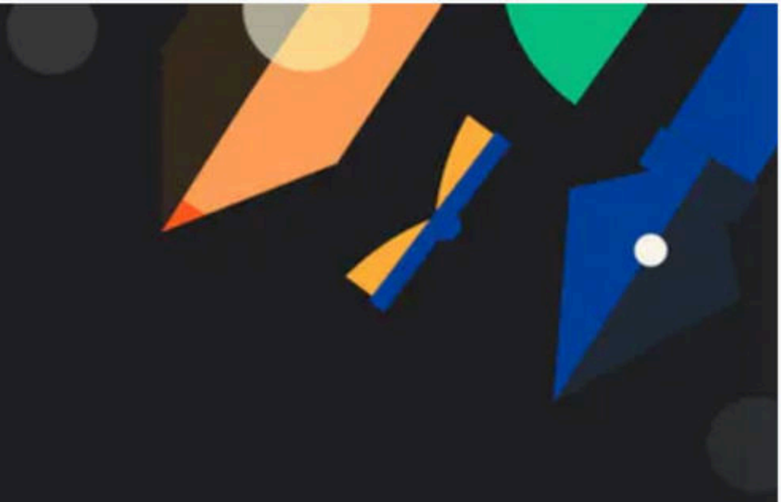


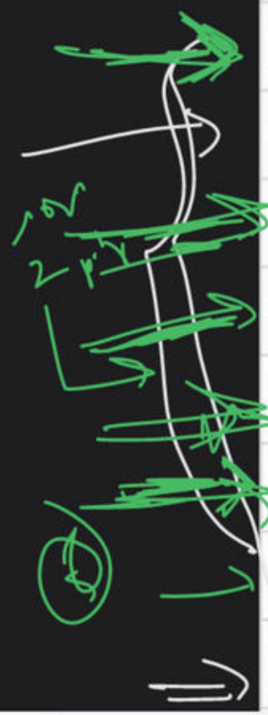
Array Problems - I

Foundation Course on Data Structures & Algorithm - Part I

Love Babbar • Lesson 7 • Apr 26, 2022

Questions:

- Linear Search in an array
- Reverse an Array
- find max and min element in an array
- Swap Alternates in an array
- Sort an Array of 0, 1 and 2
- Move all negative number to one side of array
- Find union and intersection of 2 sorted arrays
- Program to cyclically rotate an array by one
- find duplicate in an array of N+1 integers
- find the pair that sum to a given value
- find the triplet that sum to a given value
- Check whether an array is palindrome or not
- Minimum swaps required bring elements less equal k together
- Unique Number of Occurrences - Leetcode
- Kadane's Algo



→ Array Problems :-

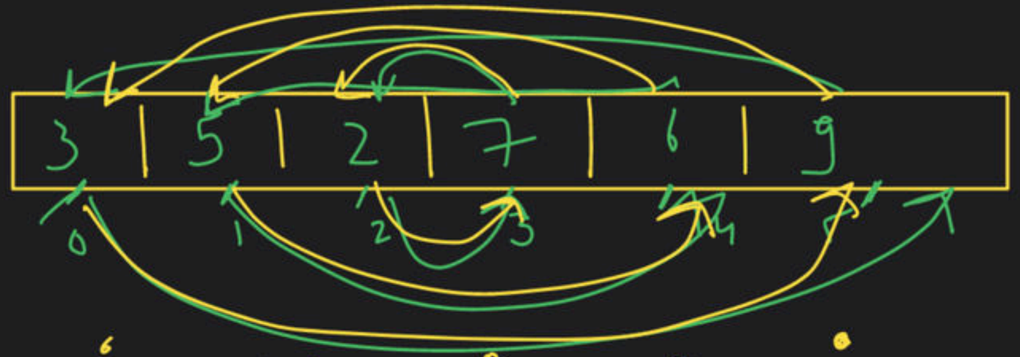
→ Reverse an array

Reverse

o/p → | 9 | 6 | 7 | 2 | 5 | 3 |

0 1 2 3 4 5

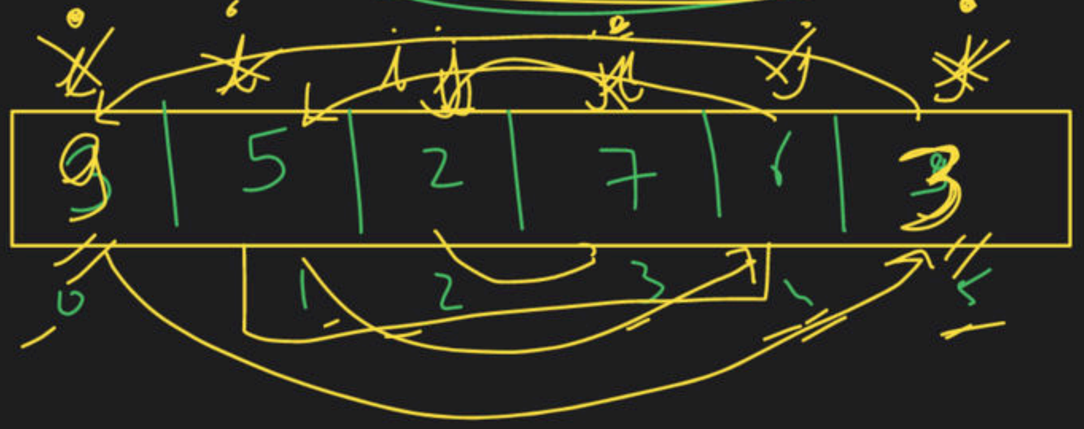
i/p →



approach :-

n = 6

index 0 - (n-1)




```
i = 0
j = n-1
while( i < j)
{
    swap(arr(i), arr(j))
    i++;
    j--;
}
```

→ swap (src , dest)

swap (arr[i] , arr[j])

