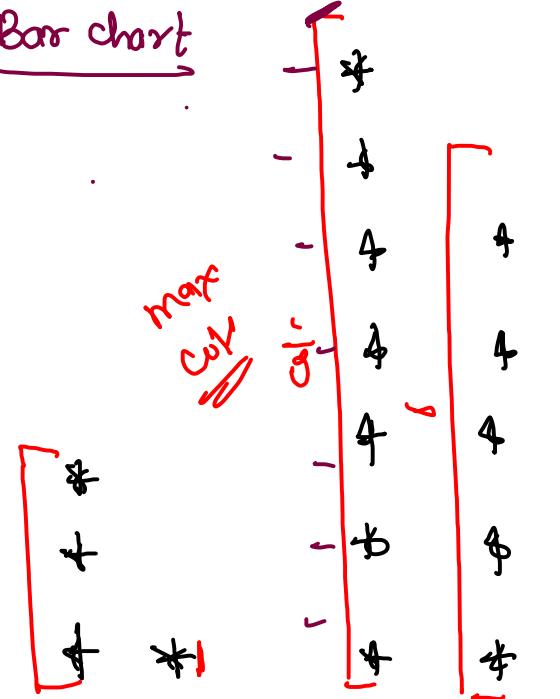


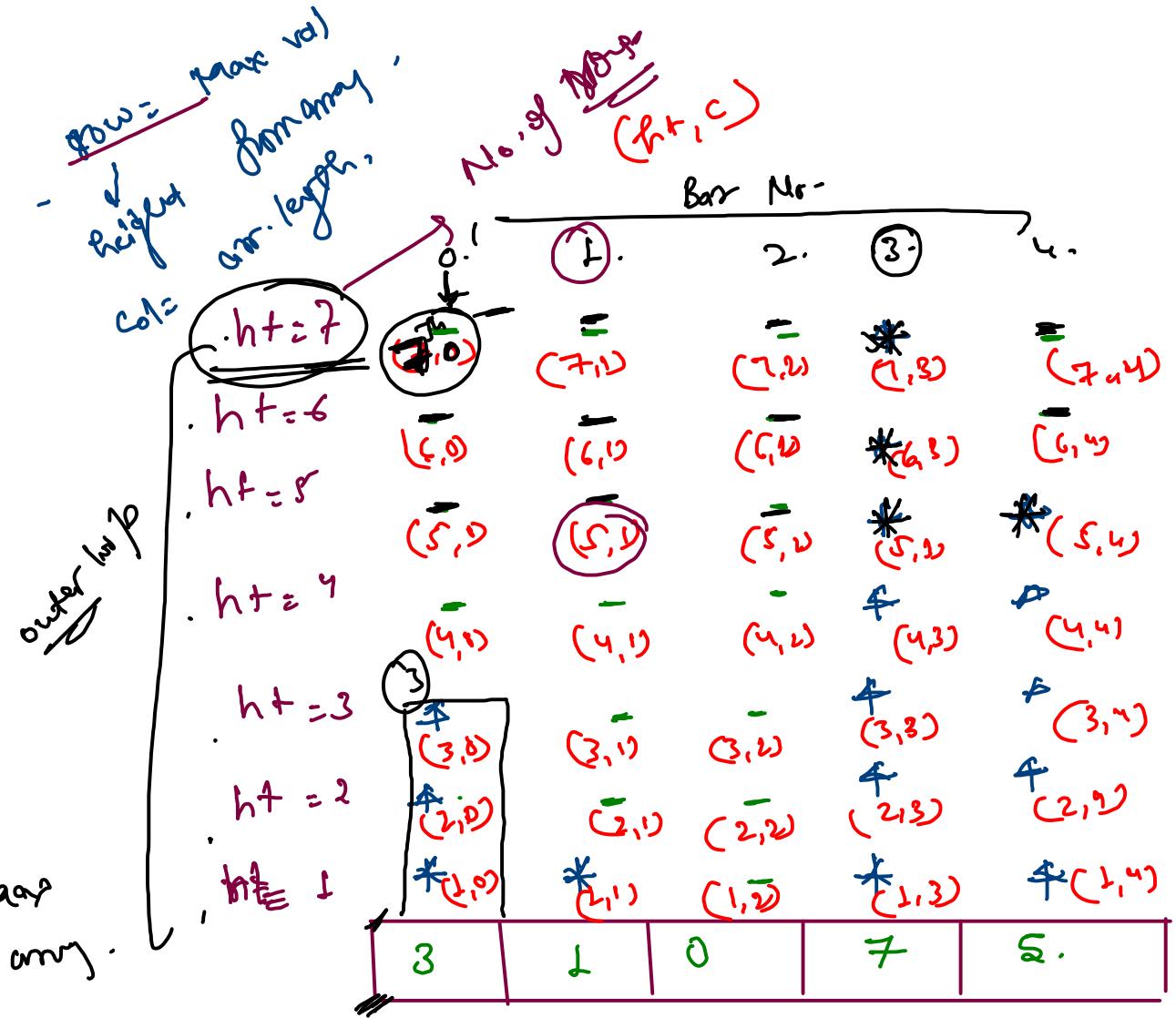
Bar chart



diff: 3
1
0
7
5

steps

→ find max
from array.



