Time Table { FJP - 2 }

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
9 PM class	9PM class	9PM class	off	9PM class	Weekly Test 9PM	Test + Doubts Discussion

Continuation of class

① Getting
② Patterns

Level ①

- Functions & Arrays
- 2D Arrays
- String & StringBuilder

DSA

- Recursion & Backtracking
- Searching & Sorting
- Dynamic Programming

My Introduction

→ Pointers

→ Pepcoding Articles (30 - 40%)

→ Job-Switch program - 4 { 3-4 months }

→ leetcode

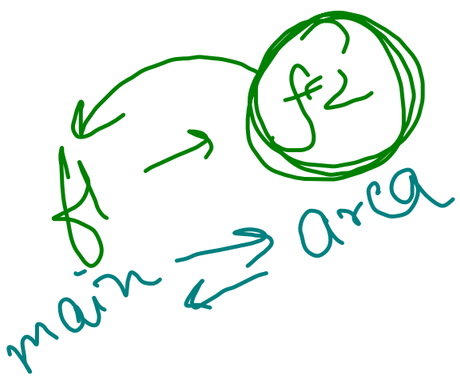
→ linkedIn

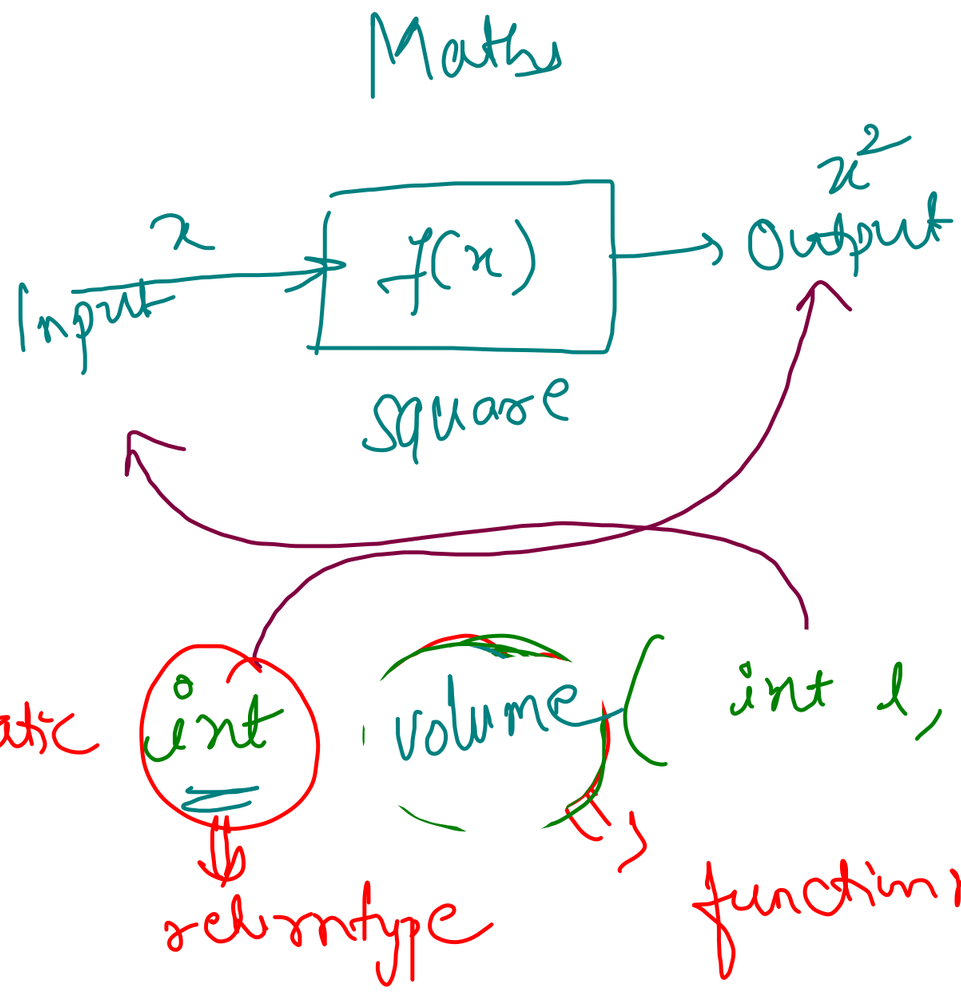
→ Competitive Programming

→ Product Based company

- Google (3)
- Salesforce (AMTS) (2)
- Sprinklr (1)

Functions

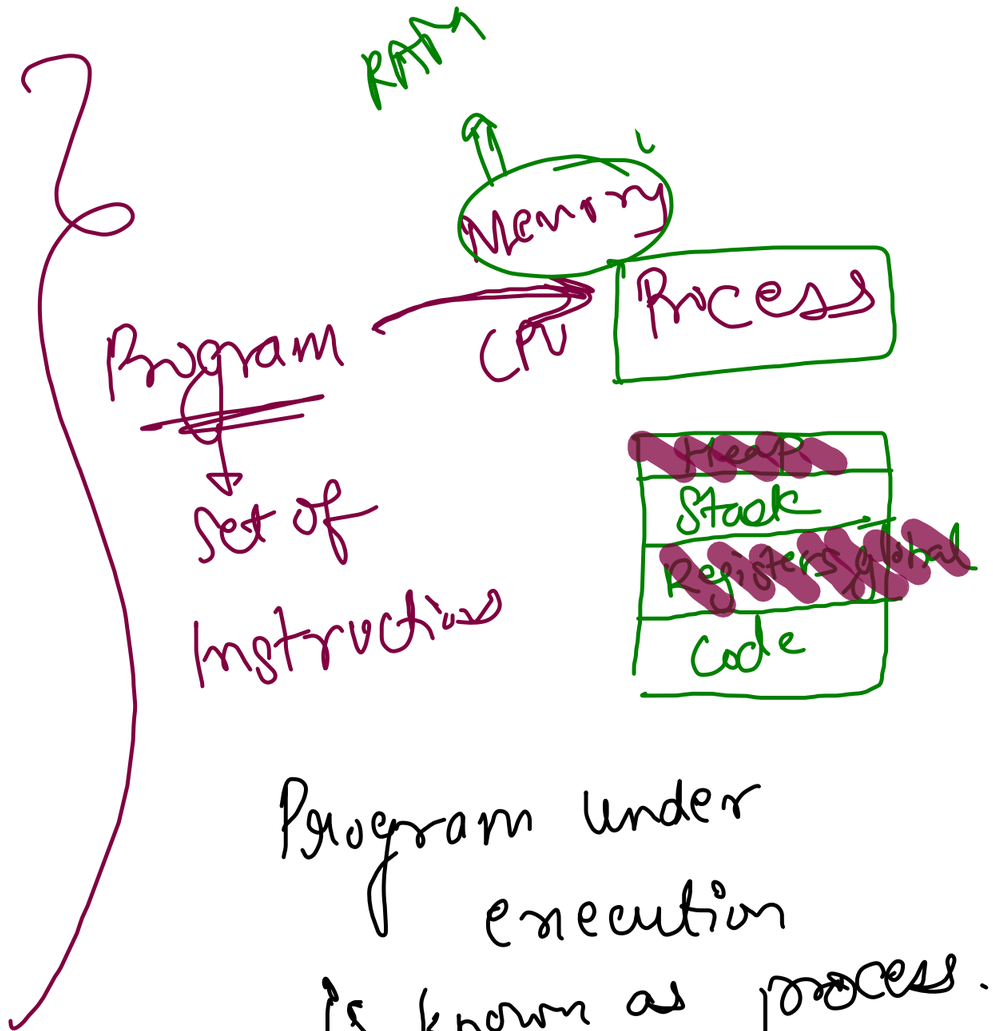





```
public static int volume( int l, int b, int h) {

}
```