after swap

→ gswap

n times

next class → static

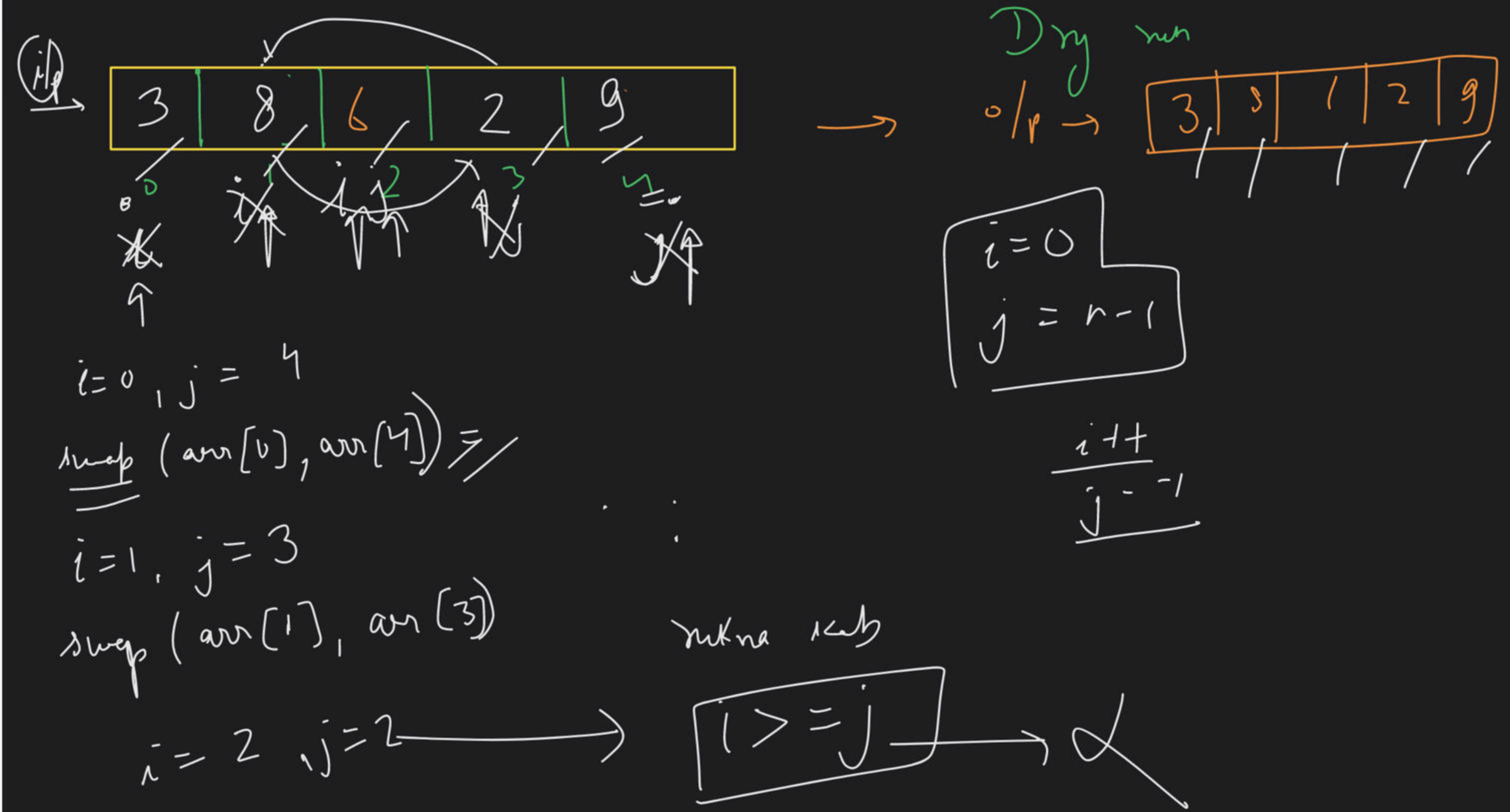
private channel

Link

→ recording → Y/N

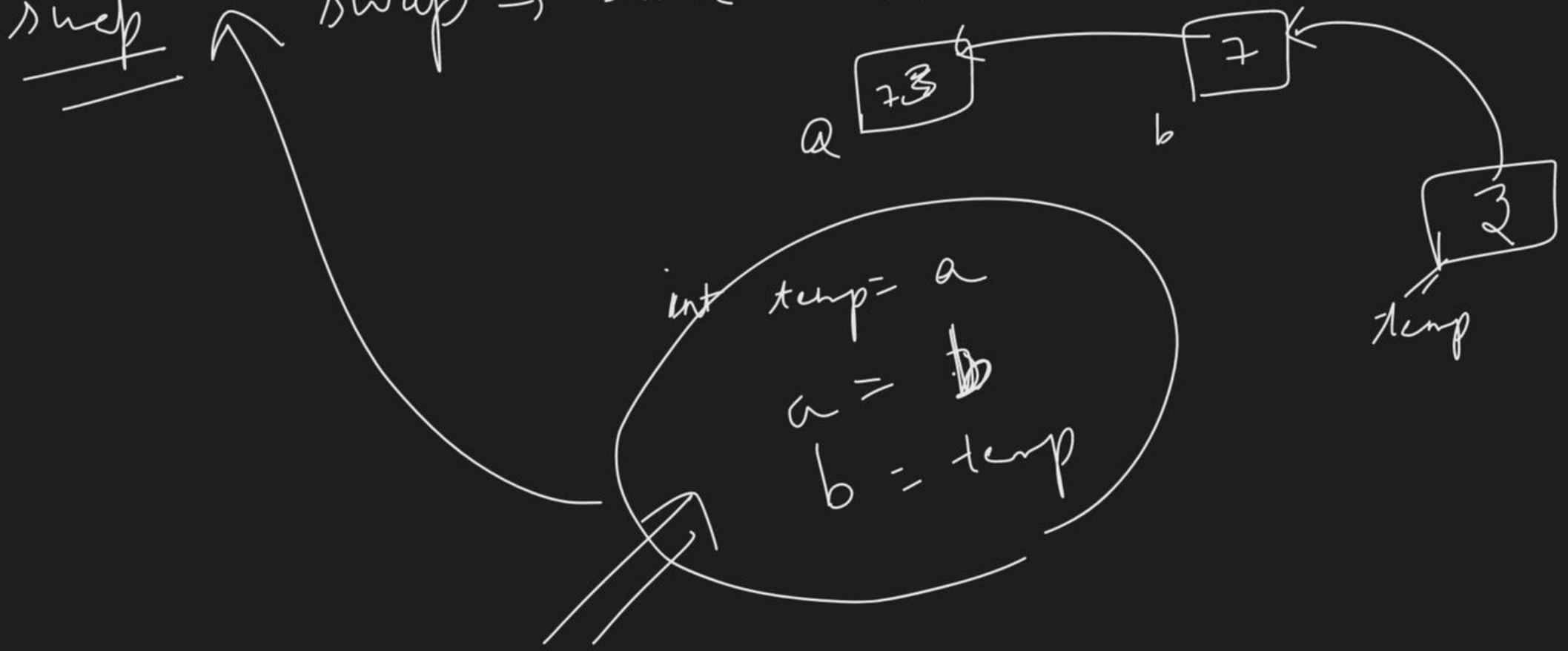
NO → help@unacademy.com

guide


i/p
3 8 6 2 9
0 1 2 3 4
Dry run
o/p
3 8 1 2 9
i=0
j=n-1
i++
j--
i=0, j=4
swap (arr[0], arr[4])
i=1, j=3
swap (arr[1], arr[3])
i=2, j=2
Jakna rukna
i>=j

swap

swap → Khud Ka likhna



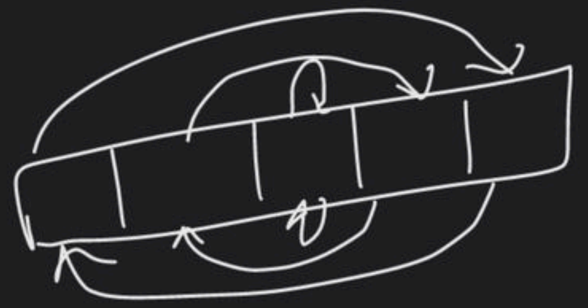
```
int i = 0 ← starting index
int j = n - 1 ← ending index

while (i < j)
{
    swap(arr(i), arr(j))
    i++
    j--
}
```

same o/p

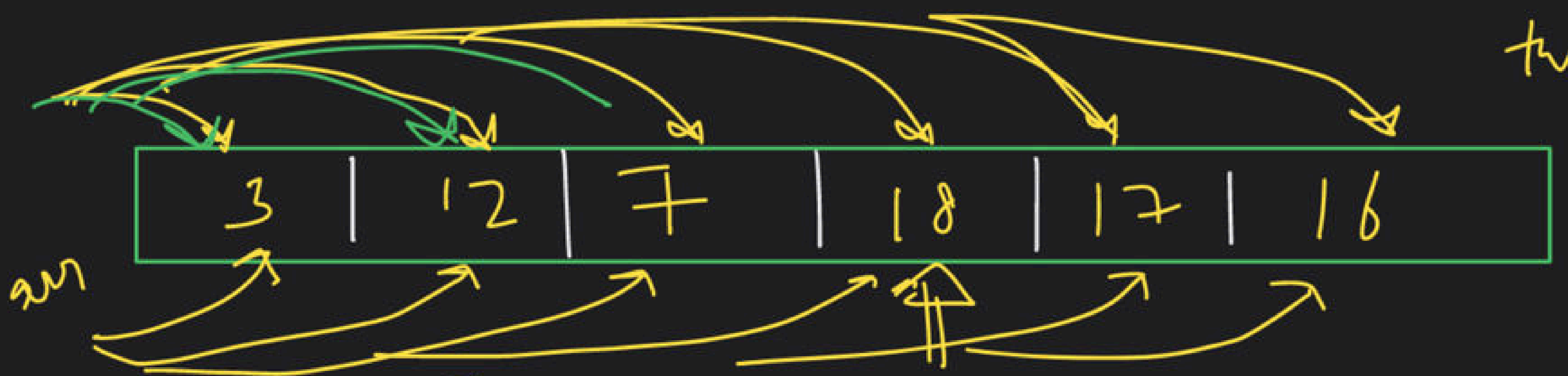
Kadane

while (i <= j)



target = 16

find max^m no in an array?



int maxi = INT_MIN;

```
for( i -> 0 < n )
{
    ans = max(ans, arr[i])
}
return ans;
```

maxi

ans

int -> [INT_MIN, INT_MAX]

predefined

max(a, b) -> maxn

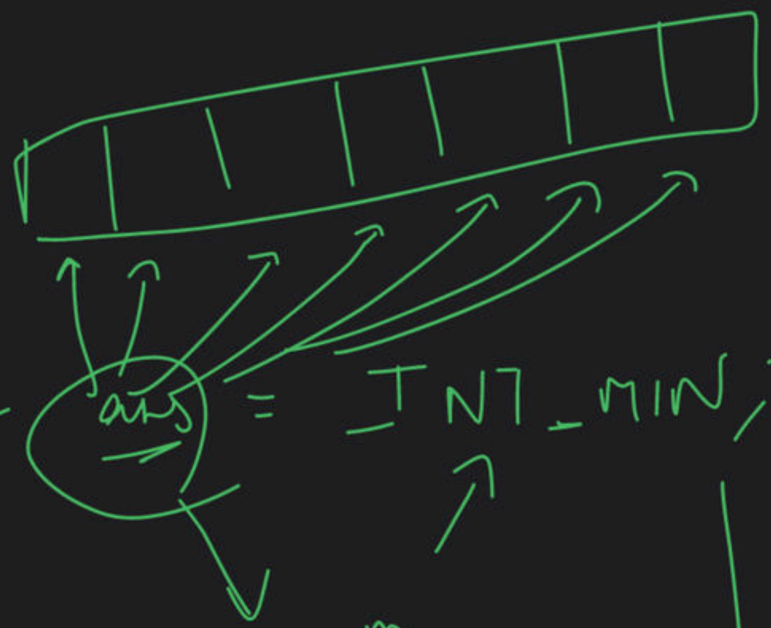
swap()

H/w

XOR -> swap?