RAM → ① Stack → Primitive Data type ↪

② Heap → Non primitive data.

Data types - ① primitive D.T.

② Non primitive D.T. → Except primitive d.type, all are non primitive :-

Primitive D.T.

① Byte

② Short

③ Int

④ Long

⑤ Float

⑥ Double

⑦ Char

⑧ Boolean

```
public static Scanner scn = new Scanner(System.in);

public static void takeInput(int[] arr) {
    for(int i = 0; i < arr.length; i++) {
        arr[i] = scn.nextInt();
    }
}

Run | Debug
public static void main(String[] args) {
    int n = scn.nextInt();
    int[] arr = new int[n];
    takeInput(arr);
}
```

```
int[] arr;
```

arr = new int[5];

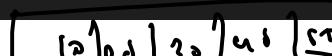
10
20
30
40

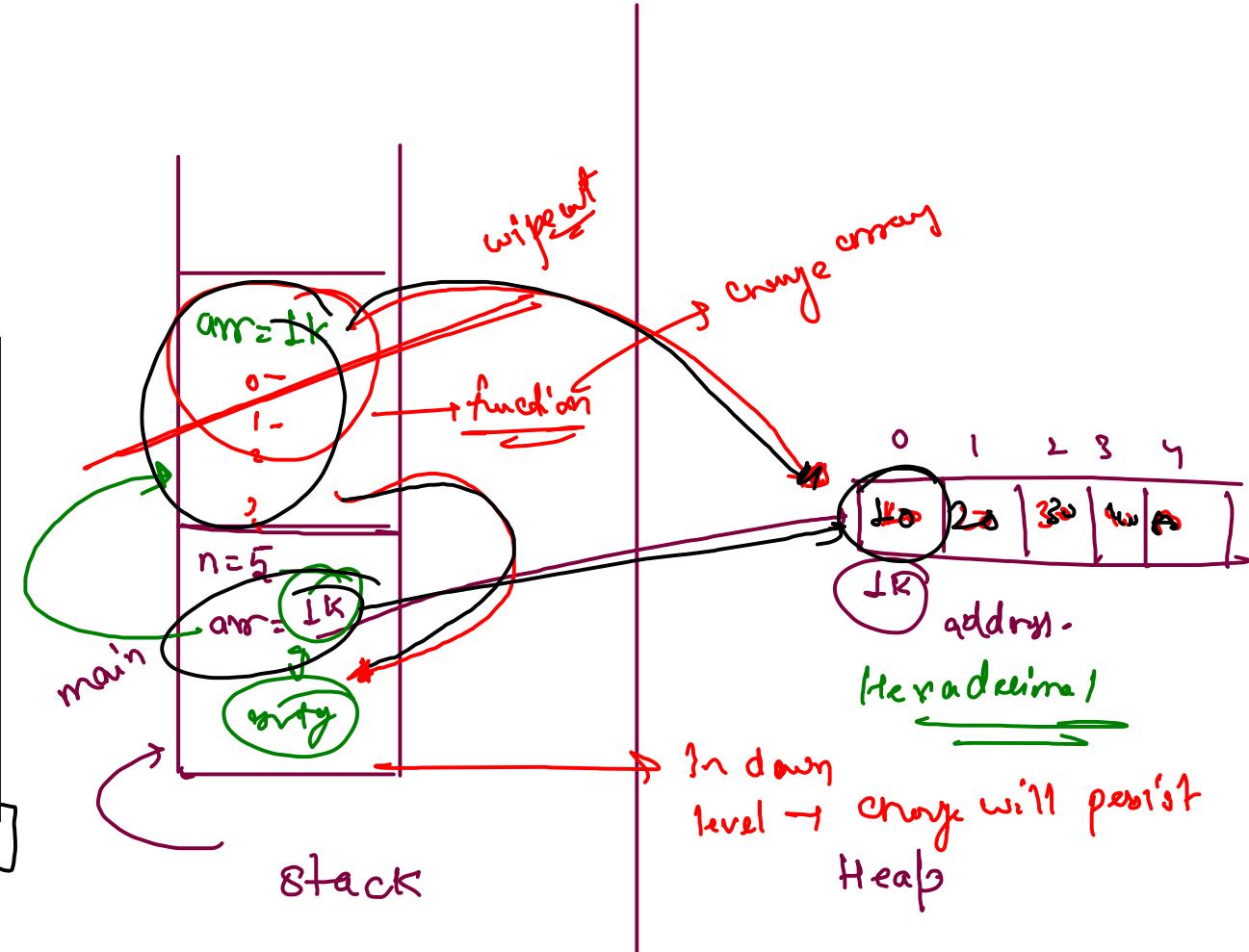
```
public static Scanner scn = new Scanner(System.in);

public static void takeInput(int[] arr) {
    for(int i = 0; i < arr.length; i++) {
        arr[i] = scn.nextInt();
    }
}

Run | Debug

public static void main(String[] args) {
    int n = scn.nextInt(); ←
    int[] arr = new int[n]; →
    takeInput(arr); →
}
} ← arr
```