returntype

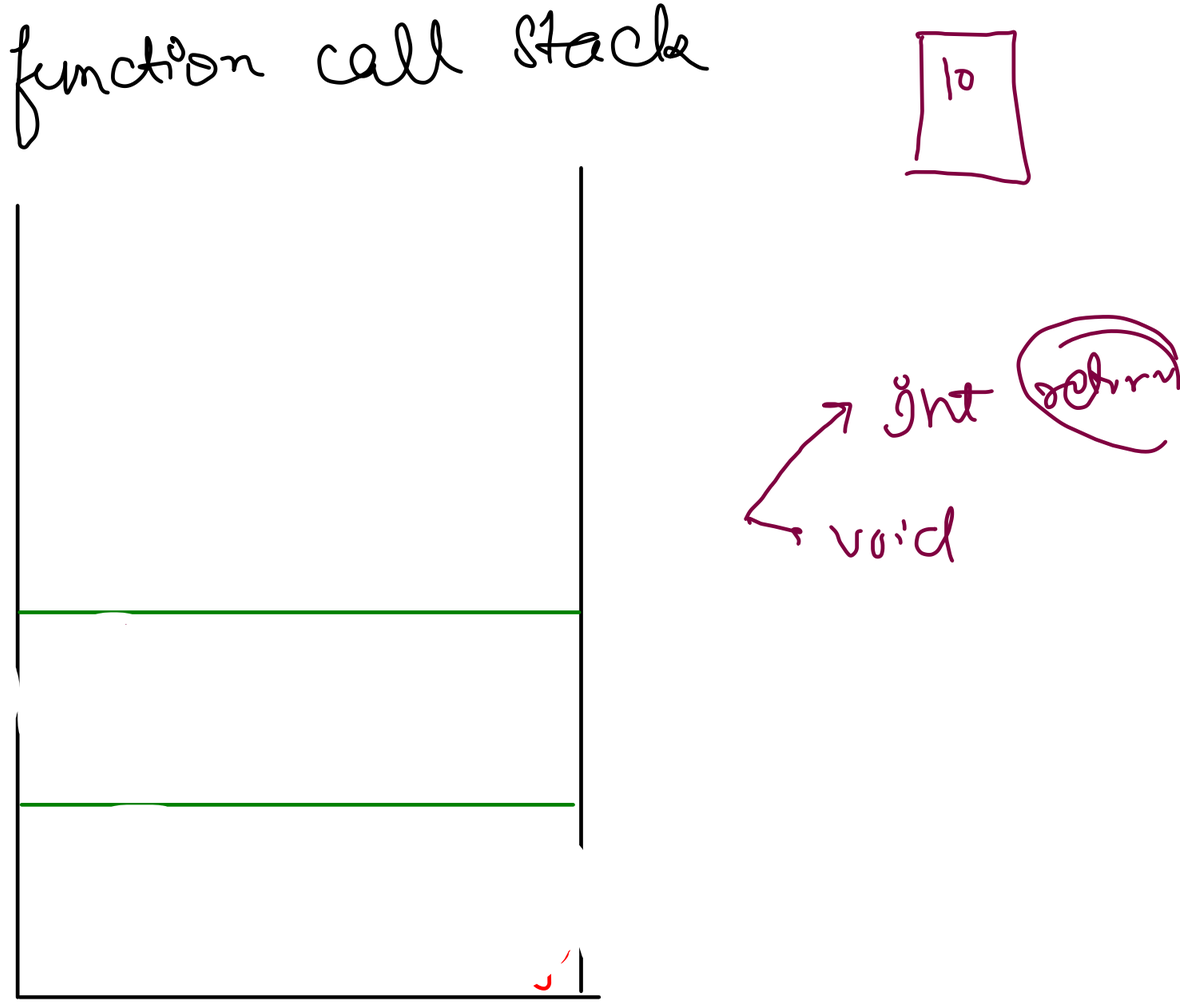
functionname



OS

Program under execution is known as process.

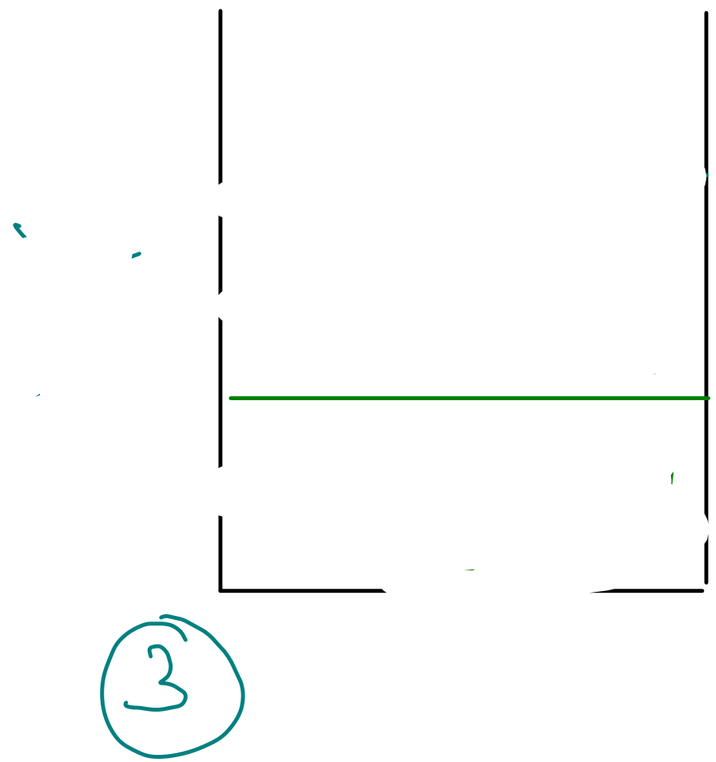
```java
public static int factorial(int n){
    int res = 1;

    for(int i=1; i<=n; i++){
        res = res * i;
    }

    return res;
}

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

    int n = scn.nextInt();
    int r = scn.nextInt();


    int nfact = factorial(n);
    int rfact = factorial(r);
    int nmrfact = factorial(n - r);


    int ncr = (nfact / (rfact * nmrfact));
    System.out.println(ncr);
}
```



n = 1292126429

d = 2

freq = 0 1 2 3



3

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int d = scn.nextInt();
    int f = getDigitFrequency(n, d);
    System.out.println(f);
}


public static int getDigitFrequency(int n, int d){

    int freq = 0;