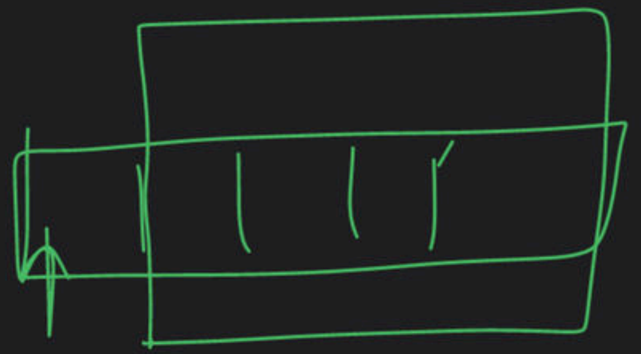
+ . -> swap

H/w



6 note

(2)Rs (5)Rs (10)Rs (20)Rs (100) (200)



compare

int ans = arr[0]

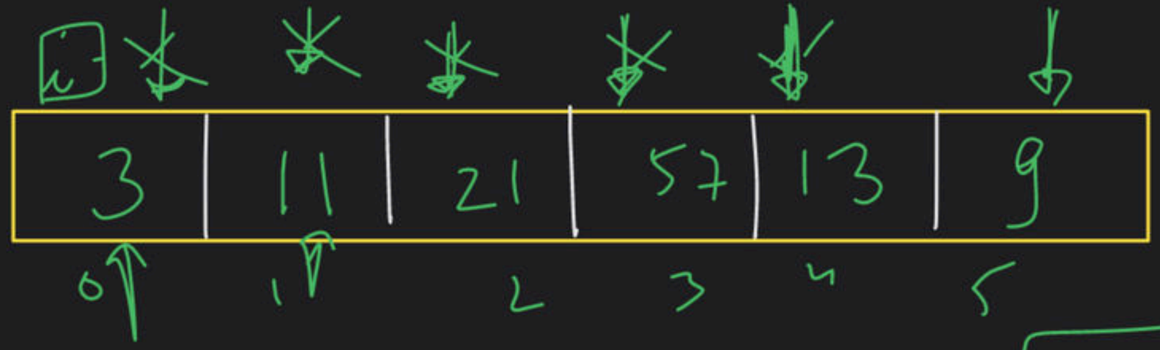
for(i → <n)

{

}

max (a , b)

```
int getmax ( int a , int b)
{
    if ( a > b)
        return a;
    else
        return b;
}
```



DRY RUN

INT_MIN

→ int ans = -INT_MIN (MAX)

ans = -INT_MAX

//include <limits.h>

i = 0

ans = max(ans, arr[0]) = max(INT_MIN, 3) = 3

i = 1

ans = max(ans, arr[1]) = max(3, 11) = 11

i = 2

ans = max(11, 21) = 21

i = 3

ans = max(21, 57) = 57

i = 4

ans = max(57, 13) = 57

i = 5

ans = max(57, 9)

i = 6 = 57

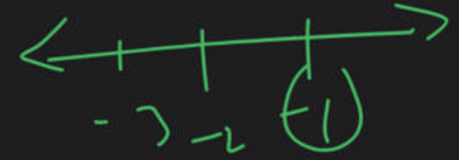
H/w find minm in an array

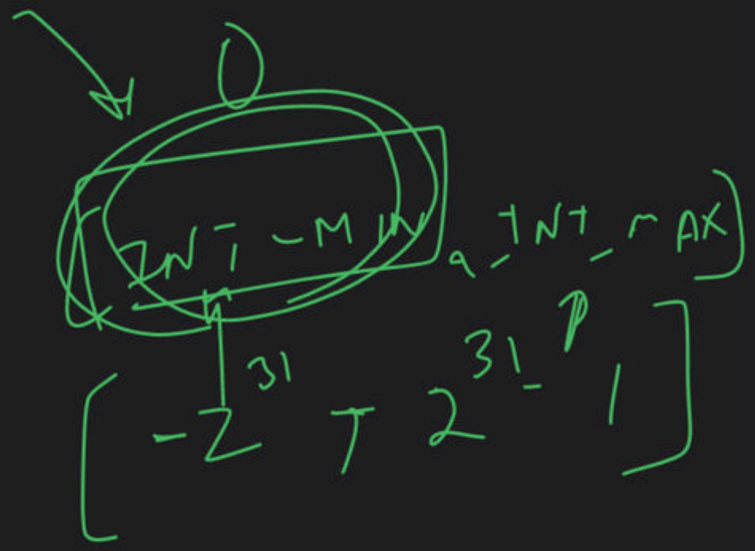
maxm

-1	-2	-3

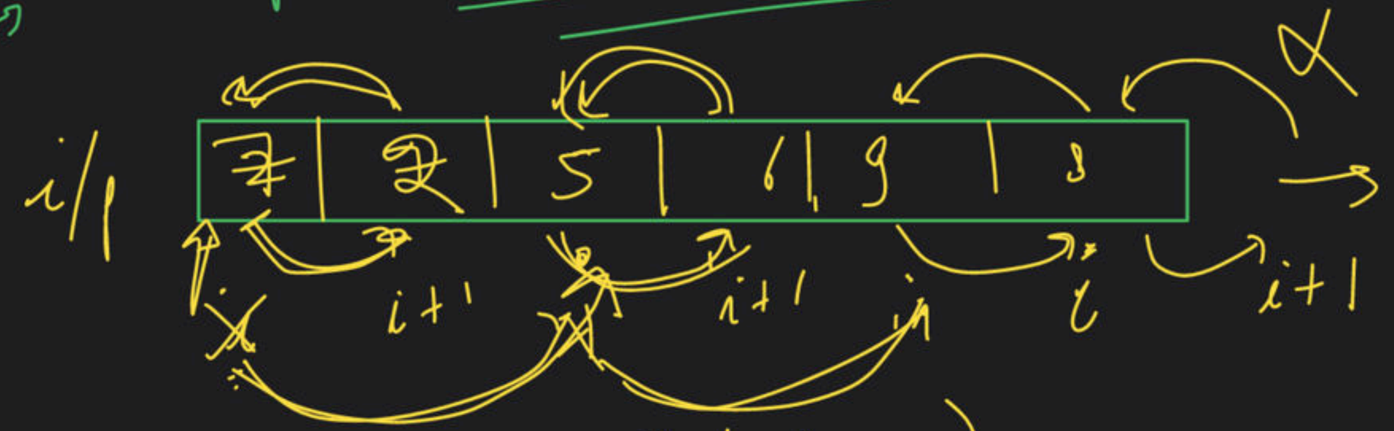
ans = -1

ans = 0





swap alternates in an array



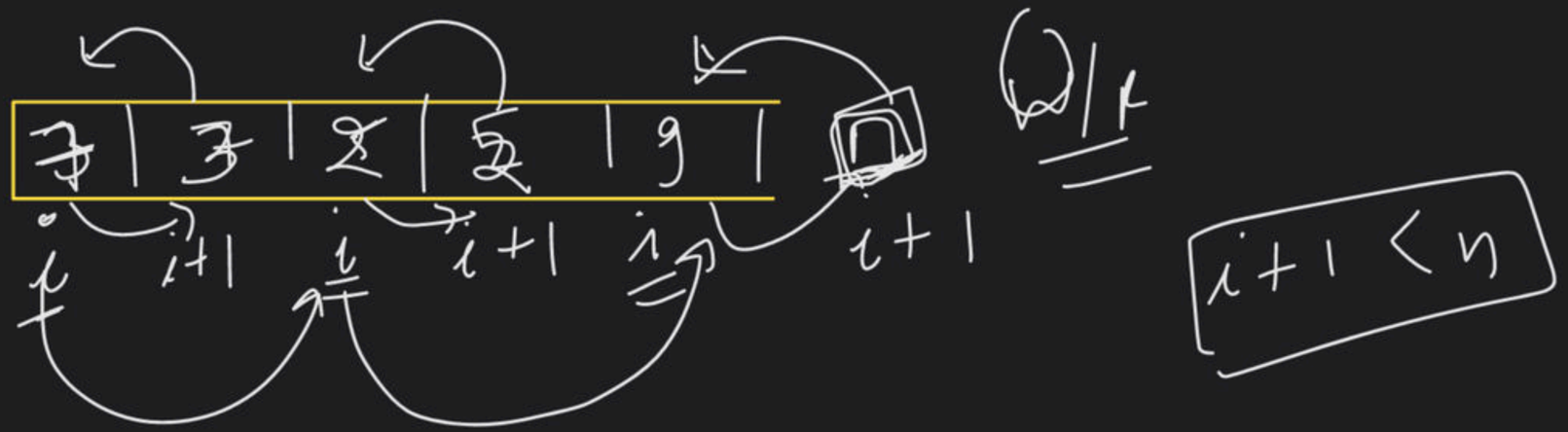
```
while ( i < n )
    if ( i+1 < n )
        swap (arr[i], arr[i+1])
    i = i + 2
```