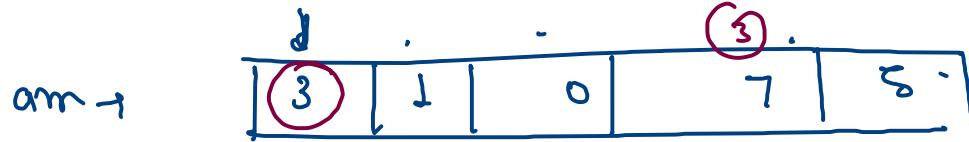




Integer.MIN_VALUE} $-\infty$

Row = Max from array] ②

```
public static void printBarChart(int[] arr) {  
    // find max from array  
    int max = max(arr);  
  
    for(int ht = max; ht >= 1; ht--) {  
        for(int i = 0; i < arr.length; i++) {  
            if(arr[i] >= ht) {  
                // print star  
                System.out.print("*\t");  
            } else {  
                // print space  
                System.out.print("\t");  
            }  
        }  
        // hit enter  
        System.out.println();  
  
        // print array  
        // for(int i = 0; i < arr.length; i++) {  
        //     System.out.print(arr[i] + "\t");  
        // }  
    }  
}
```



ht	1	2	3	4	5	6	7
ht = 7	-	-	-	*	-		
ht = 6	-	-	-	-	*	-	
ht = 5	-	-	-	-	*	*	*
ht = 4	-	-	-	-	*	*	*
ht = 3	*	-	-	-	*	*	*
ht = 2	*	-	-	-	*	*	*
ht = 1	*	*	-	-	*	*	*

Sum of two arrays.

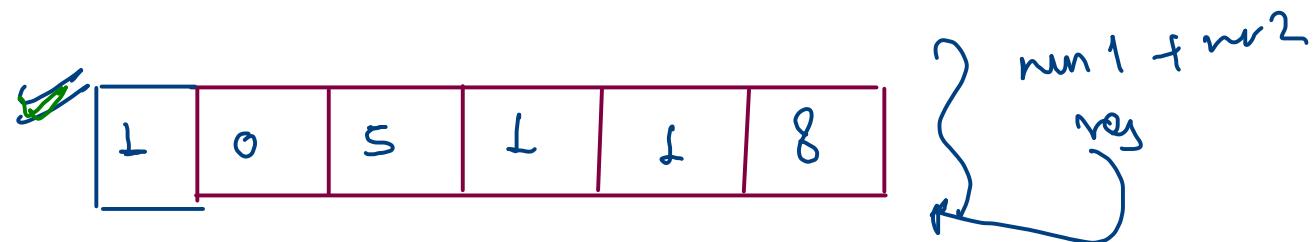
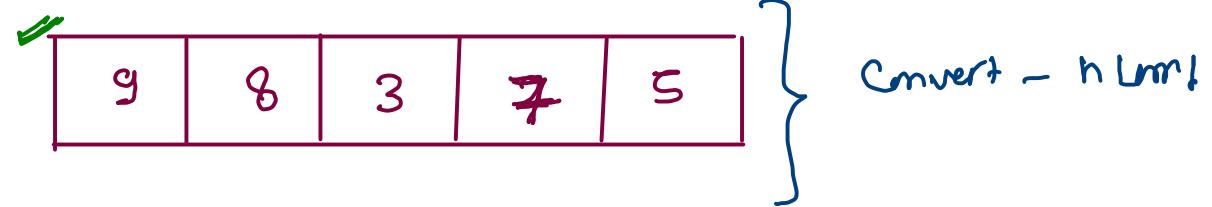
$$\begin{bmatrix} +10 \\ -10 \\ \hline 0 \end{bmatrix} \rightarrow s_1$$

$$s_2 \rightarrow \text{arr}_2$$

$n_{s_1} =$

$s_1 \geq s_2$

$n_{s_2} \rightarrow s_2 f_1$



$$\left\{ \begin{array}{l} s_2 > s_1 \\ n_{s_2} \rightarrow s_2 f_1 \end{array} \right.$$