    // For Extracting Digits
    while(n > 0){
        int digit = n % 10;

        if(digit == d){
            freq++;
        }

        n = n / 10;
    }

    return freq;
}
```

Number system

Decimal No

0-9

10 digits

$10 = 1*10^1 + 0*10$

100 → 99

$100 =$

25169_3

$= 6*10^2 + 9*10^1 + 3*10^0$

$+ 1*10^3 + 5*10^4$

$+ 2*10^5$

0 1 2 3 4 9 — 10 11 12 ---- 20 — 21 22 ---- 30 — 99 — 100

0-9 0-9 0-9 0-5

- min no 0000
- max no 9999
- count of nos 10^4

Binary No

2

0 1

00
01
10
11

0
1

000
001
010
011
1100
101
110
111

2^2 2^1 2^0

$1101011 = 1*2^0 + 1*2^1 + 0*2^2 + 1*2^3 + 0*2^4 + ---$

$2*2*2*2$

0/1 0/1 0/1 0/1

min 0000

max 1111

Count 2^4

Octal Nb System

→ 0, 1, 2, 3, 4, 5, 6, 7

$(7\ 5\ 3\ 4\ 6\ 1\ 2)_8$

$= 2*8^0 + 1*8^1 + 6*8^2$

$+ 4*8^3 + 3*8^4 + 5*8^5 + 7*8^6$

0	10	20	30	40 50 60
1	11	21	31	
2	12	:	:	47 57 67
3	13	:	:	
4	14	:		
5	15	27	37	
6	16			
7	17			

70 77

100 101 102 107 (777) 1000 7777 10000

$7*8^2 + 7*8^1 + 7*8^0$

$(777)_8 + 1$

1 0 0 0

$1*8^3$

$(999)_{10} = 1000$

$9*10^2 + 9*10^1 + 9*10^0$

$1*10^3$

6 months → leetcode (L2)

Octal

8 * 8 * 8 * 8

0/7 0/7 0/7 0/7

min ⇒ 0000

max ⇒ 7777

total ⇒ 8^4

pepcoding (L1)

L2 → gfg → a

service based   tier 3 <   tier 1 (3rd, 4th)

(2-3) hr   (8 hr)

DSA

GFG 200

2-3 Dev

DSA { 600 + leetcode }

Dev { 50 + Project }

150 → backtrack (X) → DSA

80 - 90

40 - 50

Today's Target

- Decimal To Any Base ⇒ classwork
- Any Base To Decimal ⇒ classwork
- Any Base To Any Base ⇒ homework
- Any Base Addition ⇒ classwork
- Any Base Subtraction ⇒ classwork
- Any Base Multiplication ⇒ homework

dividend

Decimal to Any Base

$(634)_{10}$ = $(?)_8$

$\uparrow 61$ $\uparrow 62$

→ What ?
→ how ?
why ?

b	N	rem	multiplier
8	634		$10^0 = 1$
8	79	2	10^1
8	9	7	10^2
8	1	1	10^3
	0		

$79 = 9 * 8 + 7$

$\{0+2\}+\{70\}+100+1000$

$= 1172$

$(1172)_8$

```
public static int getValueInBase(int n, int b){
    int res = 0;
    int multiplier = 1;

    while(n > 0){
        int divisor = n / b;
        int remainder = n % b;

        n = divisor;
        res = res + remainder * multiplier;
        multiplier *= 10;
    }

    return res;
}
```

$(634)_{10}$

$6*10^2 + 3*10^1 + 4*10^0$

$=$

$(1172)_8$

$1*8^3 + 1*8^2 + 7*8^1 + 2*8^0$

10:20

$(634)_{10} = 8^1 * (-) + 2$

$= 8^4*0 + 8^3*1 + 8^2*1 + 8^1*7 + 8^0*2$

$79 = 8^1 * (8^2*0 + 8^1*1 + 8^0*1) + 7$

$= (8^3*0 + 8^2*1 + 8^1*1 + 8^0*7)$

$634 = 8^1 * 79 + 2$

$79 = 8^1 * 9 + 7$

$9 = 8^1 * 1 + 1$

$1 = 8^1 * 0 + 1$

$9 = 8^1 * (8^1*0 + 1) + 1$

$9 = 8^2*0 + 8^1*1 + 8^0*1$

$(173)_{10} = (?)_8$

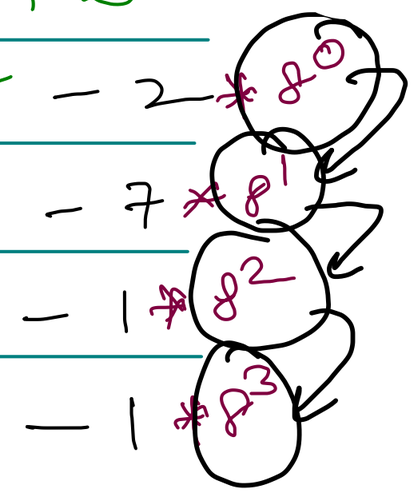
8	173
8	21 − 5 * 10^0
8	2 − 5 * 10^1
	0 − 2 * 10^2

$= 5 + 50 + 200 = (255)_8$

Any Base to Decimal

$(1172)_8 = (?)_{10}$

10	1172
10	117 — 2
10	11 — 7
10	1 — 1
	0 — 1



$= 2*8^0 + 7*8^1 + 1*8^2 + 1*8^3$

$= (634)_{10}$

```java
public static int getValueIndecimal(int n, int b){
    int res = 0;
    int multiplier = 1;

    while(n > 0){

        int divisor = n / 10;
        int remainder = n % 10;

        n = divisor;

