1 1 1 0

$n_1 = 0\ 5\ 4\ 5$

$n_2 = 0\ 9\ 8\ 6$

1 5 3

Base = 10 → Decimal

$(3 \times 10) + 1$

n_1	n_2	d_1	d_2	Carry	Sum	digit
545	986	5	6	0	11	1
54	98	4	8	1	13	3
5	9	5	9	1	15	5 ✓
0	0	0	0	1	1	1
0	0			0		

$pow = 1 \times 10 = 10 \times 10 = 100 \times 10 = 1000 \times 10 = 10000$

$10^0 \quad 10^1 \quad 10^2 \quad 10^3$

$rv = 0 + (1) + (30) + (500) + (1000) \Rightarrow 1531$

```java
public static int getSum(int b, int n1, int n2){
    int rv = 0 , pow = 1 , carry = 0;

    while(n1 > 0 || n2 > 0 || carry > 0){
        int d1 = n1 % 10;
        int d2 = n2 % 10;

        int sum = d1 + d2 + carry;
        int digit = sum % base;
        carry = sum / base;

        rv = (digit*pow)+rv;

        n1 = n1 / 10;
        n2 = n2 / 10;
        pow = pow * 10;
    }

    return rv;
}
```

Base 10

n_2 : 1 0 0 2 4

n_1 : 0 5 6 7 8

---

0 4 3 4 6

$d_2 = 1$

$d_1 = 0$

borrow = 0

Diff = $d_2 - d_1$ − borrow

= 1 − 0 − 1

= 0

```
if (Diff < 0) {
    Diff = Diff + base;
    borrow = 1;
}
```

```
if (Diff >= 0) {
    borrow = 0;
}
```

B = 10

n_2 = 1024

n_1 = 05698

04346

n_1	n_2	d_1	d_2	borrow	diff	pow
5698	1024	8	4	0	6	1 [10]
569	102	9	2	1	4	10
56	10	6	0	1	3	100
5	1	5	0	1	4	1000
0	1	0	1	1	0	10000
0	0			0		100000

```
public static int getDifference(int b, int n1, int n2){
    int rv = 0 , pow = 1 , borrow = 0;

    while(n2 > 0){
        int d1 = n1 % 10;
        int d2 = n2 % 10;

        int diff = d2 - d1 - borrow;
        if(diff < 0){
            diff = diff + b;
            borrow = 1;
        }else{
            borrow = 0;
        }

        rv = (diff * pow) + rv;

        n1 = n1 / 10;
        n2 = n2 / 10;
        pow = pow * 10;
    }

    return rv;
}
```

rv = 0 + (6) + (40) + (300) + (4000) + 0 ⇒ 4341

Arrays

8 = b

1 = n_1

100 = n_2

pow = 1

1 8 1

1 8 2

9

-1 -1

100

1

0 -1 -1

0 7 7

n_1	n_2	d_1	d_2	borrow	diff	pow
1	100	1	0	0	0-1-0 = -1+8 = 7	1
0	10	0	0	1	0-0-1 = -1+8 = 7	10
0	1	0	1	1	1-0-1 = 0	100
0	0			0		1000

rv = (7) + (70) + 0 ⇒ 77

```java
public static int getDifference(int b, int n1, int n2){
    int rv = 0 , pow = 1 , borrow = 0;

    while(n2 > 0){
        int d1 = n1 % 10;
        int d2 = n2 % 10;

        int diff = d2 - d1 - borrow;
        if(diff < 0){
            diff = diff + b;
            borrow = 1;
        }else{
            borrow = 0;
        }

        rv = (diff * pow) + rv;

        n1 = n1 / 10;
        n2 = n2 / 10;
        pow = pow * 10;
    }

    return rv;
}
```

Arrays → Continuous space allocation of some type

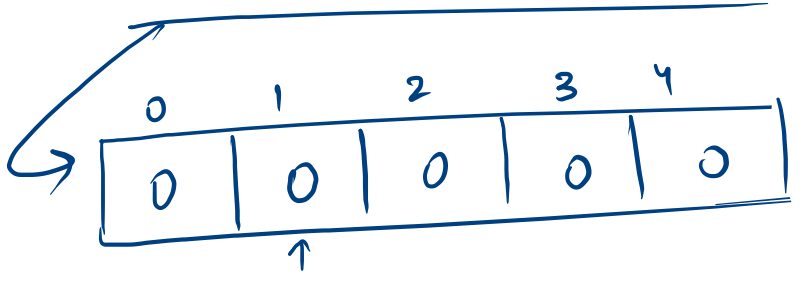
① Declaration :-

dataType arr[] = new datatype [length];