$\text{res} = 0$
 $\left\{ \begin{array}{l} i=0; i < \text{arr.length} \\ \text{res} += \text{arr}[i] \end{array} \right.$

if $(a * e = a)$ then e is
 Identity for
 operator $*$

$$a + e = a$$

$$e = 0$$

0 is identity
 $+ \text{ operator}$.
 power = 1; $\xrightarrow{\text{To 10}}$
 quantity
 $\text{power} = 10, \text{ quantity} = 1$

$$a - e = a$$

$$e = 0$$

0 is identity
 $- \text{ operator}$.

$$a \times e = a$$

$$e = 1$$

1 is identity
 for ' \times ' operator-

1 is identity
 for division oper.

$$a / e = a$$

$$\boxed{e = 1}$$

$$\min(a, e) = a$$

$$e = ?$$

$$\boxed{e = +\infty}$$

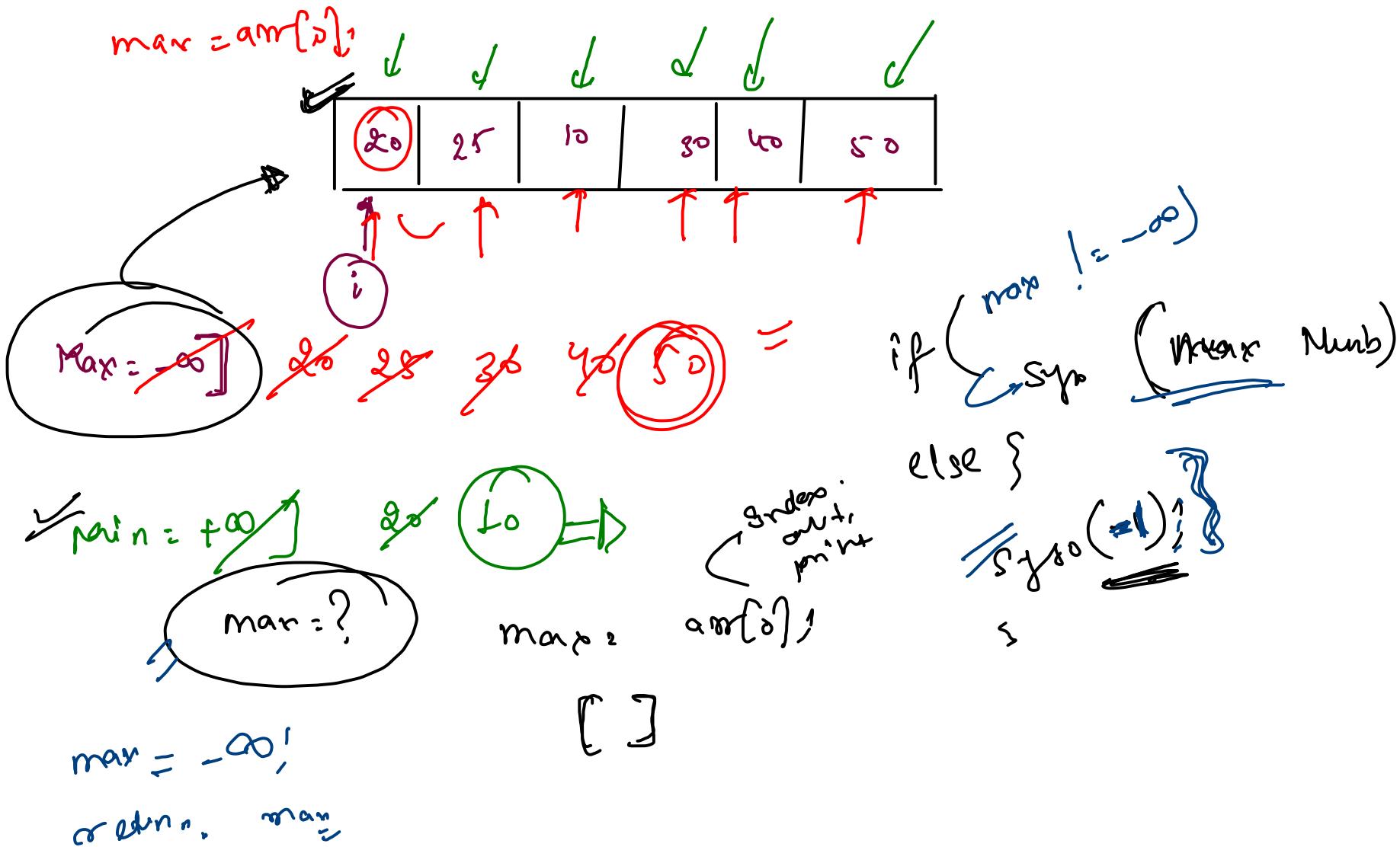
$$e = \text{Integer.MAX_VALUE}$$

$$\max(a, e) = a$$

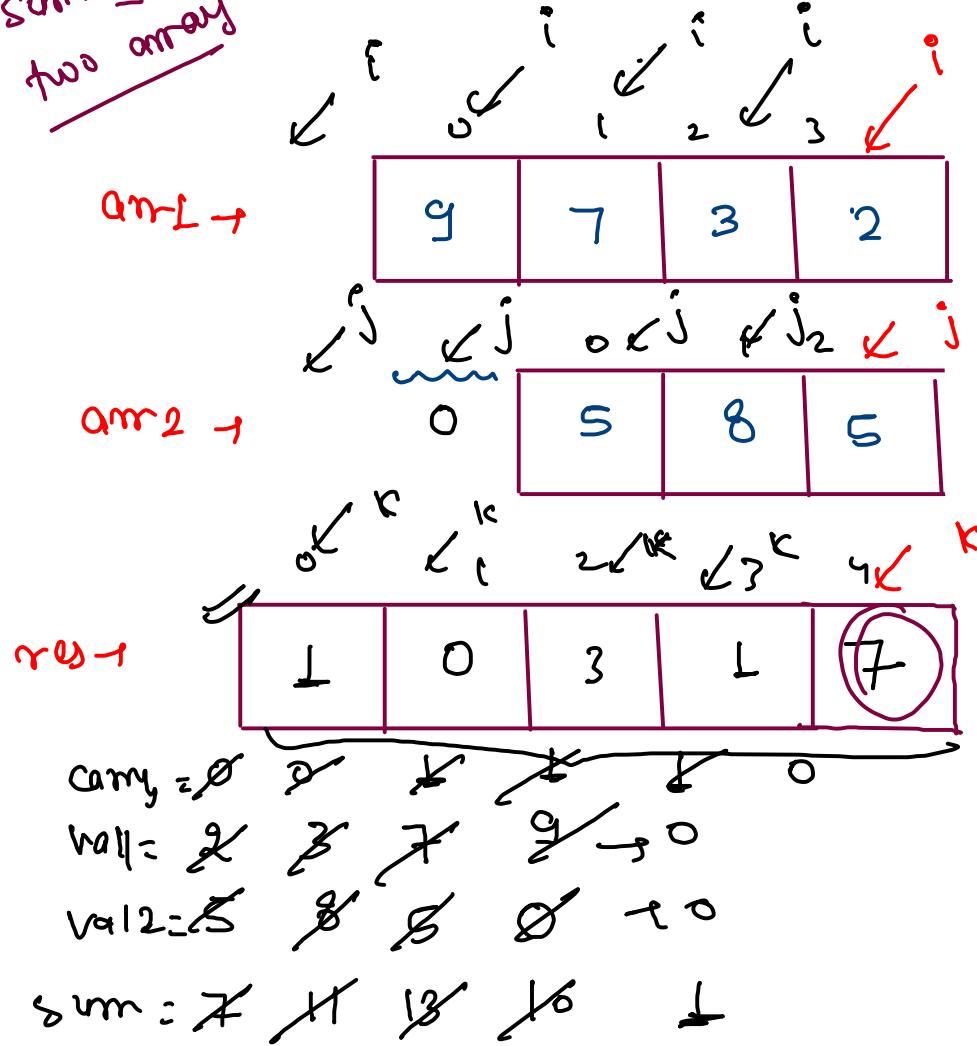
$$e = ?$$

$$\boxed{e = -\infty}$$

$$e = \text{Integer.MIN_VALUE}$$



sum of
two arrays.



int carry = 0

int val1 = arr1[i]

int val2 = arr2[j]

int sum = val1 + val2 + carry

res[k] = sum % 10,

carry = sum / 10.

i--;

j--;

k--;

i = 8 2 1 0

j = 2 1 0 -1

k = 4 3 2 1

ternary operator →

int val = condition ? Statement ① : Statement ②

if (condition == true) {

Statement ①

} else {

Statement ②

}

int val = i >= 0 ? arr[i] : 0;