        res = res + remainder * multiplier;
        multiplier *= b;
    }

    return res;
}
```

$(110010)_2 = (?)_{10}$

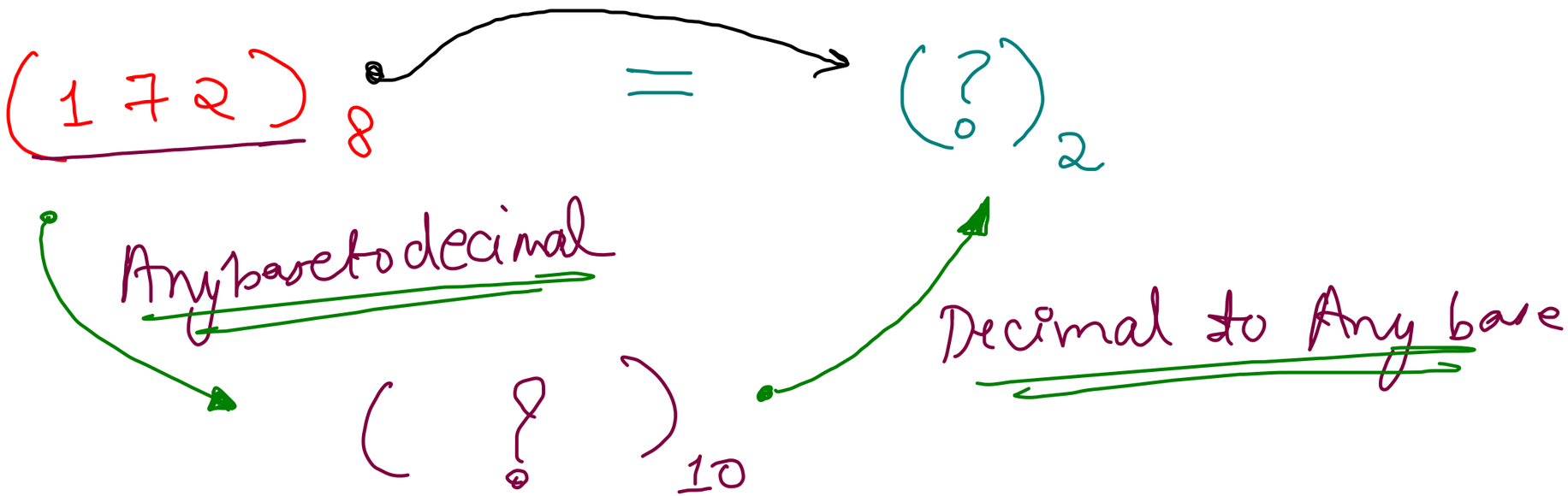
10	110010	
10	11001	$-0 * 2^0$
10	1100	$-1 * 2^1$
10	110	$-0 * 2^2$
10	11	$-0 * 2^3$
10	1	$-1 * 2^4$
10	0	$-1 * 2^5$

$= 0 + 2 + 0 + 0 + 16 + 32$

$= 18 + 32$

$= (50)_{10}$

Any base to any base



Any Base addition

carry 1 0 1 0

$(4\ 3\ 6)_{10}$

$+\ (7\ 5\ 4)_{10}$

$(1\ 1\ 9\ 0)$

temp = (0 + 6 + 4)

$\frac{10}{10}$ → 1 {quot}, 0 {rem}

1 + 3 + 5 = $\frac{9}{10}$ → 0, 9

0 + 4 + 7 = $\frac{11}{10}$ → 1, 1

Carry 1 1 1 0

0 (2 3 6)8

0 (7 3 4)8

---

1 2 1 2

$\{2*10^0 + 1*(0^1 + 2*10^2 + 1*10\}$

1212 = 1000 + 200 + 10 + 2

236 → 23

236/10 = 23

① st

c d1 d2

6 + 6 + 4

= 16 → 1 (quot), 6 (rem)

8

② nd

1 + 3 + 5

= 9 → 1, 1

8

③ rd

1 + 2 + 7 = 10 → 1, 2

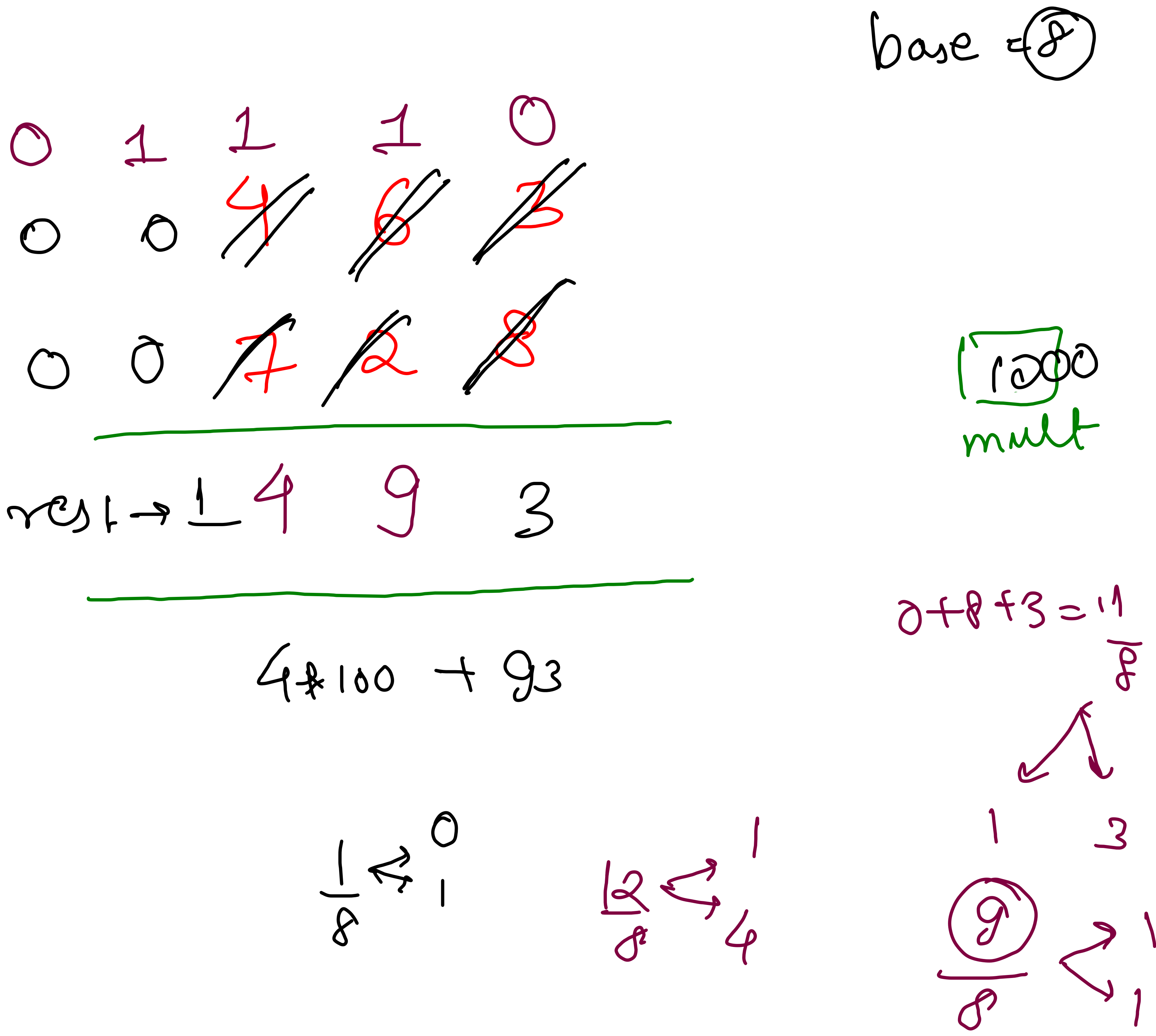
8

④ th

1 + 0 + 0 = 1 → 0, 1

8

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

    while(carry > 0 || n1 > 0 || n2 > 0){

        int d1 = n1 % 10;
        int d2 = n2 % 10;

        int temp = carry + d1 + d2;
        int quot = temp / b;
        int rem = temp % b;

        res = res + (rem * multiplier);
        carry = quot;

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

    return res;
}
```

Pseudo Code

```
carry = 0
res = 0
multiplier = 1

while(carry || n1 > 0 || n2 > 0) {
    d1 = n1 % 10
    d2 = n2 % 10
    temp = carry + d1 + d2
    quot = temp / base
    rem = temp % base

    carry = quot
    res += rem * multiplier

    multiplier *= 10
    n1 /= 10
    n2 /= 10
}
```

quotient will become next carry

&

rem will be Contributing in result.