2 checks - Observation

$i, i+1$ → swap

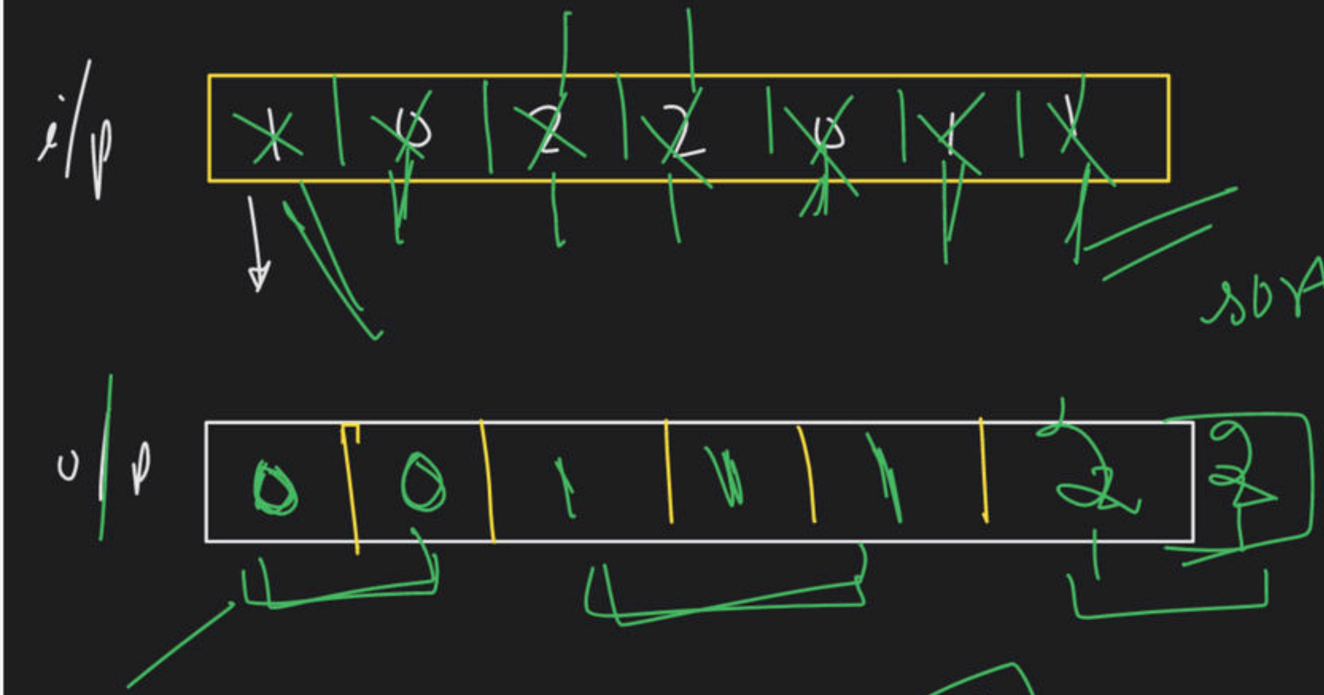
$i = i + 2$

Thus



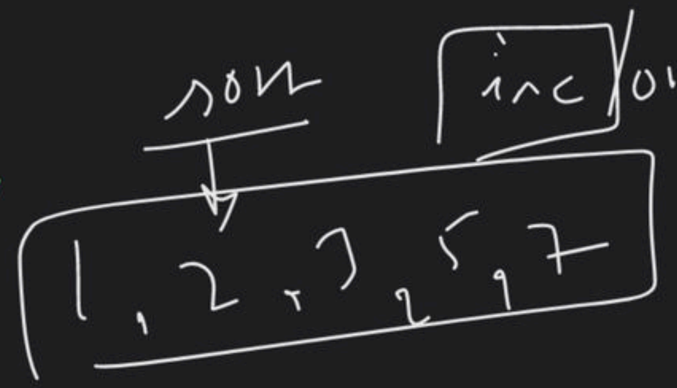
$i+1 < n$

→ Sort an array of 0's, 1's, 2's



3, 5, 2, 1, 7

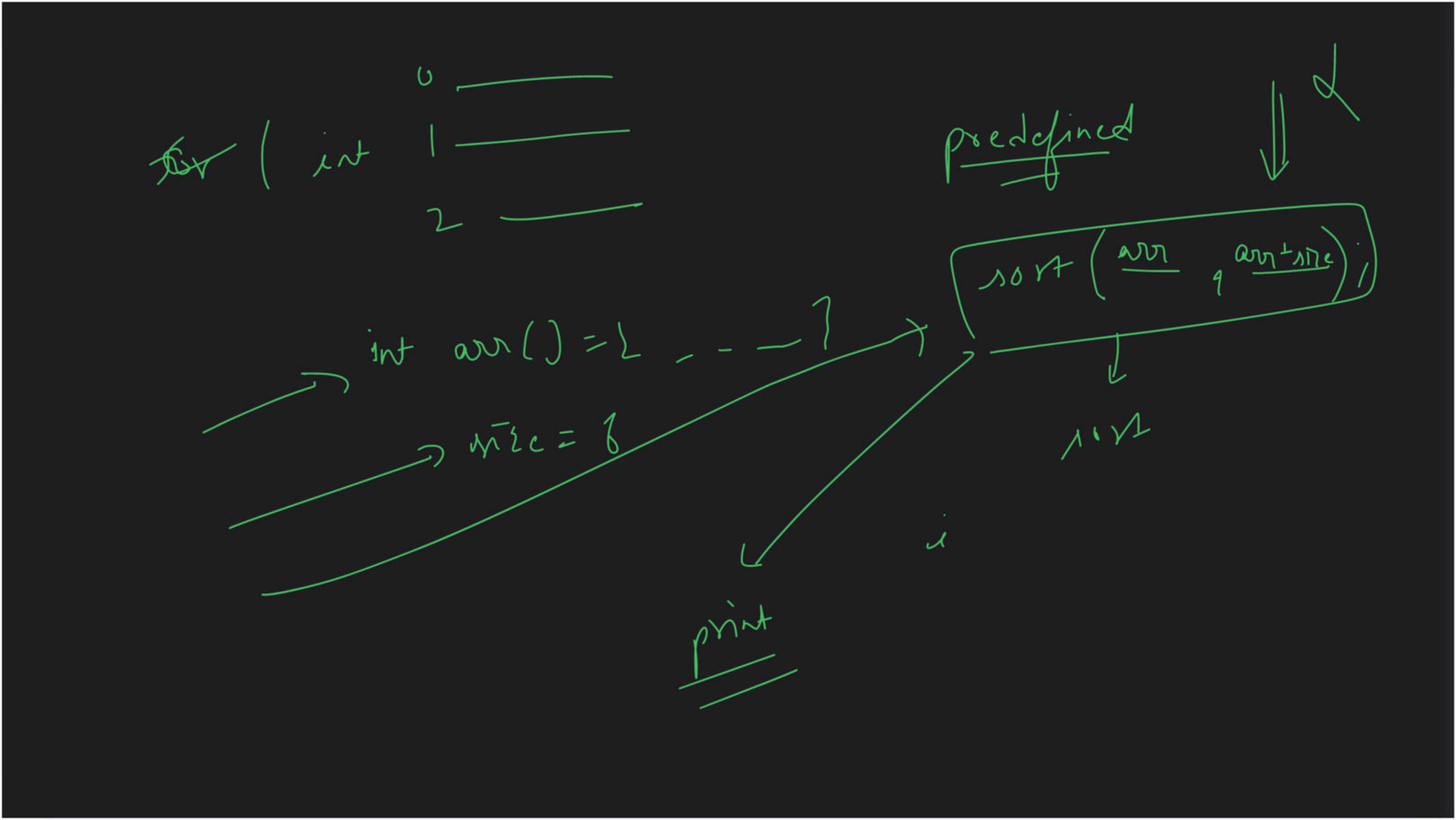
sort( );

