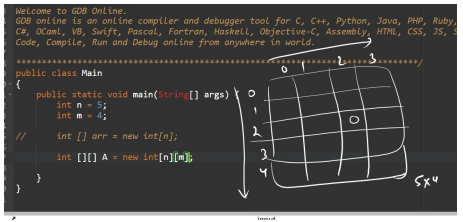
5 1D array

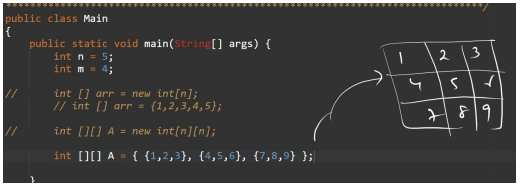


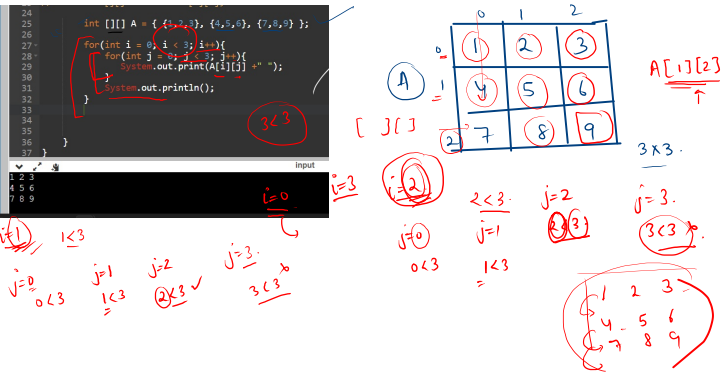
arr[2] = 12;

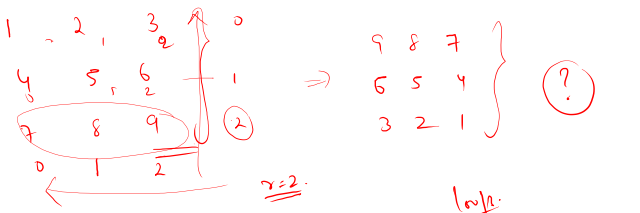
arr[3][1] = 14;

2D - Arrays (Matrix)









Print Alternate Row

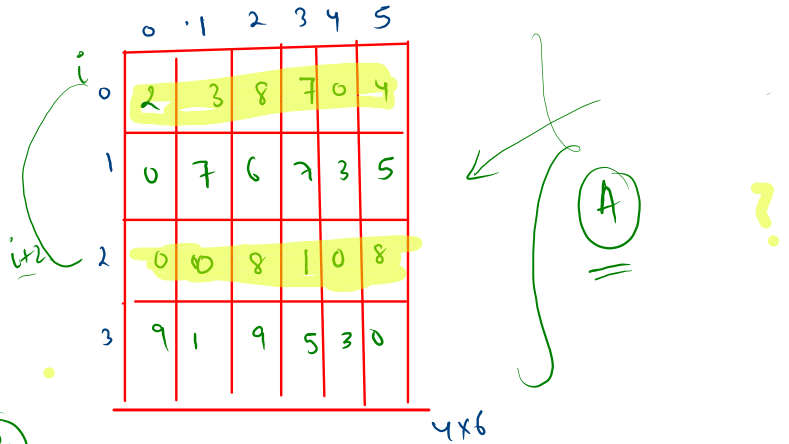
You are given a 2D matrix , your task is print the **alternate rows** of the matrix starting from the **0th** row.

Sample Input 0

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

Sample Output 0

```
2 3 8 7 0 4
0 0 8 1 0 8
```



i+2

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();  //4
        int m = scn.nextInt();  //6

        int [][] A = new int[n][m];

        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                A[i][j] = scn.nextInt();
            }
        }

        //logic
        for(int i = 0; i < n; i += 2){
            for(int j = 0; j < m; j++){
                System.out.print(A[i][j] + " ");
            }
            System.out.println();
        }


    }
}
```

i, j

odd / even.

```java
public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();  //4
        int m = scn.nextInt();  //6

        int [][] A = new int[n][m];

        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                A[i][j] = scn.nextInt();
            }
        }

        //logic
        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                if(i % 2 == 0){
                    System.out.print(A[i][j] + " ");
                }
            }
            if(i % 2 == 0){
                System.out.println();
            }
        }
```

$O(n^2)$