False →

True →

```

public static void printSumOfArray(int[] arr1, int[] arr2) {
    int s1 = arr1.length;
    int s2 = arr2.length;

    int rsize = s1 > s2 ? s1 + 1 : s2 + 1;
    int[] res = new int[rsize];

    int i = s1 - 1;
    int j = s2 - 1;
    int carry = 0;
    for(int k = rsize - 1; k >= 0; k--) {
        int val1 = i >= 0 ? arr1[i] : 0;
        int val2 = j >= 0 ? arr2[j] : 0;

        int sum = val1 + val2 + carry;
        res[k] = sum % 10;
        carry = sum / 10;
        i--;
        j--;
    }

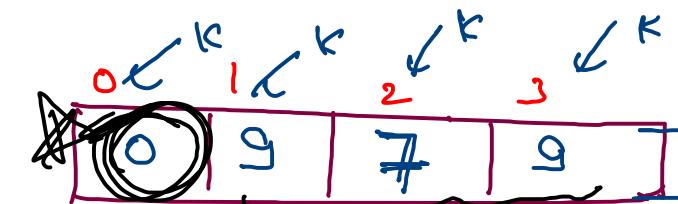
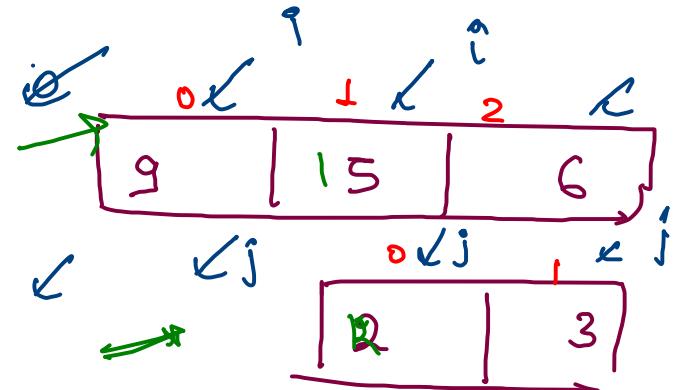
    // print array
}