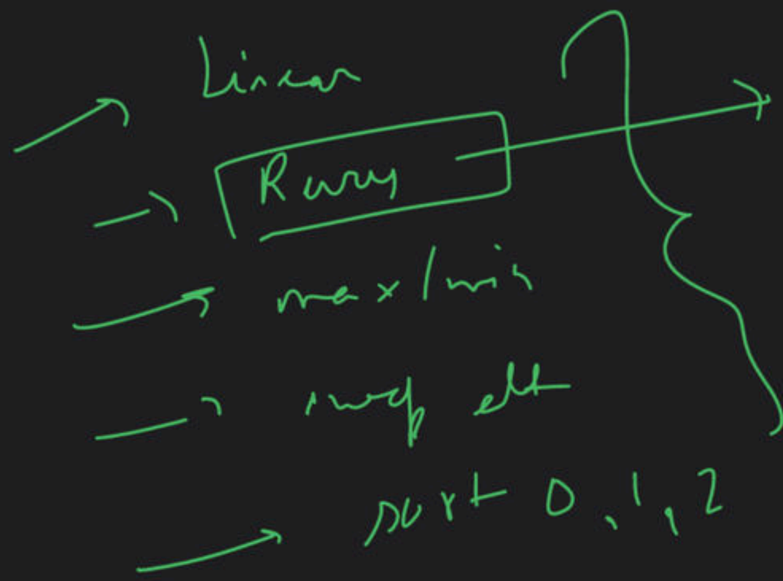


approach:

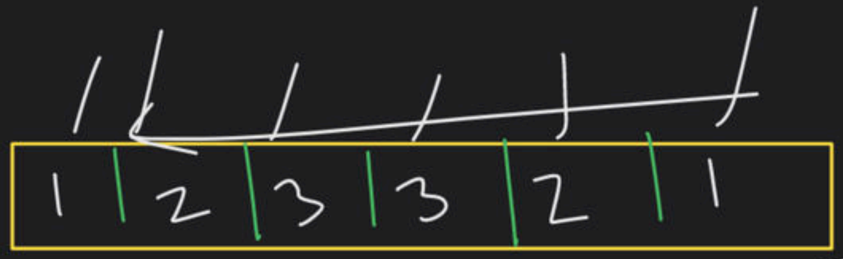
int countOne = 0, countZero = 0, countTwo = 0;

for(int i → 0 - < n)
{ if(arr(i) == 0) countZero++ else if(arr(i) == 1) countOne++ else countTwo++ }


for (int
0
1
2
int arr[] = {
size = 6
predefined
sort (arr , arr+size);
print



→ Check Palindrome



palindrome

Yes/NO

Palindrome

madam

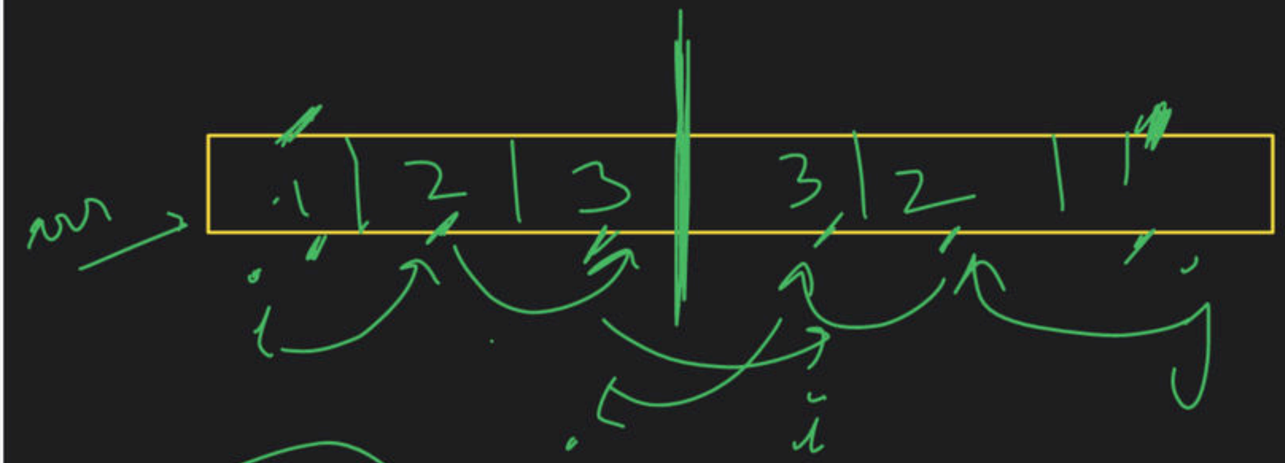
rabbar

papa

LOL

OYO

DALDAL



T/F

i = 0
j = n - 1

PAUSE

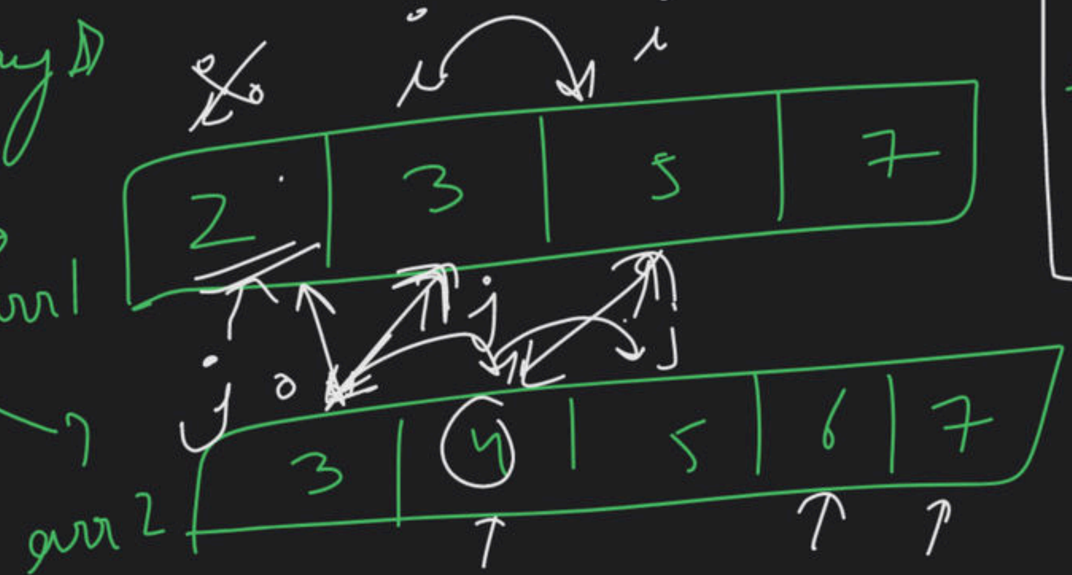
```
public (arr, n)
{ while( i<j )
  { if( arr[i] == arr[j] )
      i++
      j--;
    else
      return false;
  }
return true;
```

```
if( arr[i] == arr[j] )
    i++
    j--;
```

