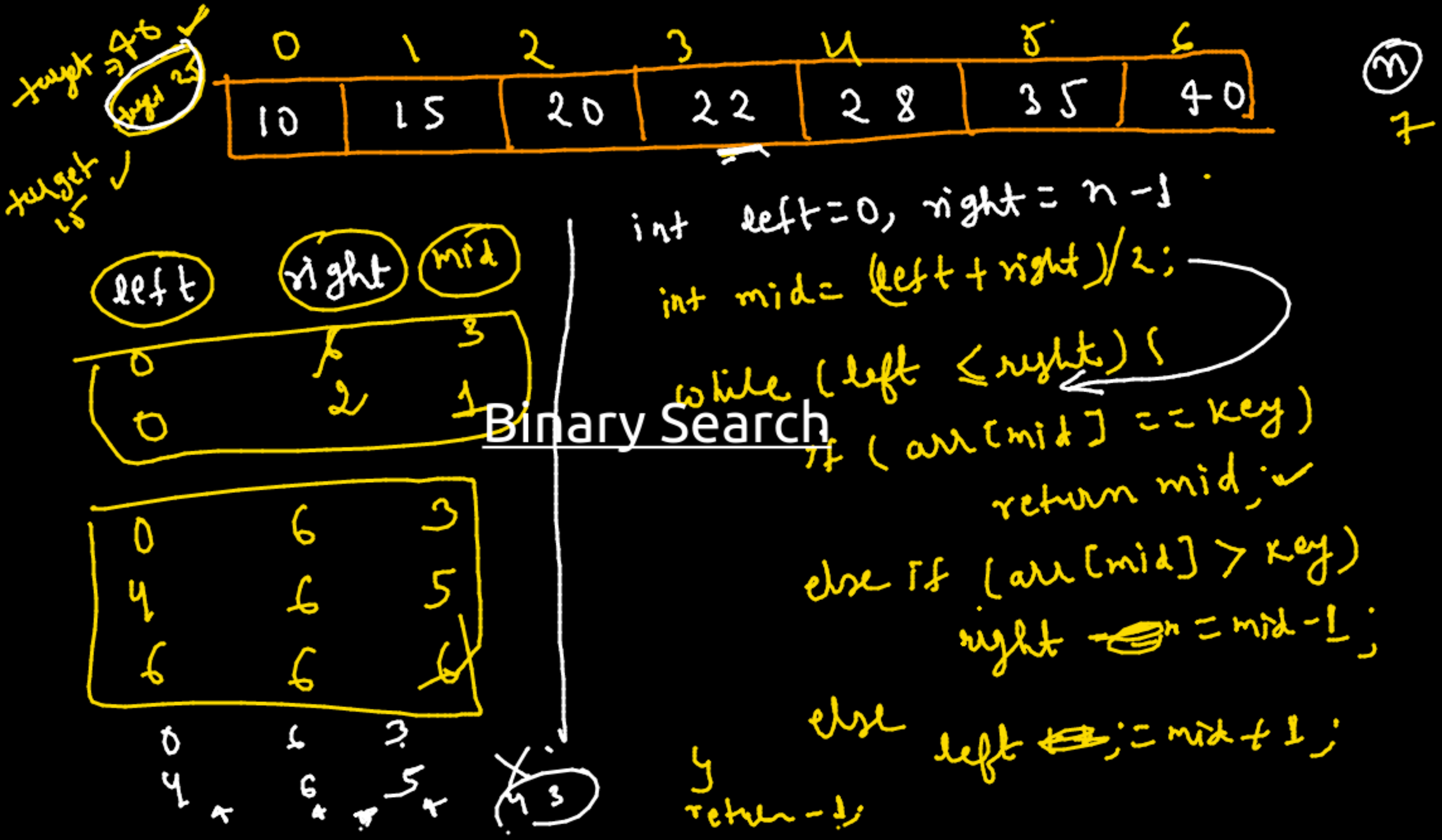