- int → 0
- String → null
- boolean → false
- ⋮

Example

int arr[] = new int[5]

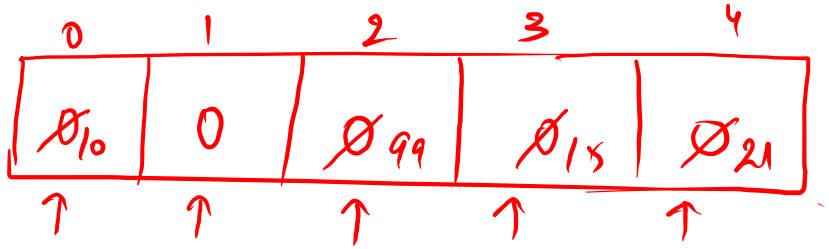


Index → [0 → length-1]

arr.length

```java
int n = 5;
int arr[] = new int[n];

arr[0] = 10;
arr[2] = 99;
arr[3] = 15;
arr[4] = 21;
// arr[idx] = scn.nextInt();
for(int idx = 0 ; idx < arr.length ; idx++){
    System.out.println(idx +" --> "+ arr[idx]);
}

for(int val : arr){
    System.out.println(val);
}

System.out.println(arr.length);
```

n=5



0 → 10

1 → 0

2 → 99

3 → 15

4 → 21

10

0

99

15

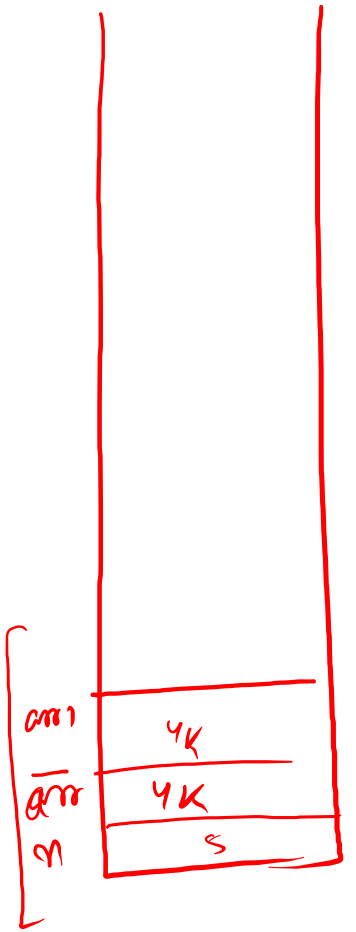
21

```java
int n = 5;
int arr[] = new int[n];

arr[0] = 10;
arr[2] = 99;
arr[3] = 15;
arr[4] = 21;
// arr[idx] = scn.nextInt();
for(int idx = 0 ; idx < arr.length ; idx++){
    System.out.println(idx +" --> "+ arr[idx]);
}

for(int val : arr){
    System.out.println(val);
}

System.out.println(arr.length);
```

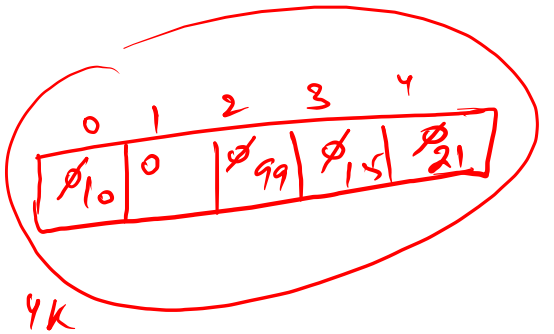
Heap

int arr1[]= arr;