i >= j → return

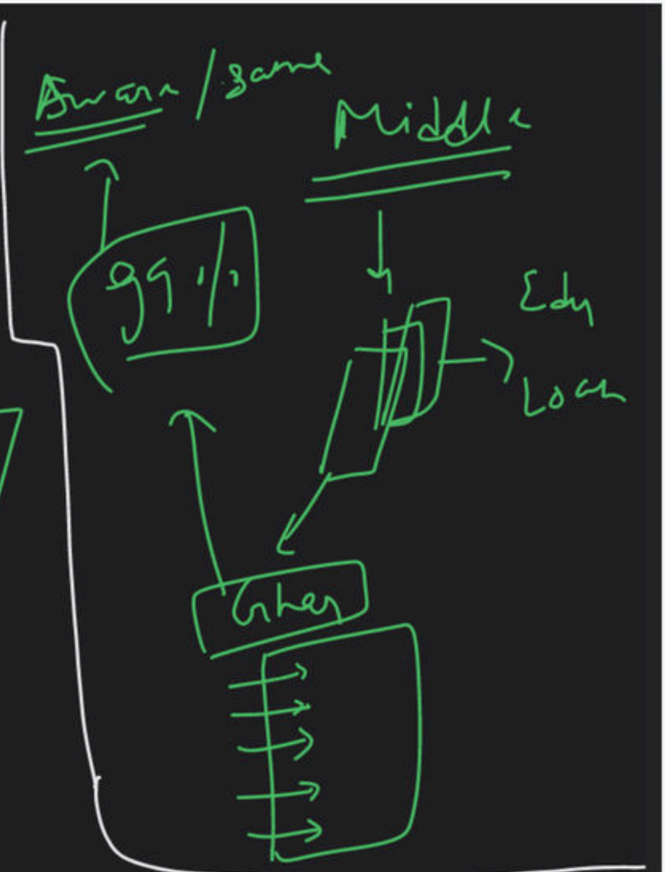
→ 2 sorted arrays

i/p →



→ Union :- {2, 3, 4, 5, 6, 7}

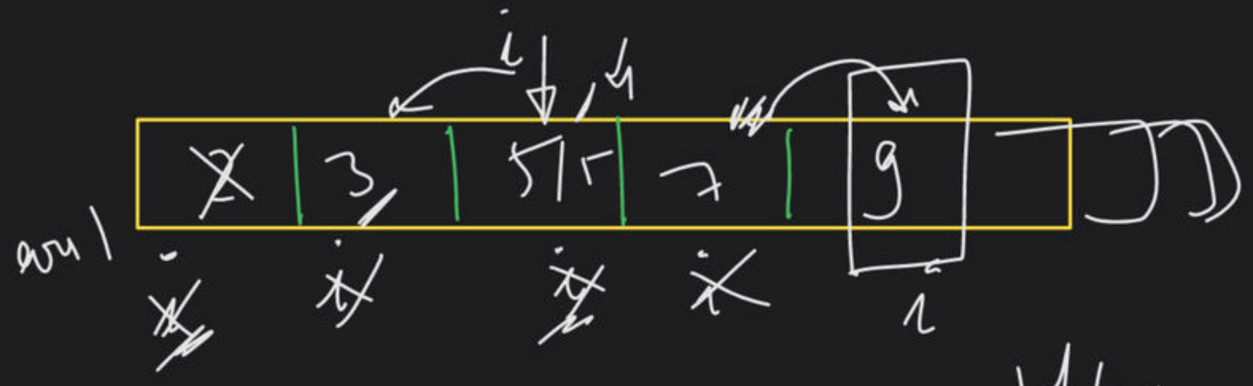
arr3 = {2, 3, 4

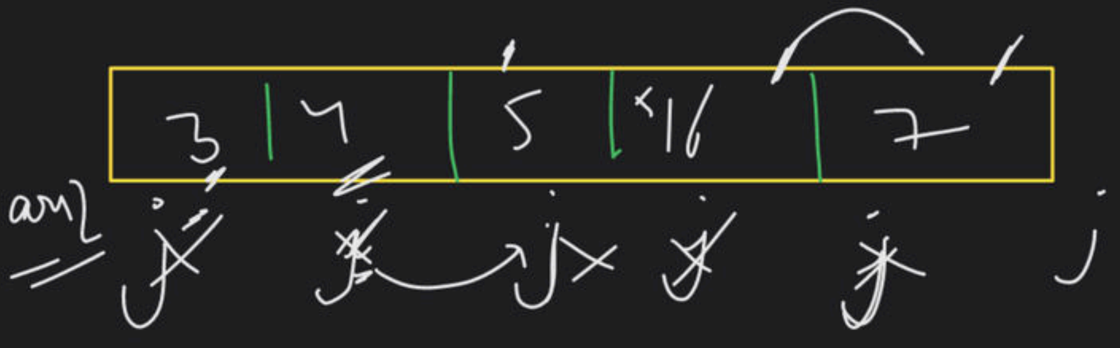


= → i++, j++

arr[i] > arr[j] → j++

arr[i] < arr[j] → i++





H/w

code

2 < 3

arr(i) < arr(j) → i++

3 == 3

arr(i) == arr(j) → i++, j++

5 > 4

arr(i) > arr(j) → j++

Approach :-

↳ default

Duplicates

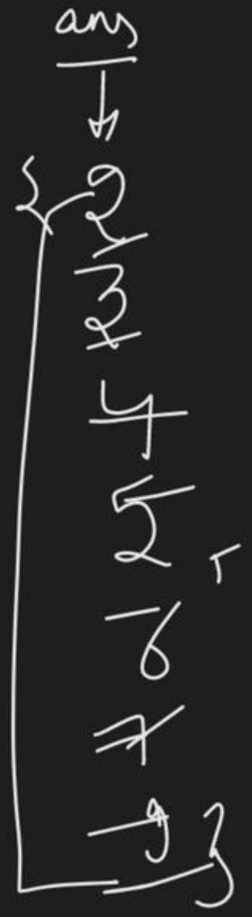
Hashmap

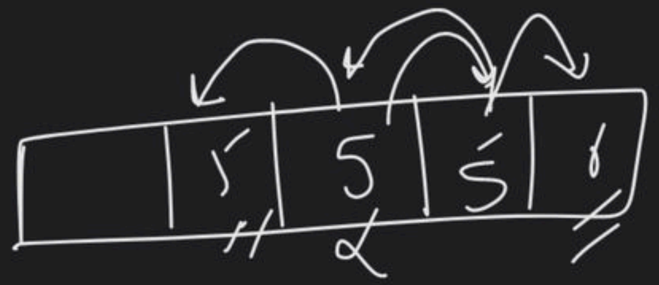
explore

set

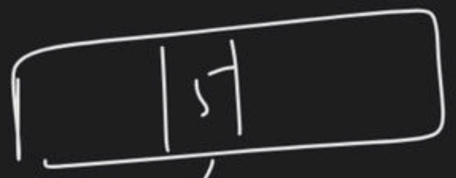
size

arr





Instruction



Logic

Code

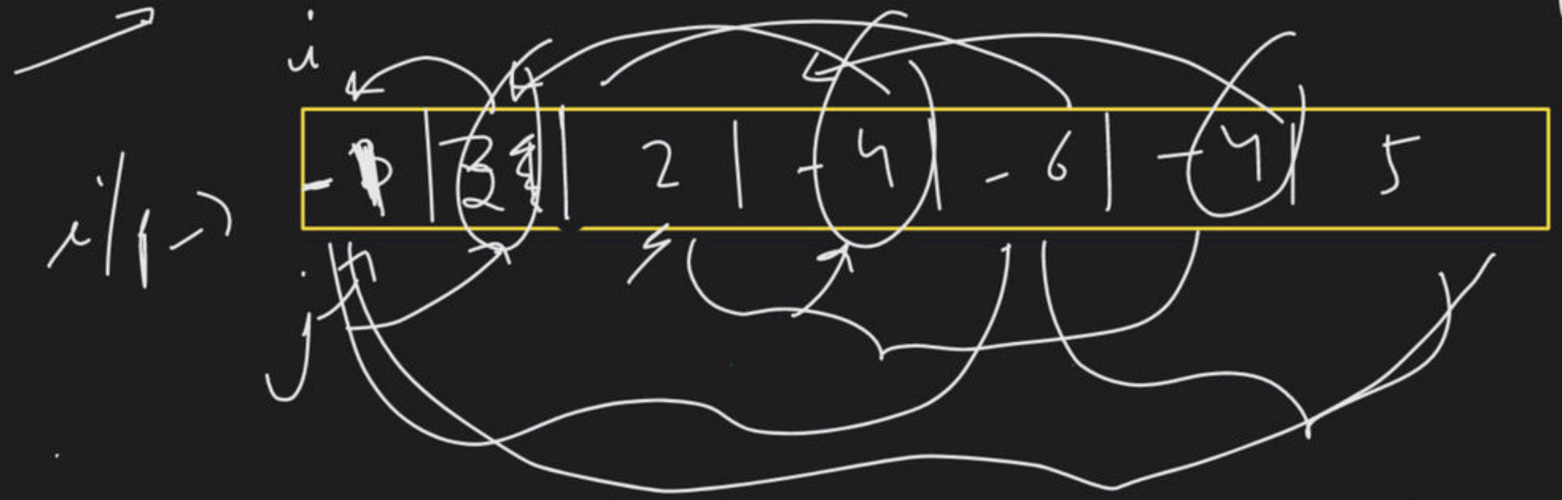
i-1

i

if(i+1<n)

if(i-1>=0)