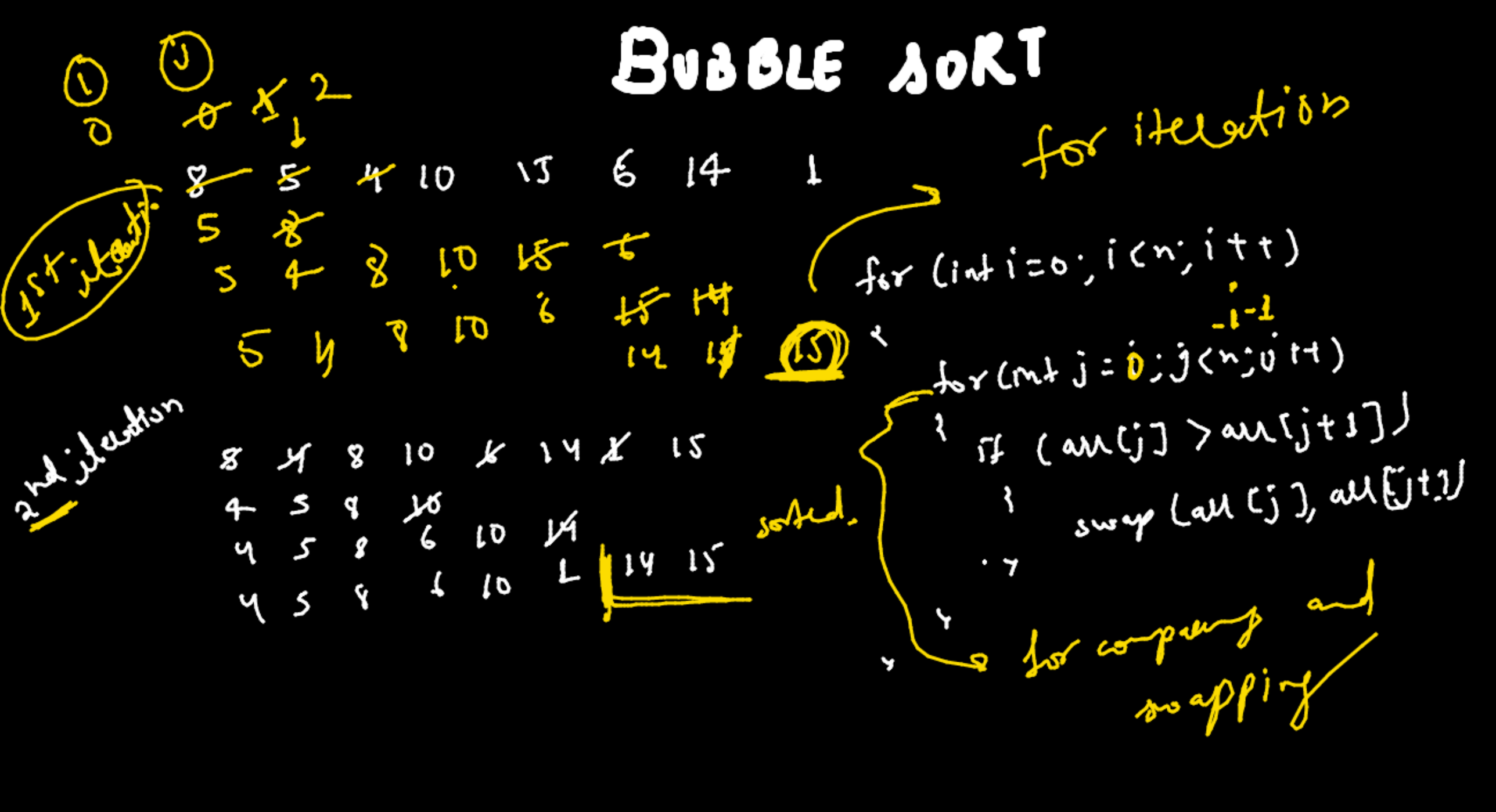
Binary Search

