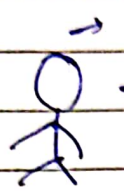
Function : call by value and by reference

Reference Variable

int n=5;

n → 5

↳ is n ko aap kisi aur nam se bulana chahte ho toh.



int &P = n

↓

Reference Variable.

n, k → 5

ab iska naam k bhi hai (alias name)

original name - n

cout << n → 5

cout << k → 5

int &t = 6; → you cannot do that

↳ kisi original variable ka alias name hai

call by value.

n → 5 → copy value.

```
int increment (int n)
{
    n = n+1;
    return n;
}
```

↳ 6

```
int main() {
    int n;
    cin >> n;
    n = increment(n);
    cout << n << endl;
    return 0;
}
```

n → 5

↳ 6

In this method, the value of each variable in the calling function is copied into corresponding dummy variables of the called function.

Call by Reference

```
Void increment (int &n)
{
    n++;
    return;
}
int main(){
    int n;
    cin >> n;
    increment (n);
    cout << n << endl;
    return 0;
}
```

↳ sent the reference of actual variable.

In this method, the address of actual variables in the calling function is copied into the dummy variables of the called function.

Array - Level 2

→ find unique element

i/p 2 10 11 13 10 2 15 13 15

Each element → occurs twice except one
↳ find out.

Solve XOR → same value → 0
different value = 1

So all numbers becomes zero except one

① init ans = 0
② ans ^= arr[i]

2	10	11	13	10	2	15	13	15
0	1	2	3	4	5	6	7	8

arr

ans = 0^10^2^11^10^2^13^15^13^15
= 0^11
ans = 11

→ i/p → array → [] → {10, 20, 30}
↳ print all pairs.

10	20	30
0	1	2

arr →

i = 0, j = 0 → arr[i], arr[j] → (10, 10)
j = 1 → (10, 20)
j = 2 → (10, 30)

i = 1, j = 0 → (20, 10)
j = 1 → (20, 20)
j = 2 → (20, 30)

i = 2 j = 0 → (30, 10)
j = 1 → (30, 20)
j = 2 → (30, 30)

→ Same concept with all triplets.

→ Sort 0's and 1's

i/p

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

arr →

o/p 0 0 0 0 0 0 1 1 1

① Counting
② Two pointer approach.
③ sort()

① Simply count the no.s of zeroes or ones and then put in the Array.

Two pointer Approach.

(1) We have to initialize 2 variables.

(i) int left = 0

(ii) int Right = size - 1.

(2) if left = 0

left ++

else swap (left, Right)

Code Snippet

```
int low = 0, high = size - 1;
while (low <= high)
{
    if (nums [low] == 0) {
        low++;
    }
    else {
        swap (nums [low], nums [high]);
        high;
    }
}.
```

→ Shift arrays element by 1.

i/p

10	20	30	40	50	60
0	1	2	3	4	5

o/p

60	10	20	30	40	50
0	1	2	3	4	5

(1) store last element

(2) move array by one place.

(3) place on first index that stored element.

→ Shift arrays element by k.

(1) Store k^{th} elements on temp [ ]

(2) Shift elements by k times.

(3) Copy temp.

classmate
Date
Page 124

▶ Doubts with Lakshay Khaiya [Week 3]

→ Representation of STACK and with specific parts.