Program stack



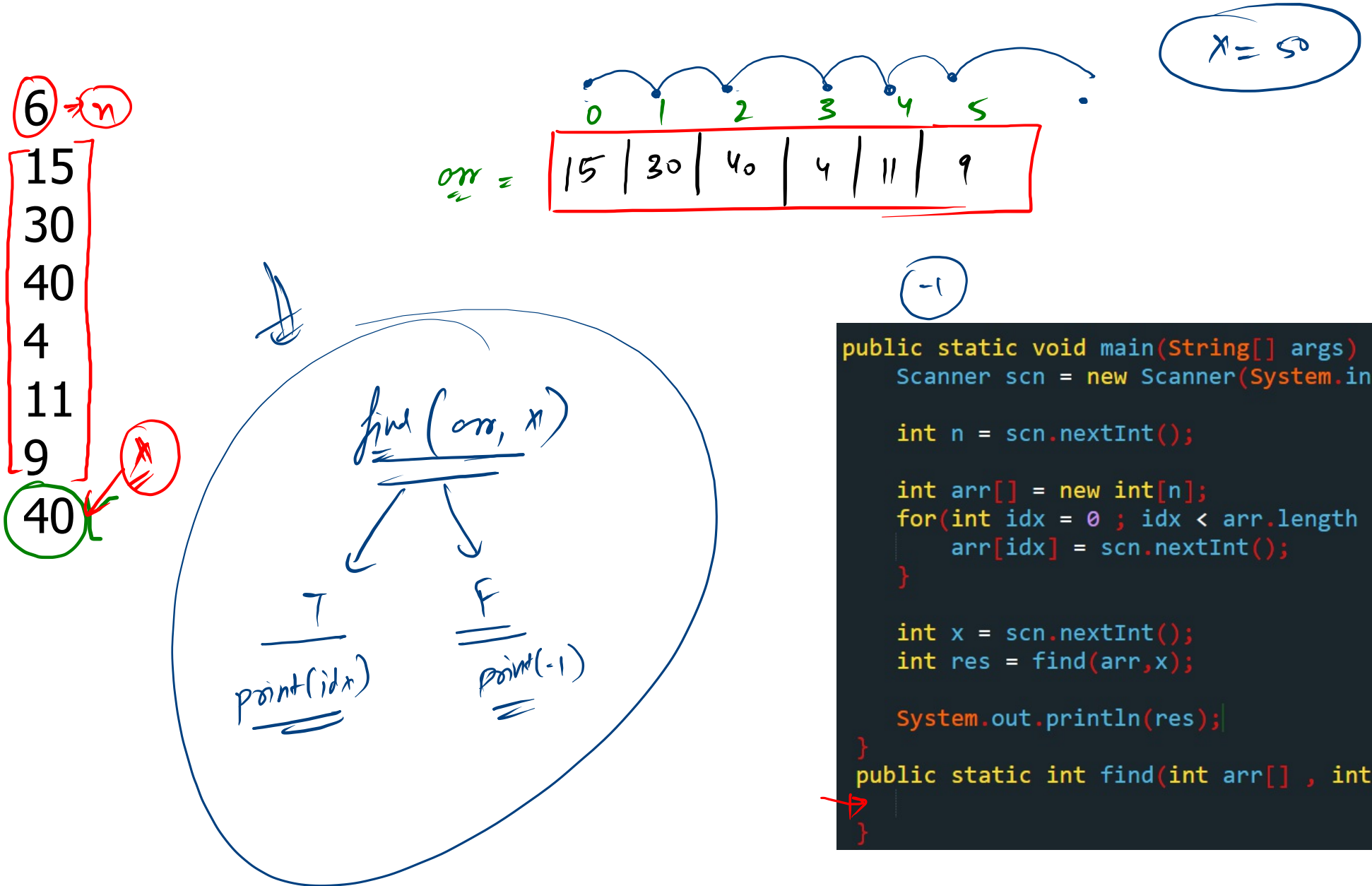
Heap



new

↳ Scanner

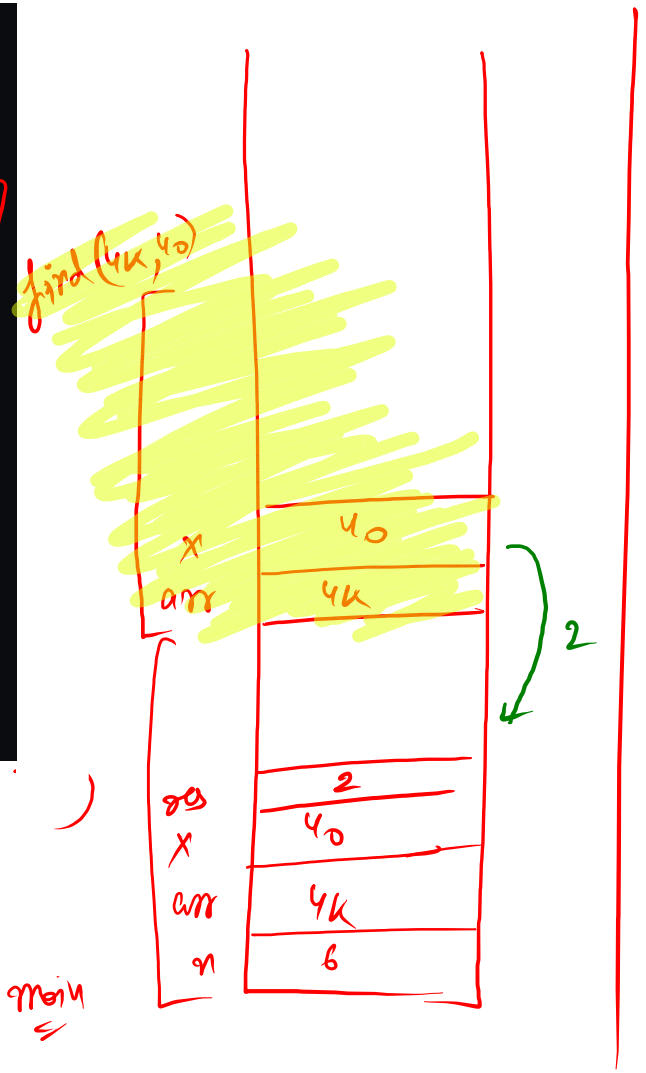
Memory Management

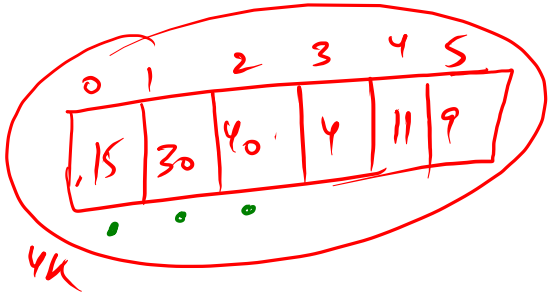



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);

    int n = scn.nextInt();

    int arr[] = new int[n];
    for(int idx = 0 ; idx < arr.length ; idx++){
        arr[idx] = scn.nextInt();
    }

    int x = scn.nextInt();
    int res = find(arr,x);

    System.out.println(res);
 }
 public static int find(int arr[] , int x){

 }
```

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);

    int n = scn.nextInt();

    int arr[] = new int[n];
    for(int idx = 0 ; idx < arr.length ; idx++){
        arr[idx] = scn.nextInt();
    }

    int x = scn.nextInt();
    int res = find(arr,x);

    System.out.println(res);
}
public static int find(int arr[] , int x){
    for(int idx = 0 ; idx < arr.length ; idx++){
        if(arr[idx] == x){
            return idx;
        }
    }

    return -1;
}
}
```