move -ve no to one side of array

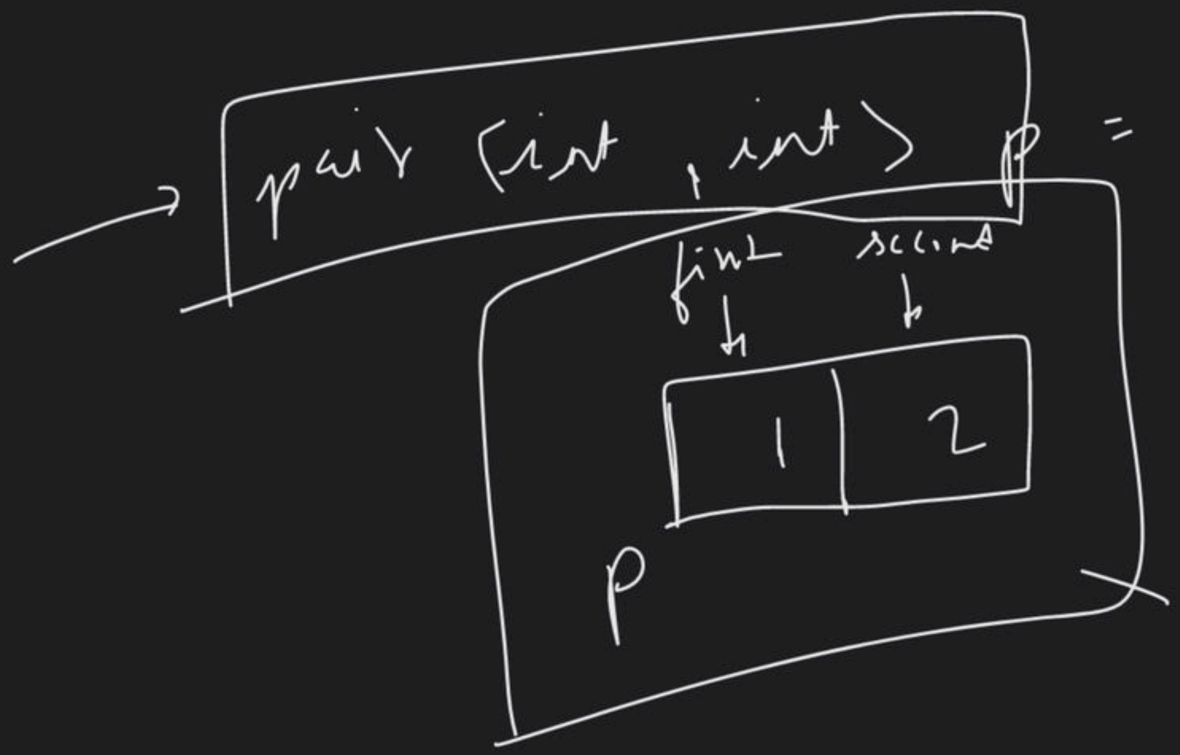


swap

arr[i] , arr[j]

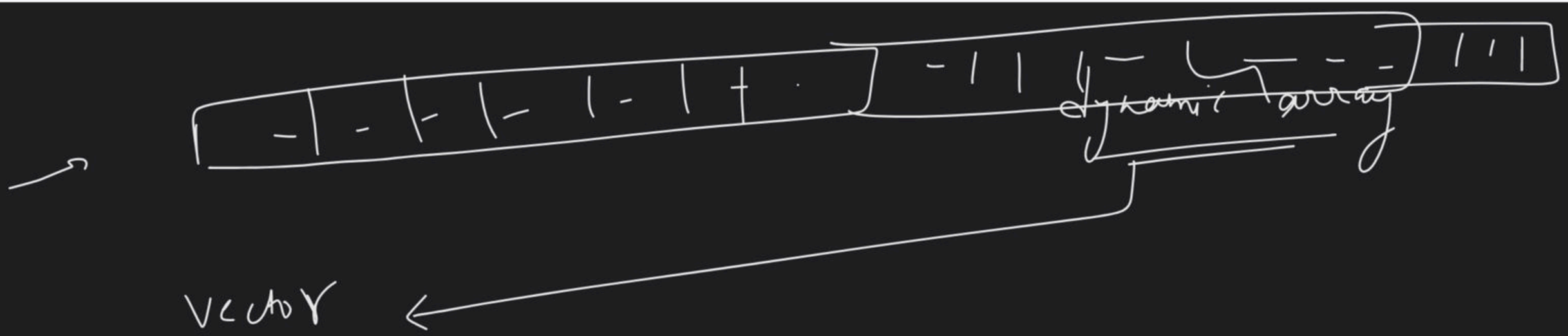
```
for ( int i=0  -> <n)
{   if (arr[i] <0)
    { swap (arr[i] , arr[j])
      j++
    }
}
```

arr[-1] -> x



pair<int, char>

pair<RAJA, RAJA>


dynamic array
vector

→ KADANE's Algo:-

→ find duplicate in an array of (N+1) integers

arr[] = {1, 2, 3, 3, 4}

{1, 2, 3, 4}

cond

↓

value

1 - N

repeat

↓

know

{1, 2, 3, ... x ... x, N}

ans

sum1 ← {1, 2, 3, 4, 5, 6, 7, 7, 8}

sum2 ← {1, 2, 3, 4, 5, 6, 7, 8}

ans = sum1 - sum2

sum

1, 2, 3, 4 - - - N → $\frac{n*(n+1)}{2}$

int sum1 = 0

→ for (int i=0 → <n)

{ sum1 = sum1 + arr[i];

}

int sum2 = n*(n+1) /2;

sum1 - sum2

int ans =

return ans;

1 —— N

{2, 2, 2, 2, 2}

< < < <

i/p

INT_MIN

sum > range

Google

Kadane's algo:-

[Largest sum Contiguous Subarray]

-2	-3	4	-1	-2	1	5	-3

= 7

Brute force

→ search subarray sum

```
for (i = 0 → <n)
{
    for (j = i → <n)
    {

    } maxi = max(maxi, ...)
}
```

→ max^m → ans

Questions:

Find common elements in 3 sorted arrays

Find first repeating element in array

Find first non-repeating element in an array

Find subarrays with equal 0s and 1s

Find subarray with 0 Sum in an array

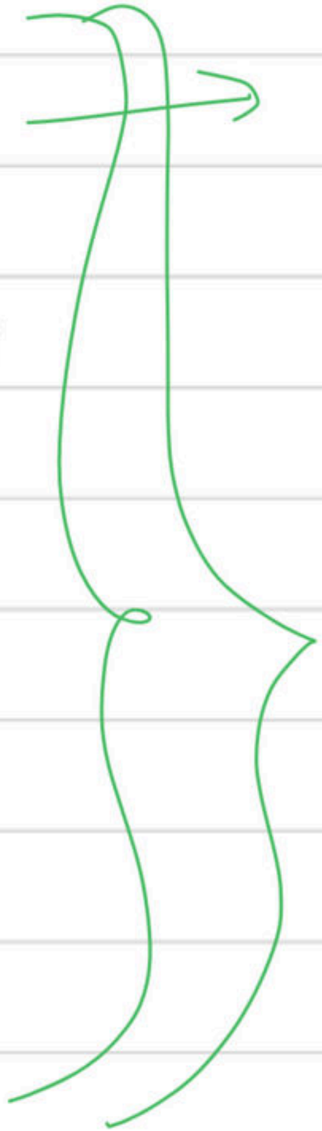
find factorial of a Large Number

Minimum Platforms Problem - GFG

Minimise the Heights II - GFG

Majority Element Problem in array - GFG

Array SubSet of another array



→ for (i = 0 → < n)

{

sum = 0

for (j = i → < n)

{

sum += arr[j]

}

// max update

}

return max

Brute force

25+ Qns

Ans

Kadane's algo

single traversal

↳ O(N)

-2	-3	4	-1	-2	1	5	-3

ab tak kaha

arr

max_so_far = INT_MIN;

max_ending_here = 0

↑ maximum sum till ith index

for (i=0 → <n)

{

→ max_ending_here += arr(i)

→ if (max_so_far < max_ending_here)

max_so_far = max_ending_here;

if (max_ending_here < 0)

max_ending_here = 0

}

return max_so_far;

0+(-2)