```

$s1 = 3$

$s2 = 2$

$rsize = 4$



Carry = 1
val1 = 9
val2 = 7

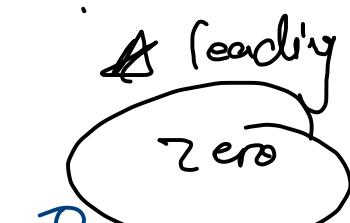
Sum = 16

Sum = 6

Sum = 0

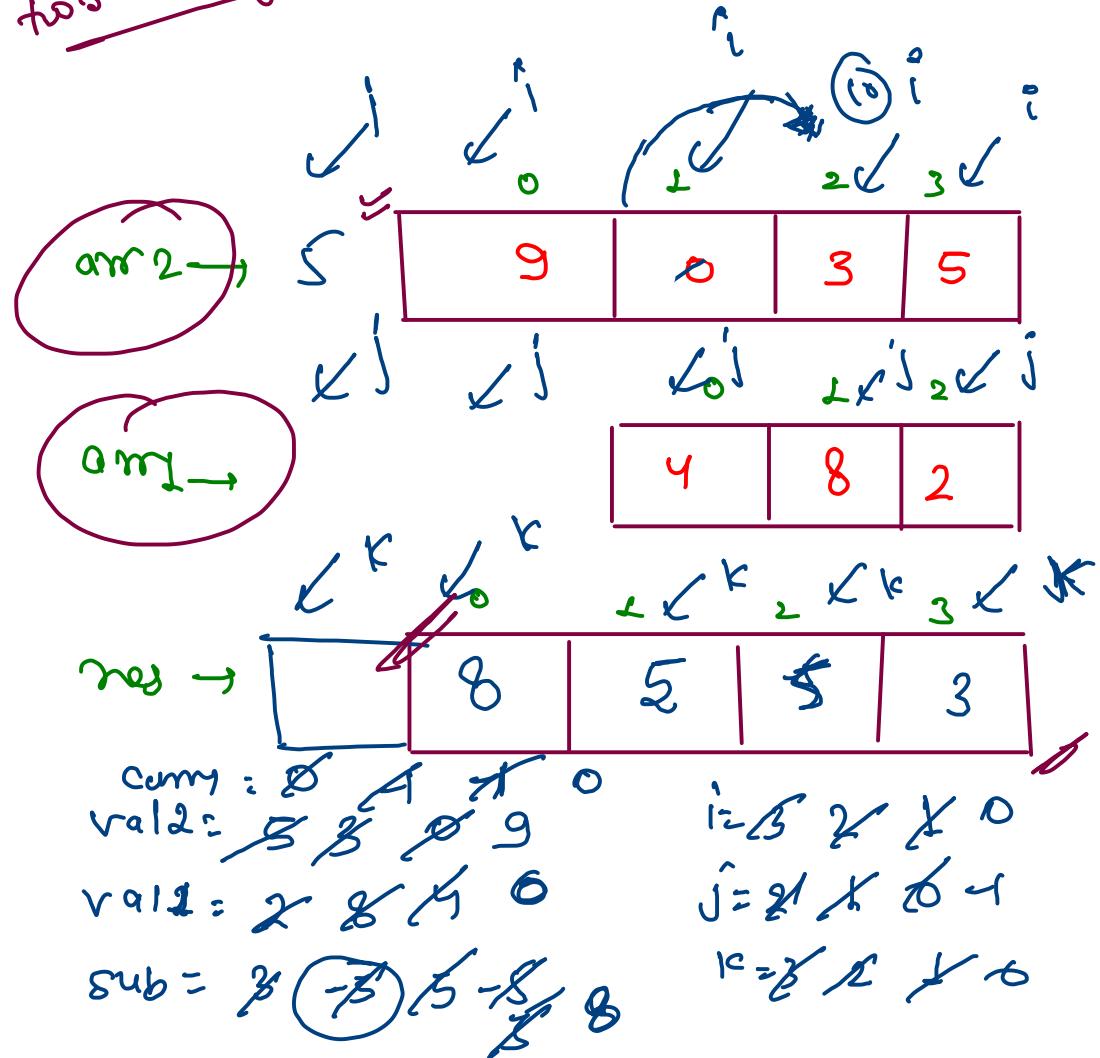
j = 1

k = 3



9
7
9

diff of two arrays



borrow

carry = 0

val2 = arr2[i]

val1 = arr1[j]

Sub = (val2 + carry) - val1

if (sub < 0) {

 sub += 10;

 carry = -1;

} else {

 Reset carry = 0;

 neg[k] = sub;

i--;

j--;

k--;

10

```
int carry = 0;
for(int k = s2 - 1; k >= 0; k--) {
    int val2 = i >= 0 ? arr2[i] : 0;
    int val1 = j >= 0 ? arr1[j] : 0;

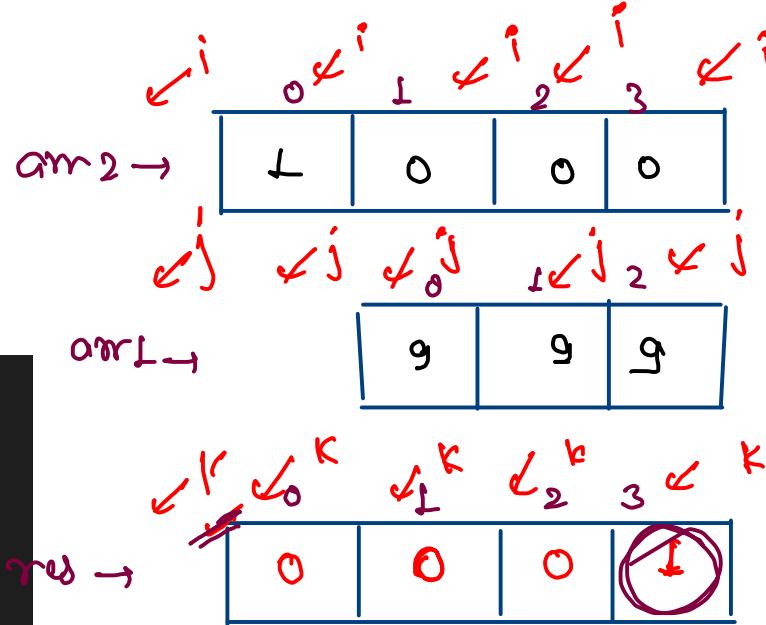
    int sub = (val2 + carry) - val1;

    if(sub < 0) {
        sub += 10;
        carry = -1;
    } else {
        // reset the carry
        carry = 0;
    }

    res[k] = sub;

    i--;
    j--;
}

// print res
```