Any base substraction

borrow

$$\begin{array}{cccc} -1 & -1 & -1 & \\ (1 & 2 & 1 & 2)_{10} \\ - & (2 & 5 & 6)_{10} \\ \hline 0 & 9 & 5 & 6 \end{array}$$

$d1 - d2 + \text{borrow}$

$2 - 6 + 0 = -4$

$2-6 = -4+10$

$= 6$

gfl

$2-2+(-1)+10$

$= -1+10 = 9$

$1-5+(-1)$

$= -4-1 = -5+10$

$= 5$

Any base subtraction

borrow

$$\begin{array}{cccc} -1 & -1 & -1 & 0 \\ (1 & 2 & 1 & 2)_{10} \\ - & (2 & 5 & 6)_{10} \\ \hline 0 & 9 & 5 & 6 \end{array}$$

$-4+0$

$2-6 = -4+10$

$= 6$

Q − d1 + borrow

$2 - 6 + 0 = -4 + 10$

$= 6$

gf1

$2-2+(-1)+10$

$= -1+10 = 9$

$1 - 5 + (-1)$

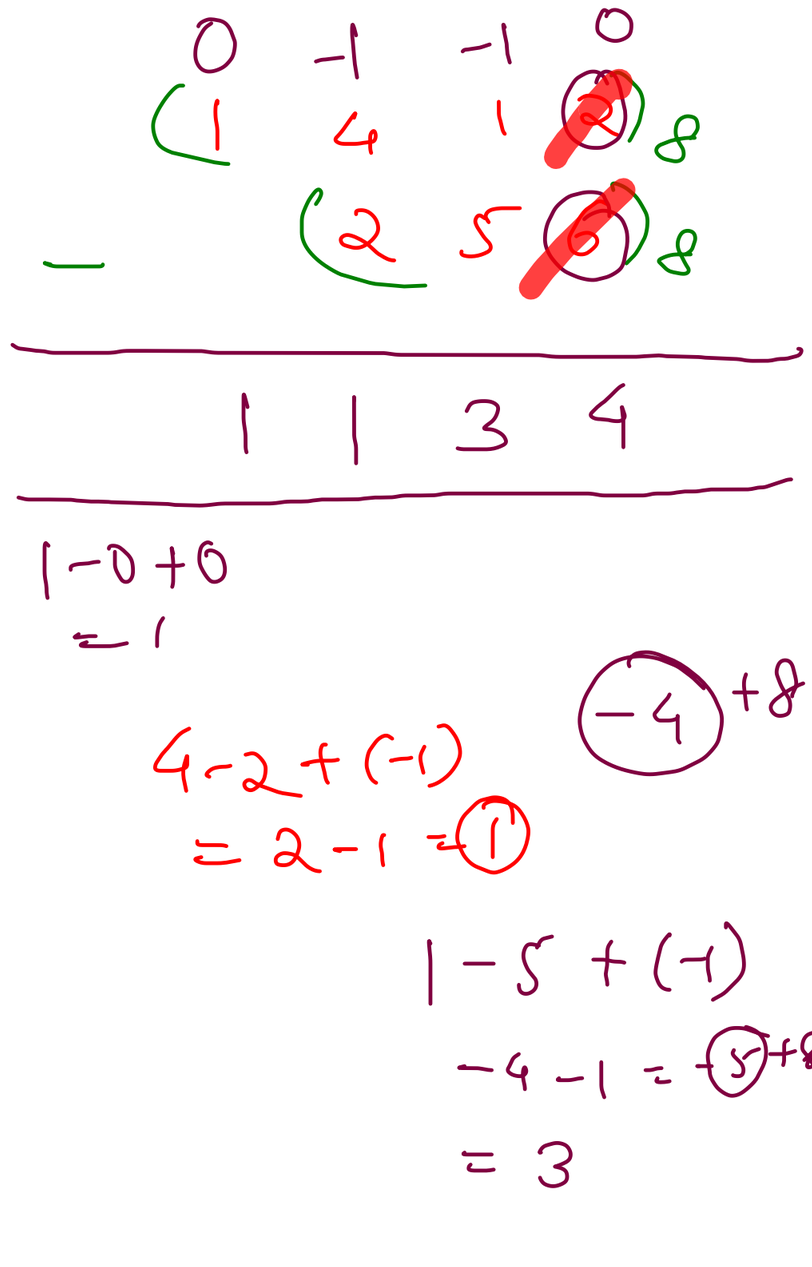
$= -4 - 1 = -5 + 10$

$= 5$

Pseudo code

$n2 > n1$

```
borrow = 0      res = 0      multiplier = 1

while( n2 > 0) {

    d1 = n1%10 , d2 = n2%10

    temp = d2 - d1 + borrow

    if(temp < 0) {
        res += (temp + base) * multiplier;
        borrow = -1;
    } else {
        res += temp * multiplier;
        borrow = 0;
    }

    multiplier *= 10;
    n1 /= 10;   n2 /= 10;
}
```



Any base Multiplication

$$(2\ 1\ 5\ 6)_{10} \times (7\ 4)_{10}$$

1	1	3	4	0	
	(2	1	5	6)$_{10}$	
X			(7	4)$_{10}$	
21564	{ 1	8	6	2	4	7
215620	1	5	0	9	2	X
	1	5	9	5	4	4

Any base addition

$7*2+1 = 15_{10}$ → 1, 5

$7*6 = 42_{10}$ → 4, 2

$\frac{24}{10}$ → 2, 4

$\frac{39}{10}$ → 3, 9

$5*4+2 = \frac{22}{10}$ → 2, 2

$4*1+2 = \frac{6}{10}$ → 0, 6

$\frac{8}{10}$ → 0, 8

res

Algorithm

while{ n1->0 }{

① Extract digits of n1 : d1

② multiply d1 with n2 : temp → getProductwithDigit( )

③ res = anybaseAddition(res, temp, multiplier)

multiplier *= 10

}

pencil eraser

$$
\begin{array}{cccccc}
 & 1 & 1 & 5 & 5 & 0 \\
 & (2 & 1 & 5 & 6)_8 & \\
\times & & & (7 & 4)_8 & \\
\hline
1 & (1 & 0 & 6 & 7 & 0)_8 \\
(1 & 7 & 4 & 0 & 2 & 0)_8 \\
\hline
2 & 0 & 4 & 7 & 1 & 0
\end{array}
$$

2156 * 74

= 2156 * 4 (+) 2156 * 7 * 8

octal

$\frac{24}{8}$ → 3, 0

20 + 3 = $\frac{23}{8}$ → 2, 7

$\frac{42}{8}$ → 5, 2

$\frac{6}{8}$ → 0, 6

$\frac{40}{8}$ → 5, 0

$\frac{8}{8}$ → 1, 0

$\frac{12}{8}$ → 1, 4

$\frac{1}{8}$ → 0, 1

$\frac{15}{8}$ → 1, 7

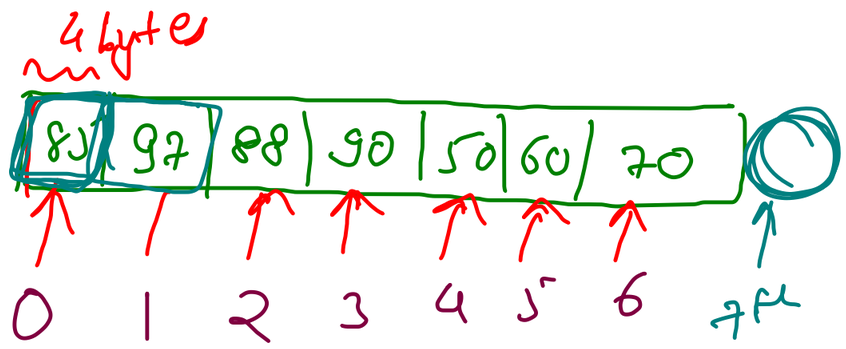
Data structure

Arrays

int m1 = 95;