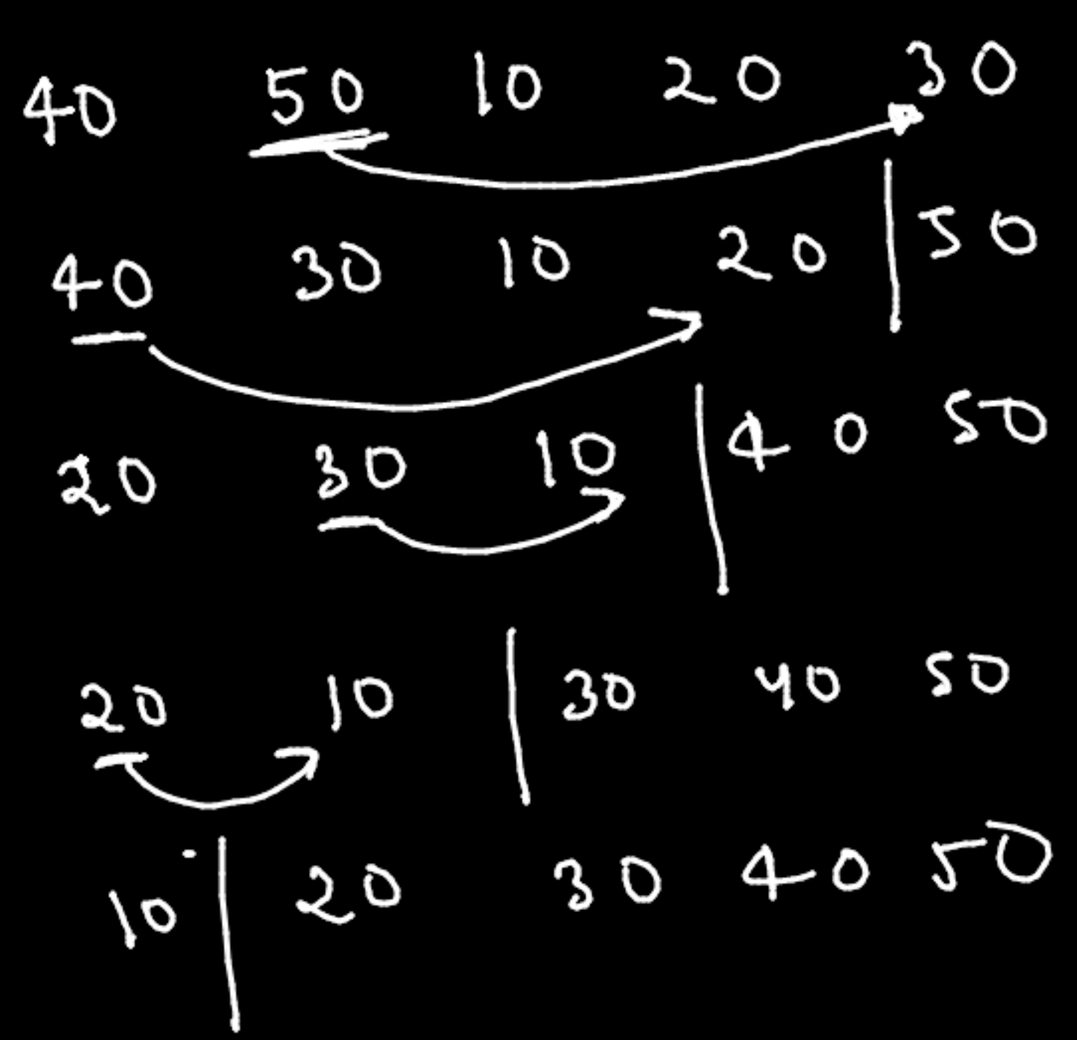


BUBBLE SORT

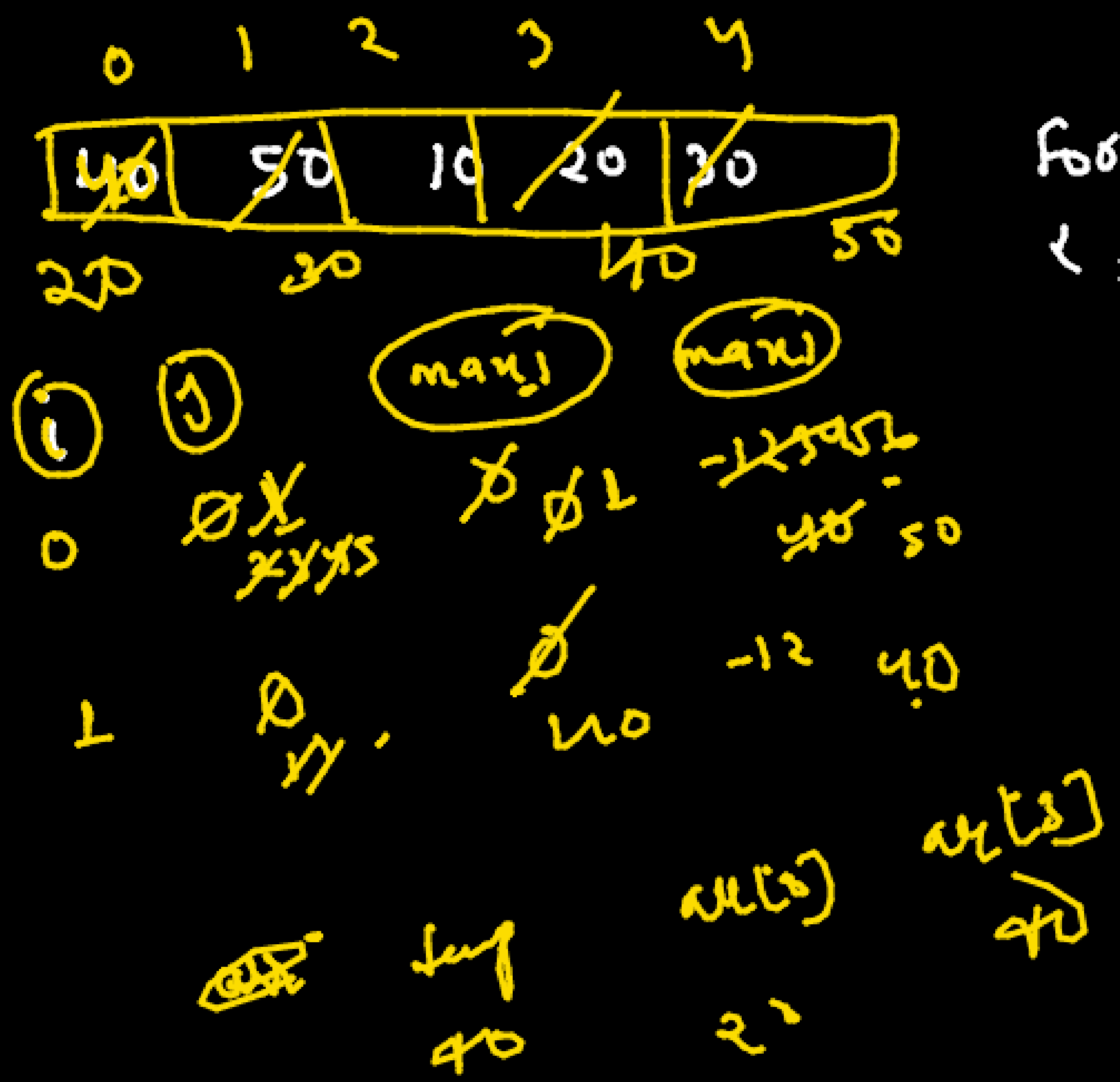

(i) (j)
0 0 1 2
1
8 5 4 10 15 6 14 1
1st iteration
5 8
5 4 8 10 15 6
5 4 8 10 6 14
5 4 8 10 14 15
for iteration
for (int i=0; i<n; i++)
i-1
for (int j=0; j<n-i-1; j++)
if (arr[j] > arr[j+1])
swap (arr[j], arr[j+1])
for comparing and swapping
2nd iteration
8 4 8 10 6 14 15
4 5 8 10
4 5 8 6 10 14
4 5 8 6 10 14
14 15
sorted.

