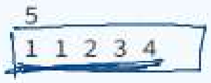
Que Missing and Repetitive # from array

15 − 11 + 1
= 15 − 10
length = 5 → Missing



5
1 1 2 3 4

1 to N ⇒
11

1	1	2	3	4

1 to N

1 − 5

2

1 to N

1 to 5 →

```
for( 1 to 5) {
    for (J − 0 to arr.length) {
        if (i == arr[i] ) {
            flag = true;
            break
        }
    }
}
```

1

15 − (11 − 1) → Rep

✓ 15 − 11 + 1
= 15 − 10 = 5 Missing

1 2 3 = 10
= 6

Bruteforce

1	1	2	3	4

missing

```
for( i-1 to n ){
    boolean flag= false; // count= 0;
    for( j-0 to arr.length ){
        if( i == arr[i] ){
            flag= true; ~~t~~ , count ++;
        }
    }
    if( flag == false ){          //  if(count ==2) {
        syso( i );                         syso( i );
    }                                 }          //Repetition
}
```

Repetition.

arr ⇒

f	f	f	f	f	f
1	1	2	3	4	
0	1	2	3	4	

i = 1 2 3 4 5
~~i~~ i ~~i~~ ~~i~~ i

$n=5 = \frac{n(n+1)}{2} = \frac{5 \times \cancel{6}^{3}}{\cancel{2}} = 15$ ⇒

$arrSum = \boxed{11} - \boxed{1} = \boxed{10}$

$15 - 10 = \boxed{5}$ → missing #

$O(n^2) = O(n)$

$O(1)$

0	1	2	3	4
-1	1	2	3	4
i	i			

val = arr[i] - 1
= 0

$arr[\underset{0}{val}] = - arr[\underset{0}{val}]$

i	x	x	i	
-1	-3	-2	3	4
0	1	2	3	4

```
for( i=0 to n-1){

    int  val = Math.abs(arr[i]); // 3
    int  index = val-1; 2
//O(n)
    if (arr[index] >0){
        arr[index] = -arr[index];

    }else { repetitive = arr[i];
    }          TC = O(n)
}
```

✓

sum = 15

rep = 15 - 13 + 3

= 15 - 10

= 5

//O(1)

x	i		
-1	2	4	4
0	1	2	3

val = 1

index = 0

Que Second Largest

	i	i				
6	2	4	1	5	3	8
0	1	2	3	4	5	6

max = 6

smax = 2

$\boxed{\text{max} = 2}$

```
if ( arr[i] > max ){
        smax = max;
        max = arr[i];

}
else if (arr[i] > smax){
        smax = arr[i];

}

sout( smax)
```

ans = $\boxed{6}$

```
{
  if (arr[0] > arr[1]){
      max = arr[0];
      smax = arr[1]
  }
  else {
      max = arr[i]
      smax = arr[0]
  }
}
```

	nst	nsp
i=1	1	4
i=2	2	3
i=3	3	2
i=4	4	1
i=5	5	0

i=~~1~~ ~~2~~ ~~3~~ 4

i	1	2	3	4	5
1	*	_	_	_	_
2	*	*	_	_	_
3	*	*	*	_	_
4	*	*	*	*	_
5	*	*	*	*	*

1method = i, j ✓

2method = # of stars

# of spaces

```
for (int i=1 to n){
    for (j=1 ; j<=i ; j++){
j=1         syso(*)
}
```

```
*
* *
* * *
* * * *
```

```
nsp = ~~4~~ ~~3~~ ~~2~~ ~~1~~ 0
nst = ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5
```

```
*----
**---
***--
****-
*****
```

```
}     nsp = n-1 // 4
      nst = 1
for( int i = 1; i<=n ;i++){

    for (int st = 1 ; st<=nst ; st++){
            syso(*);

    } for( int sp = 1, sp<=nsp ; sp++){
            syso(" ");
        }

    nst++;
    nst--;
}
```

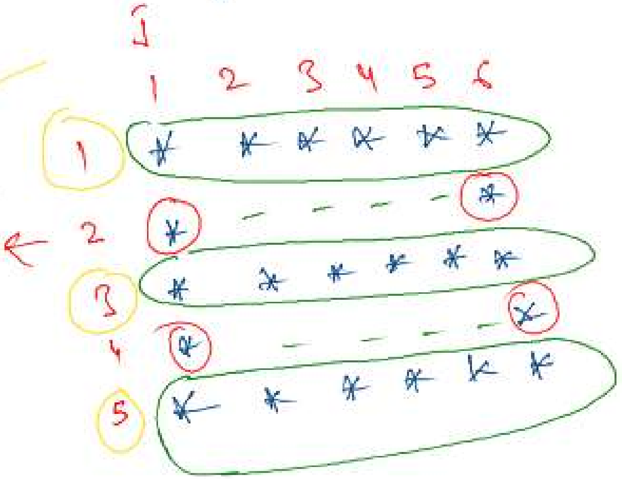
}

when i is odd

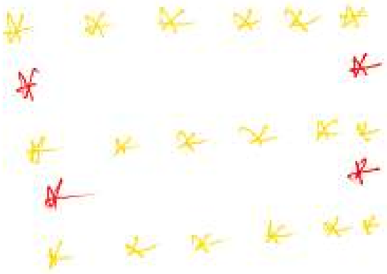
when i is even

(J=1 || J=n) ←

*

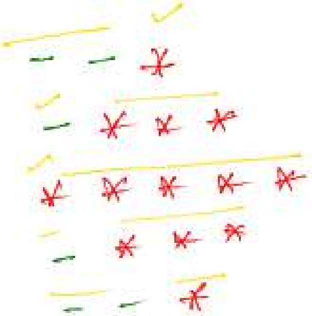


n=5

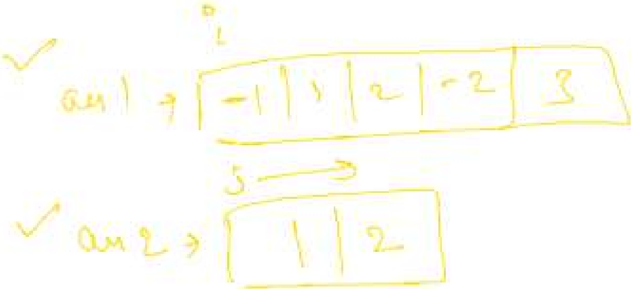


n/2

	nsp	nst
1	2	1
2	1	3
3	0	5
4	1	3
5	2	1



```
n/2 = nsp --;
    = nst += 2;
else {
    nst -= 2;
    nsp ++;
}
```



absolutevalue.

```
if (Math.abs(arr1[i]) == arr2[j]) {
        sysout(arr[i]);
```

6