→ -2

-2	-3	4	-1	-2	1	5	-3

i, i = 0

maxSF = INT_MIN,
maxEH = 0

→ maxEH = 0 + (-2) = -2 < 0

→ check if my is Job

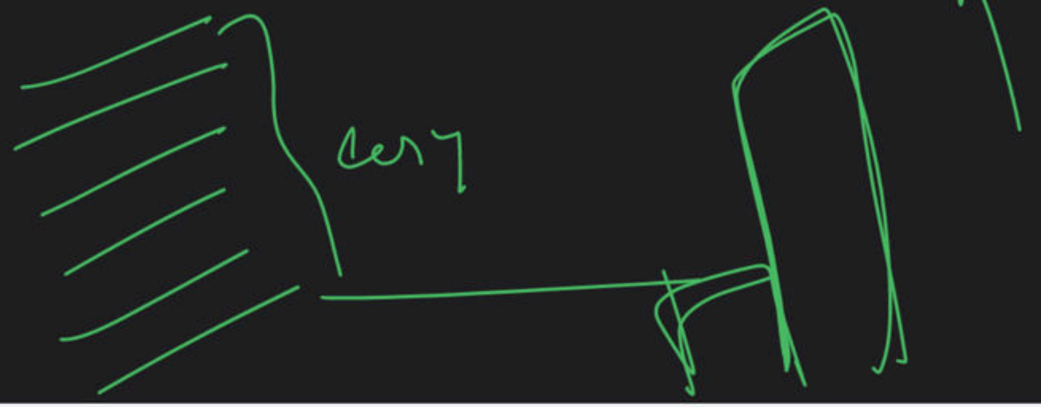
→ maxSF = max(INT_MIN, -2) = -2

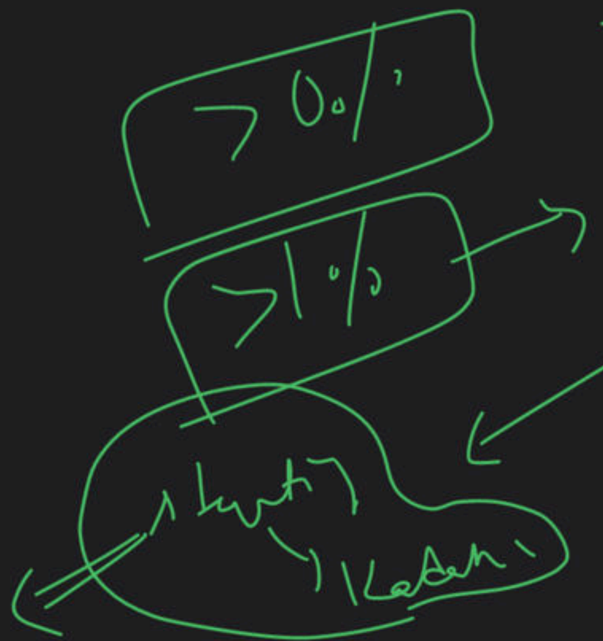
→ maxEH = 0

V.UUVUZY

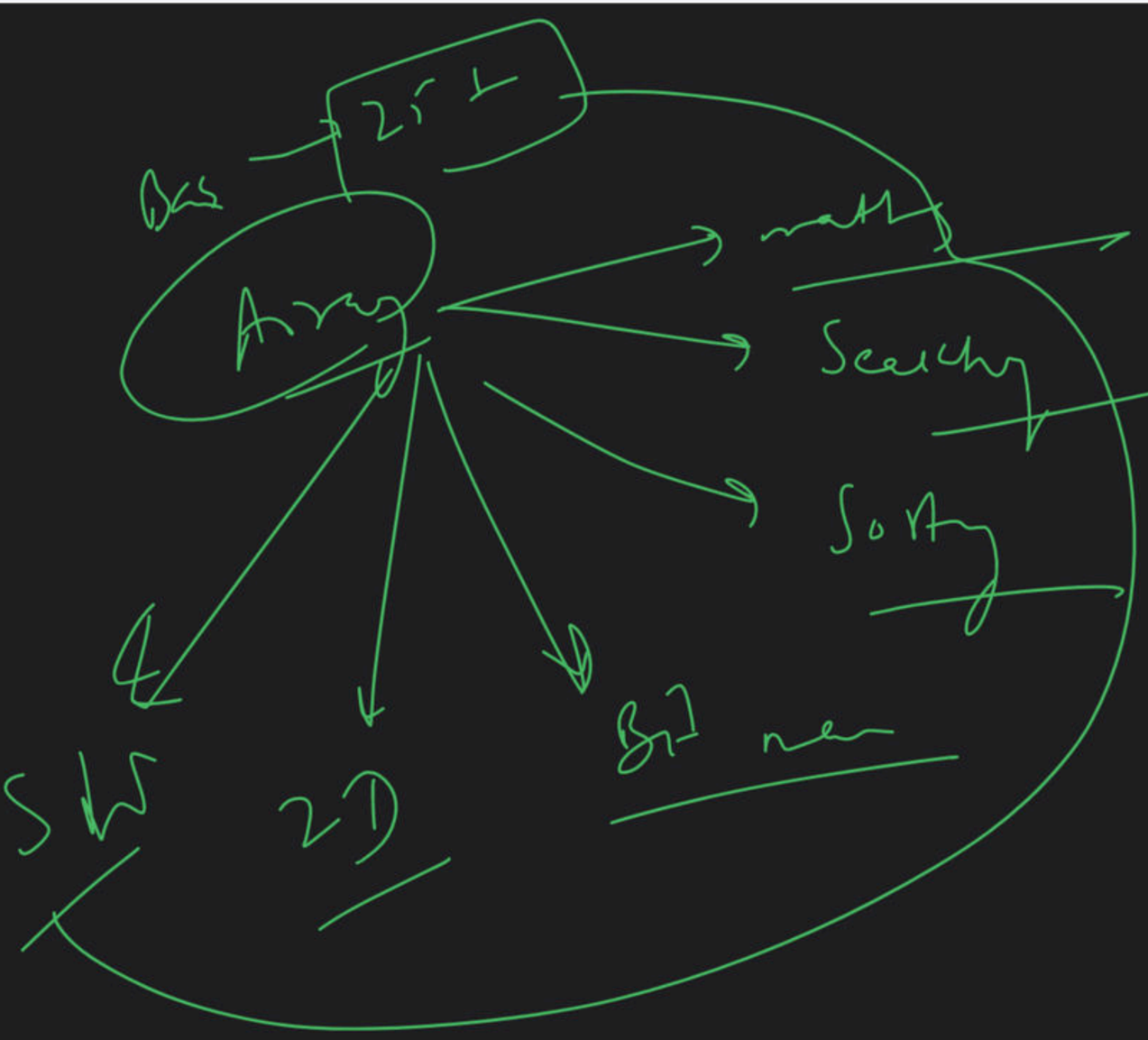
i = 1

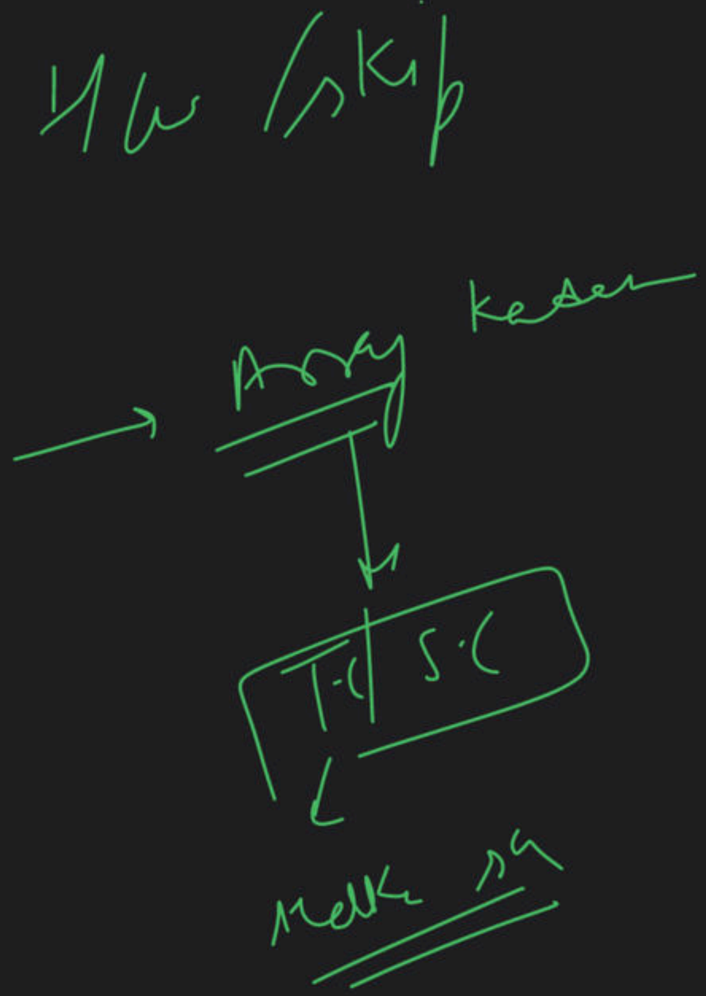
↳ maxEH = 0 + (-3) = 3





HW
↳ article
↳ Kyo
↳ Kurt
↳ alternate optimization


Array
Searching
Sorting


Array