Selection Sort

→ select and put in correct position




```
for(int i=0; i<n; i++)
{
    int max_i=0
    int maxi = INT_MIN;
    for (int j=0; j<(n-i ; j++)
    {
        if (arr[j] > maxi)
        {
            maxi = arr[j];
            max_i = j;
        }
    }
    int temp = arr[max_i]
    arr[max_i] = arr[n-i-1]
                 = temp;
```




```
for(int i=0; i<n; i++)
{ int max_i = 0;
  int maxi = INT_MIN;
  for(int j=0; j<n-i; j++)
  { if (arr[j] > maxi)
    { maxi = arr[j]
      max_i = j;
    }
  }
  int temp = arr[max_i];
  arr[max_i] = arr[n-i-1]
  arr[n-i-1] = temp;
}
```

Selection

Put maxi number in correct position.

5-0-1

Important Questions

① Rotate 0, 1, 2 — O(n) — Two pointer. // O(n)

① Sum of Two array → Concept

Push 0 to End

O(n)

for loop.


```
for(; index<n; i++)
    arr[index]=0;
```

```
int count0=0; int index=0;
for(int i=0; i<n; i++){
    if(arr[i]!=0)
    {
        arr[index] = arr[i];
        index++;
    }
    else{
        count0++;
    }
}
```

2 1 3 6 0 0 0 Push 0 to End

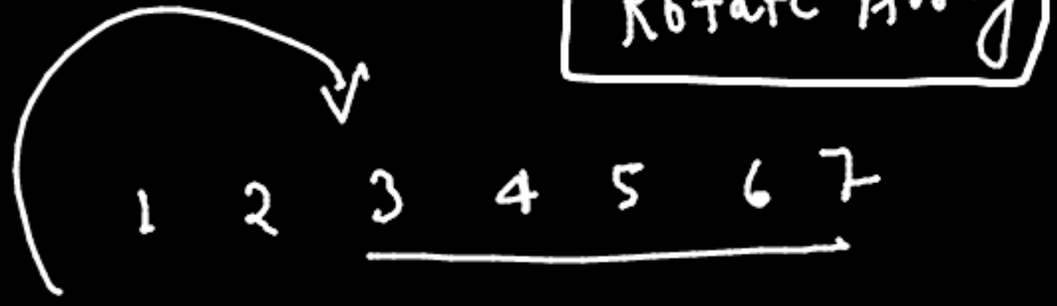
```
if
else( arr[strt] != 0)
    strt++;
else if ( arr[end] == 0)
    end++;
```

2 pointer Method

```
int strt = 0;
int end = 0;

while( strt < n  && end < n)
{
    if (arr[strt] == 0 && arr[end] != 0)
    {
        swap (arr[strt], arr[end];
        strt++; end++;
    }
```

Rotate Array



$r = 2$

↓ make copy array then modify it as per your need.

sum of two Array

10 56 →

Pushing No. into array

no of digit = 4

```
for (int i=n-1; i>=0; i--)
{
    arr[i] = num%10
    num/=10;
}
```

Sort 0 1 2

low mid high

0 1 2 0 2 0 1

int l=0, mid=0

end = n-1;

high

while (mid ≤ high)

{

switch (arr[mid])

{

case 0:

swap (arr[low], arr[mid]);

mid++; low++;

break;

case 1 mid++ break;

case 2. swap (arr[mid], arr[high])

high--;

Merge Two sorted Arrays

5 10 12 15 → i

3 8 11 → j

3	5	8	10	11	12	15

```
int i=0, j=0 , k=0;
while (i < size1 && j < size2)
{  if (arr1[i] < arr2[j]
      arr[k++] = arr1[i++];
   else
      arr[k++] = arr2[j++];
}
while (i<size1)
   arr[k++] = arr1[i++]

while (j<size2)
   arr[k++] = arr2[j++]
```

new Array