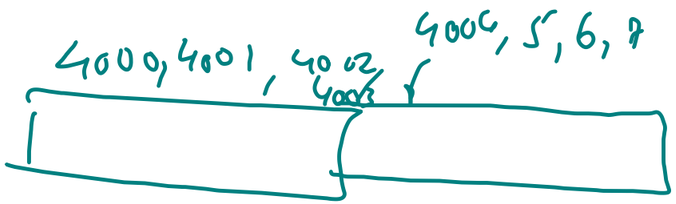
int m2 = 97;

int m3 = 88;



contiguous memory allocation

7 * 4 = 28 byte



7 size

0 - 6

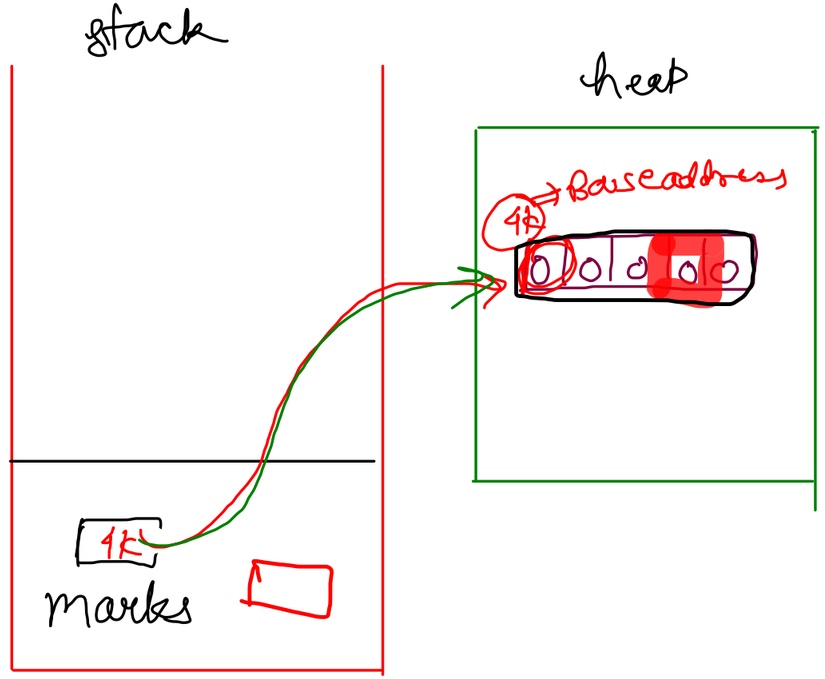
Memory mapping of arrays

```
int[] marks = new int[n];
```

{Static array}

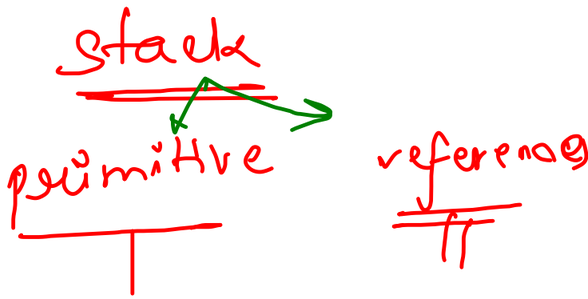
marks [i]

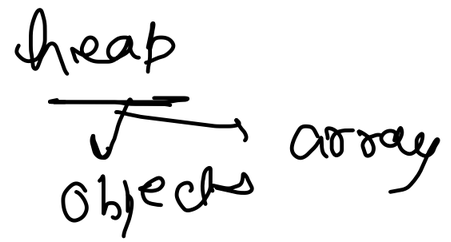
base address of array + ith block * 4 bytes



Random access {constant operation}

:=> BA + index * size of one block





marks [0]

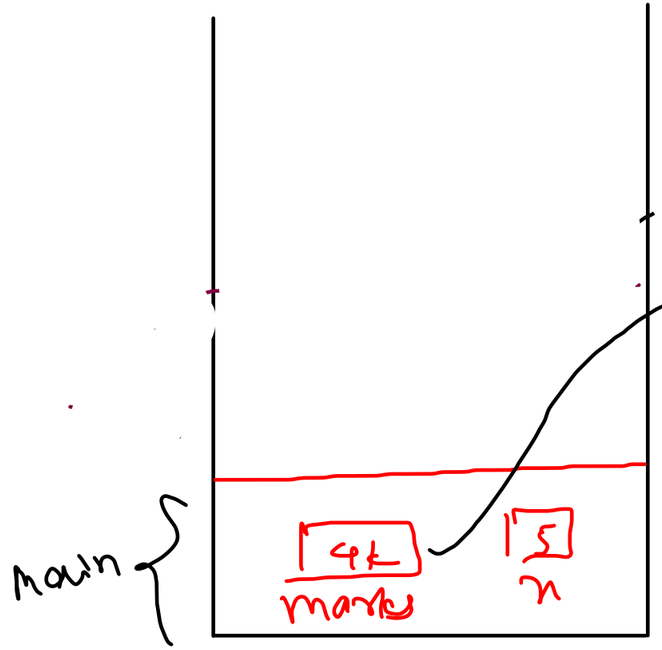
4k + 0*4 = 4k

marks[3]

4K + 3*4 = 4k12

Passing array to function

```java
public static void display(int[] marks){
    for(int i=0; i<marks.length; i++){
        System.out.print(marks[i] + " ");
    }

}

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

    int n = scn.nextInt();

    // Declaration & Initialization
    int[] marks = new int[n];

    // Random Access using Indexing
    for(int i=0; i<n; i++){
        marks[i] = scn.nextInt();
    }

    // Array Traversal
    display(marks);
}
```

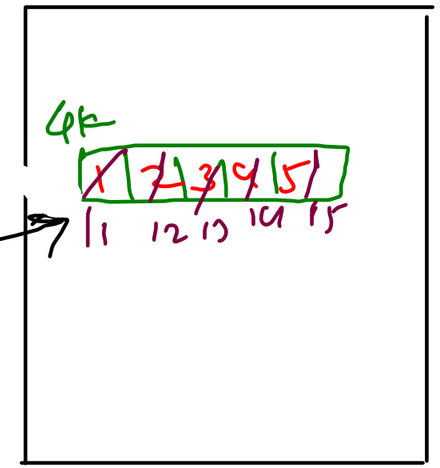


"Changes in the heap always persist"

fcnt call stack

Shallow copy

permanent



"Java is always pass by value"

Shallow copy

Java

8 primitive

4byte int

4byte float

8byte double

1byte byte

1byte boolean

2byte character

8byte short

8byte long

(-∞) Integer.MIN_VALUE

int

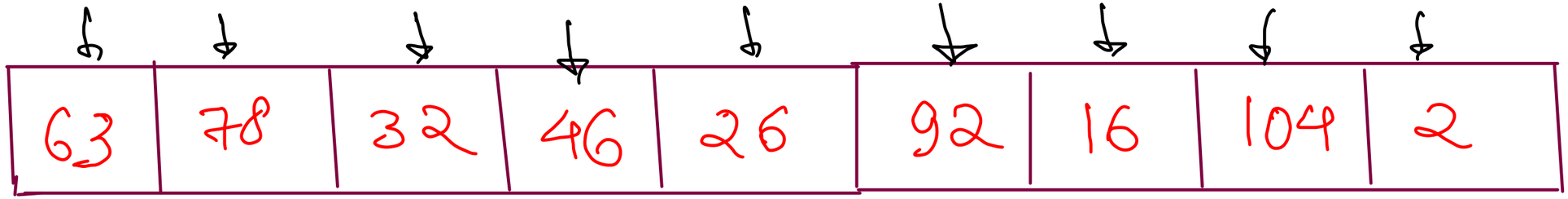
-2^{31} to $+2^{31}-1$

Integer.MAX_VALUE (+∞)