To print

carry = 0 ✗ ✗ ↗ 1

val2 = 8 ✗ ↗ 0 0

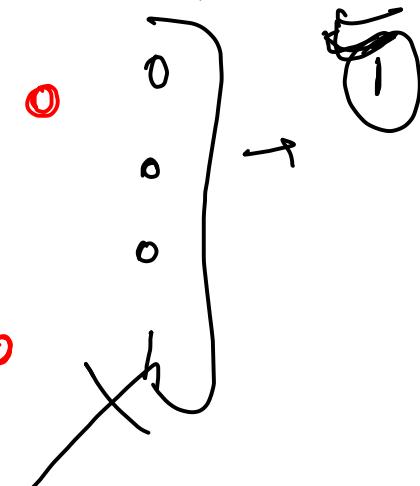
val1 = 8 ✗ 9 0

sub = 9 ✗ to 0 ✗ 0 0 0

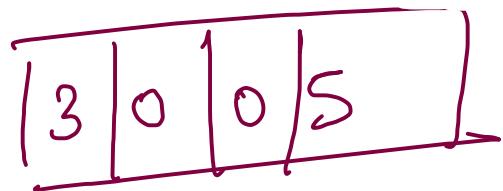
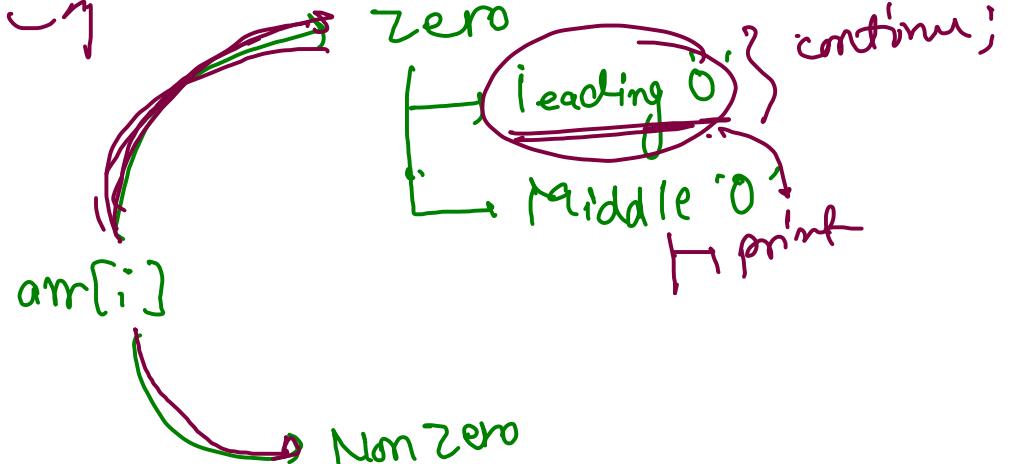
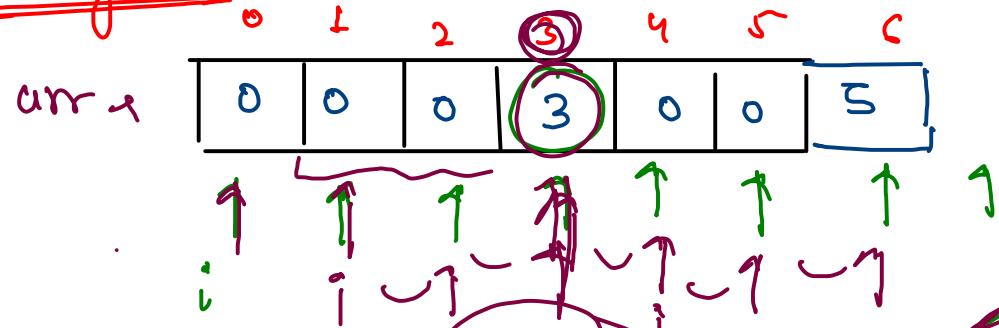
i = 3 2 ✗ 1 ✗ -1

j = 2 1 ✗ 0 ✗ -2

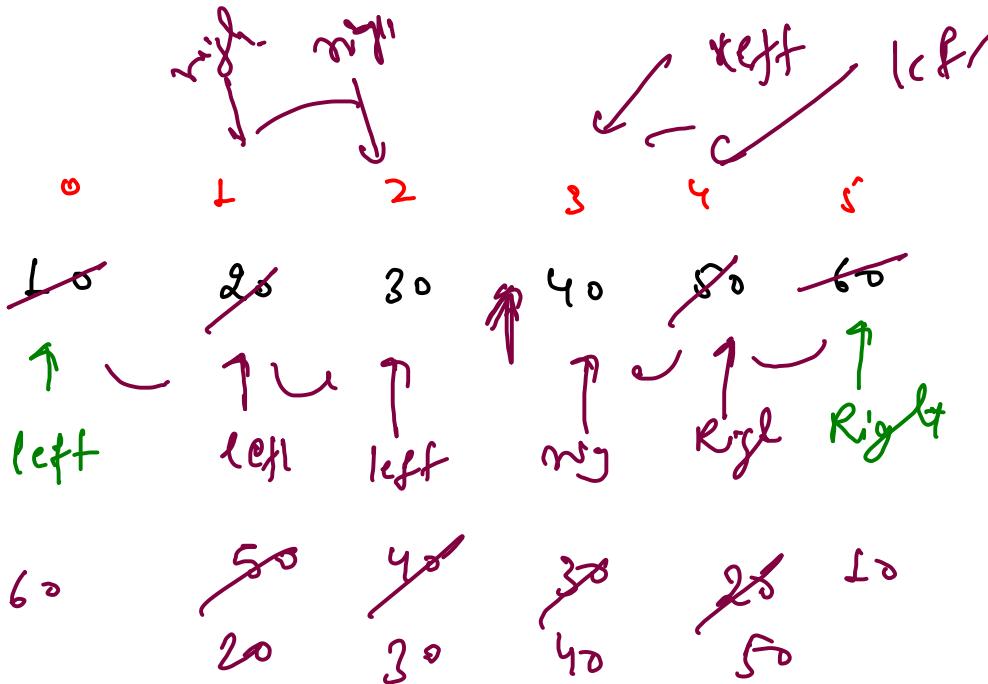
k = 3 2 ✗ 1 ✗ 0 -1



leading zero



Reverse an array -



while (left < right)

Swap,

left++;

right--;