2



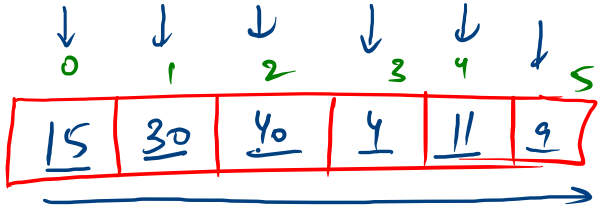


6
15
30
40
4
11
9
40

Span = Max value - Min value

Min = +∞ ~~15~~ 4

Max = -∞ ~~15~~ ~~30~~ 40



Span = Max - min

⇒ 40 - 4

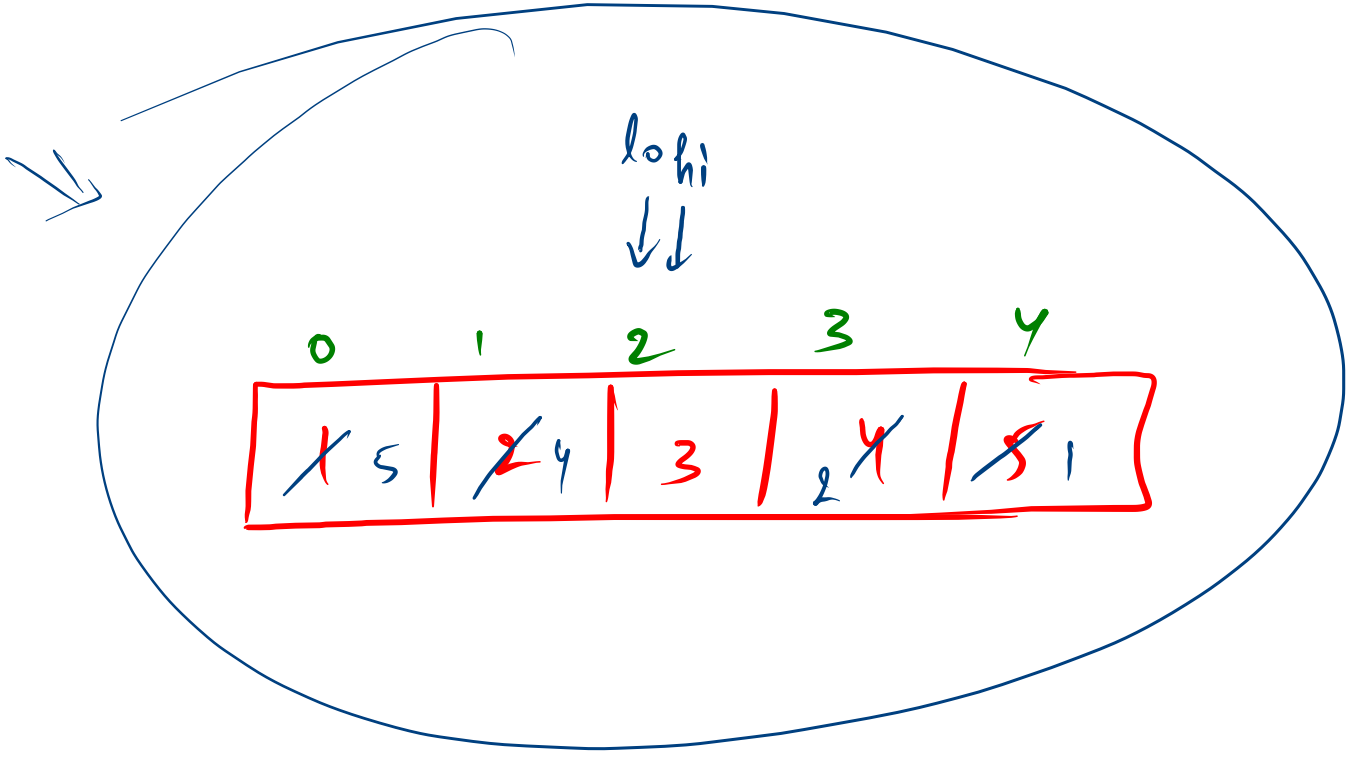
⇒ 36

[num + 0 = num]

[num * 1 = num]

Max(num, -∞) = num , -∞ = Integer.MIN_VALUE;

Min(num, +∞) = num , +∞ = Integer.Max_VALUE;


lo hi
0 1 2 3 4
5 4 3 4 1

```
5
1
2
3
4
5
```

```java
public static void reverse(int[] arr){
    int lo = 0;
    int hi = arr.length-1;

    while(lo < hi){
        int tmp = arr[lo];
        arr[lo] = arr[hi];
        arr[hi] = tmp;

        lo++;
        hi--;
    }
 }

public static void main(String[] args) throws Exception {
    BufferedReader br = new BufferedReader(new InputStreamReader(System
.in));

    int n = Integer.parseInt(br.readLine());
    int[] a = new int[n];
    for(int i = 0; i < n; i++){
       a[i] = Integer.parseInt(br.readLine());
    }

    reverse(a);
    display(a);
 }
```