$\approx 10^9$

long

-2^{63} to $2^{63}-1$



$span = max^m - min^m$

$max^m =$ ~~$-\infty$~~ ~~63~~ ~~78~~ ~~92~~ 104

$min^m =$ ~~$+\infty$~~ ~~63~~ ~~32~~ ~~26~~ ~~16~~ 2

```java
int min = Integer.MAX_VALUE;
int max = Integer.MIN_VALUE;

for(int i=0; i<n; i++){

    // is current element less than min of previous
    if(arr[i] < min){
        min = arr[i];
    }

    // is current ele greater than max
    if(arr[i] > max){
        max = arr[i];
    }
}

int span = max - min;
System.out.println(span);
```

identities

$x + 0 = x$

$x * 1 = x$

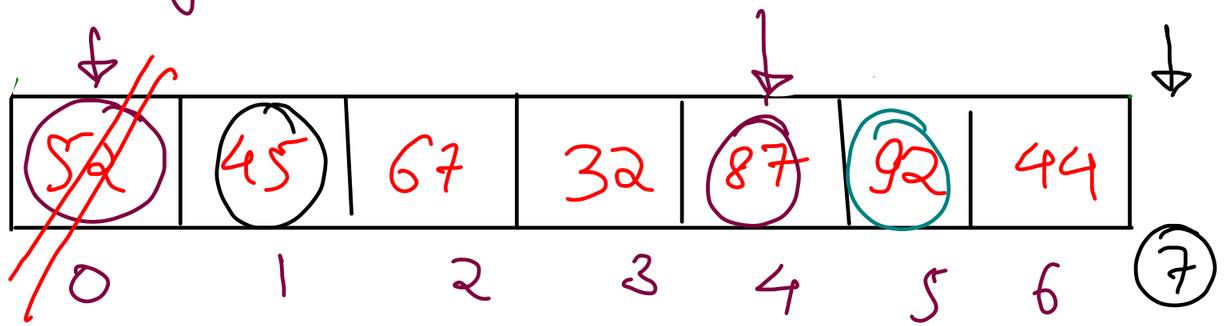
$max(x, -\infty) = x$

$min(x, +\infty) = x$

Today's questions

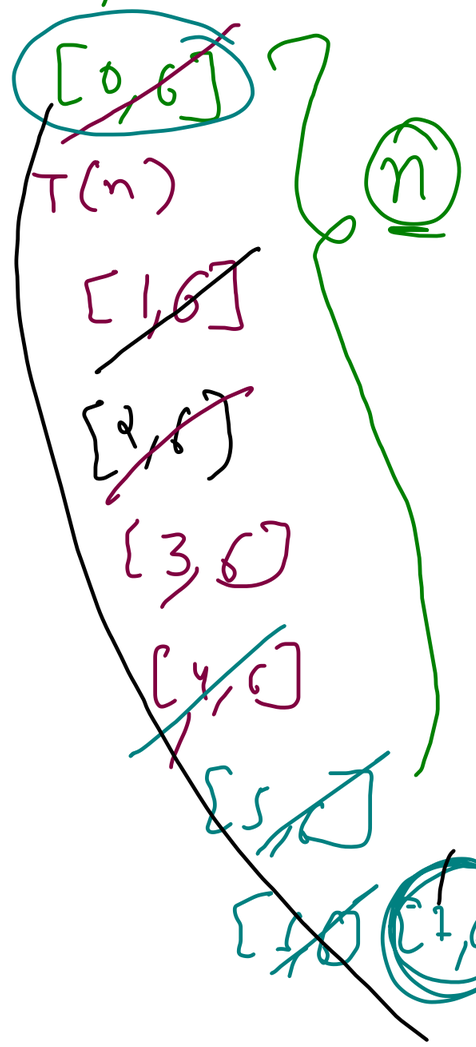
- </> Find Element In An Array → Linear Search Algorithm
- </> Bar Chart
- </> Sum Of Two Arrays
- </> Difference Of Two Arrays
- </> Reverse An Array
- </> Rotate An Array

Linear Search Algorithm



ele = 87 92 5?

search space



Execution Time / Run Time

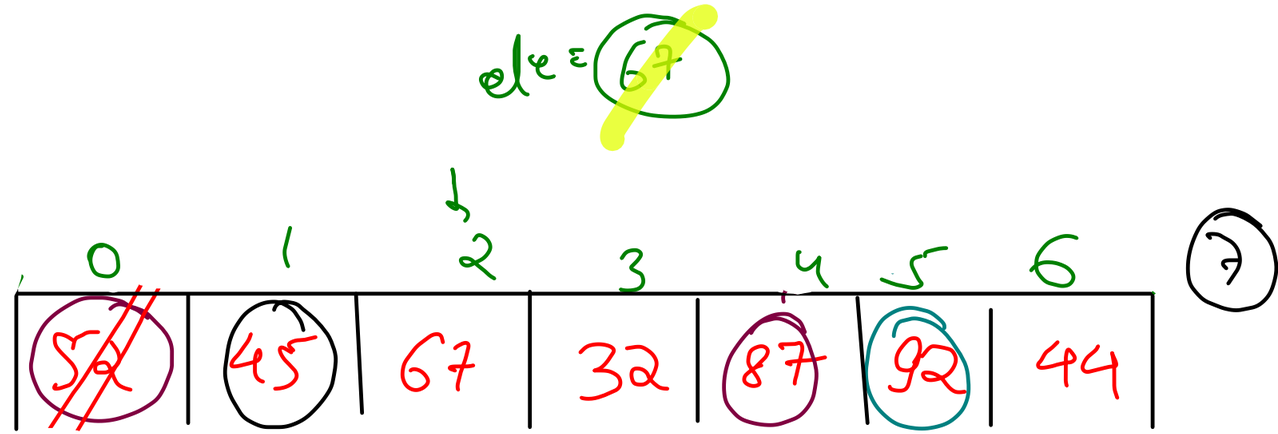
- 1 comp ⇒ best case
- n comp ⇒ worst case
- avg ; $\frac{n}{2}$ ⇒ avg case

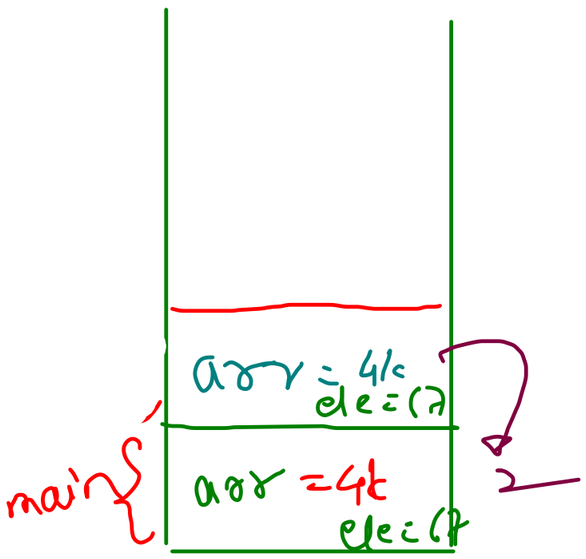
```
for(idx=0; idx < arr.length; idx++)
    if(arr[idx] == ele){
        return idx;
    }
}
return -1;
```

successful

unsuccessful search

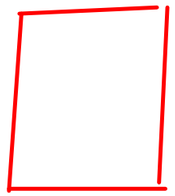
```java
public static int linearSearch(int[] arr, int ele) {

  for (int idx = 0; idx < arr.length; idx++) {

    if (arr[idx] == ele) {
      // successful search
      return idx;
    }

  }

  return -1; // unsuccessful search (element not found)
}
```





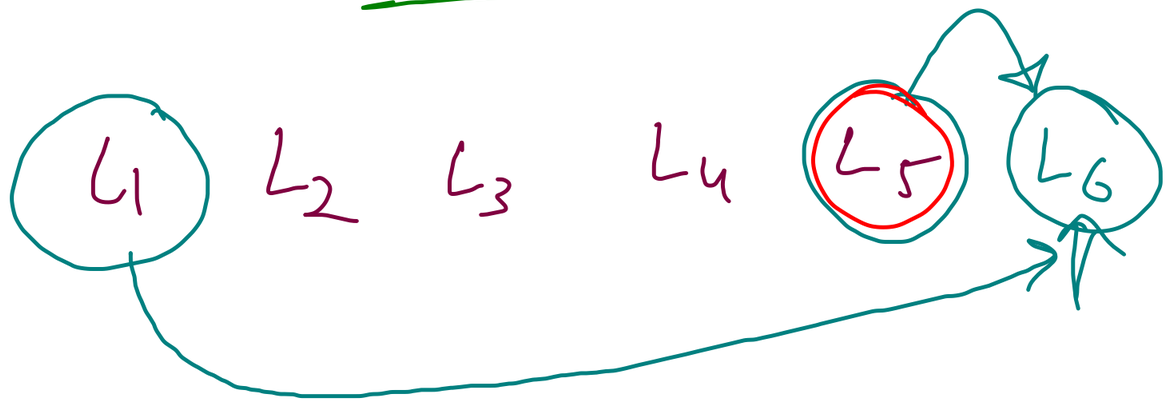

```java
public static void main(String[] args) throws Exception {
  Scanner scn = new Scanner(System.in);
  int n = scn.nextInt();

  int[] arr = new int[n];

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

  int ele = scn.nextInt();

  int ans = linearSearch(arr, ele);
  System.out.println(ans);
}
```

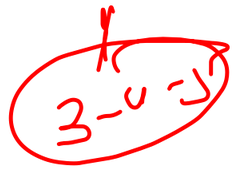
Notes

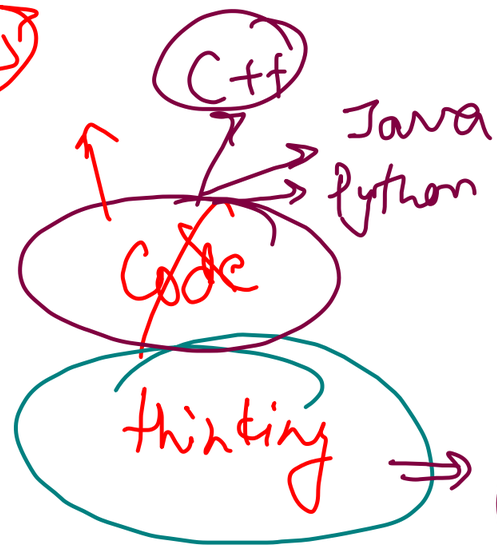


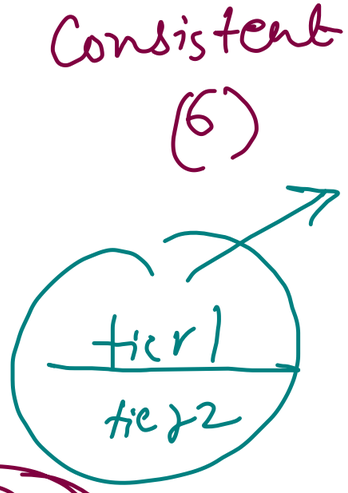
Question name
1 liner logic
Important points

Revision









DSA + Dev + Core

SB-C. (3-6)	Small Startups (6-12)	big startups (12-30)	PBC	tech giant {FAANG}
Aptitude Level 1	Level 1	Level 1 + 2	1 + 2	1 + 2
20-30%	Dev → So-Co	Dev → 30-40%	Dev	Dev
		Core	Core	Core

70-80

Prepading
leetcode
Level 1 2
MS
Virtual contest

AWS/CS
1
80-90%
Core
500
kickstart
CP 3
1
Google
Directi
Media.net
Tower Research
Uber

10:30

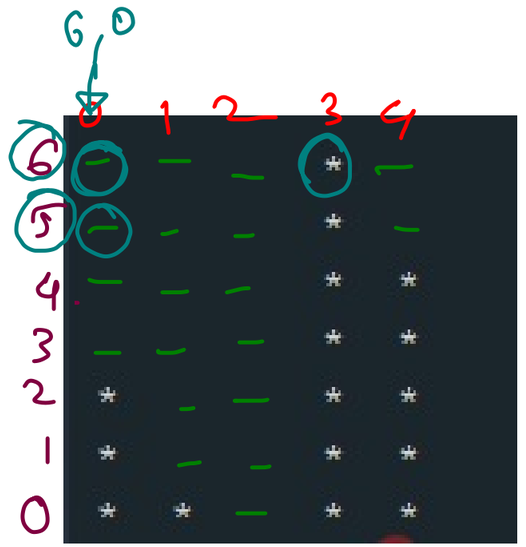
Codechef
Bit tricks DP/graphs

Tree, LL, Stack
leetcode

fin-tech
1
Aptitude
Maths + Level 1 2
Core

Bar Chart-

{ 3, 1, 0, 7, 5 }




```
int totalCols = n;
int totalRows = maxEle(arr);

for(int i=totalRows - 1; i>=0; i--){

    for(int j=0; j<totalCols; j++){

        if(/* ? */){
            System.out.print("*\t");
        } else {
            System.out.print("\t");
        }

    }
}
```

i < arr[j]

end of

i >= arr[j] : space

else i < arr[j] : star

Sum of two arrays

i1 = n1-1

	0	0	1	0	1	0	1	0	0
0	2	6	4	9	3	8	7	0	1

d1 = arr1[i1]

i2 = n2-1

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

d2 = arr2[i2]

idx = 0 idx = 1

0	2	6	4	5	4	2	1	0	9

i3 = n3-1

```
int idx = 0;
while(res[idx] == 0){
    idx++;
}

for(int i=idx; i<res.length; i++){
    System.out.println(res[i]);
}
```

long → $2^{63}-1$ → 10^{18}

{ 8 bytes }

carry = 0

arr2[-1] → index out of bound excptn

$> 10^{18}$ → Arrays → BigIntegers

factorial of a no.

n = 10!

size = max(n1, n2)+1

Pseudo Code

int [] res = new int [max(n1, n2) + 1]

carry = 0 i1 = n1-1 , i2 = n2-1 , i3 = n3-1

while (i3 >= 0) {

d1 = arr1[i1]
d2 = arr2[i2]

temp = carry + d1 + d2
quot = temp / 10
rem = temp % 10

carry = quot
res[i3] = rem;

i1--; i2--; i3--;

}

if (i1 < 0) d1 = 0
else d1 = arr1[i1]

if (i2 < 0) d2 = 0
else d2 = arr2[i2]