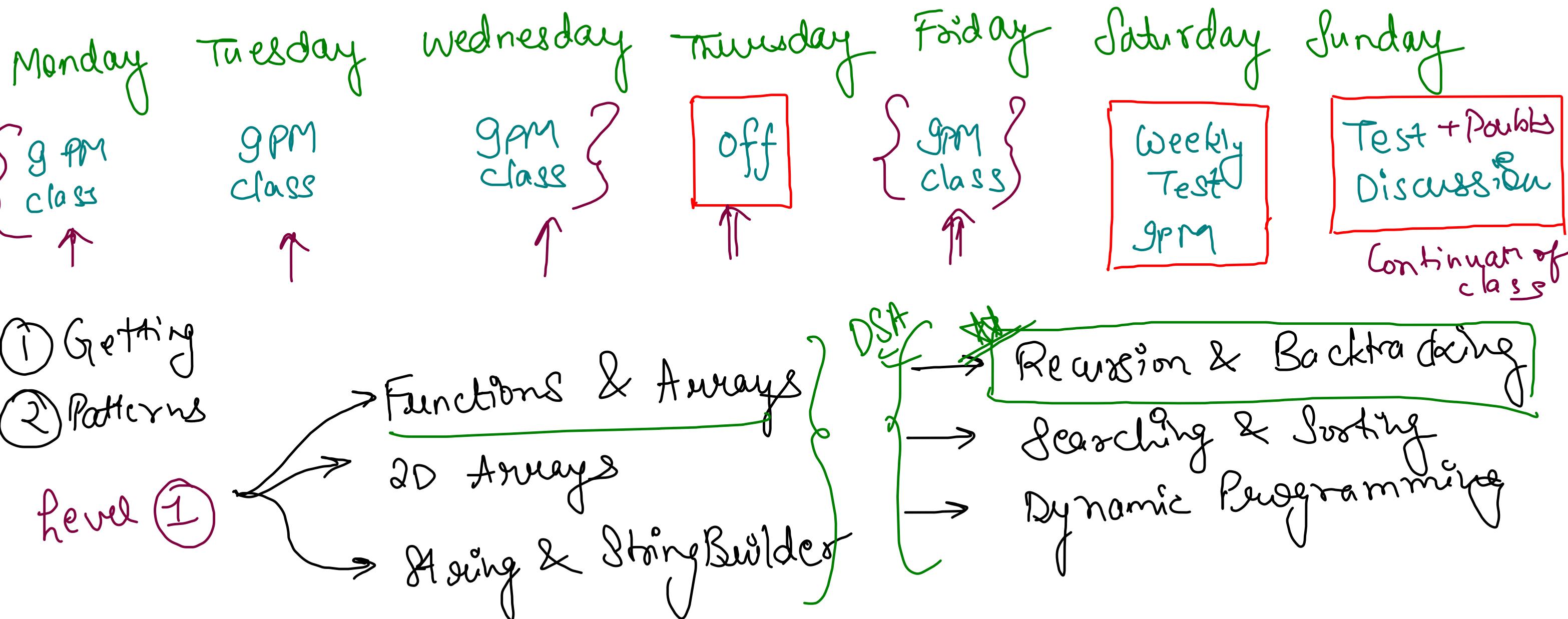


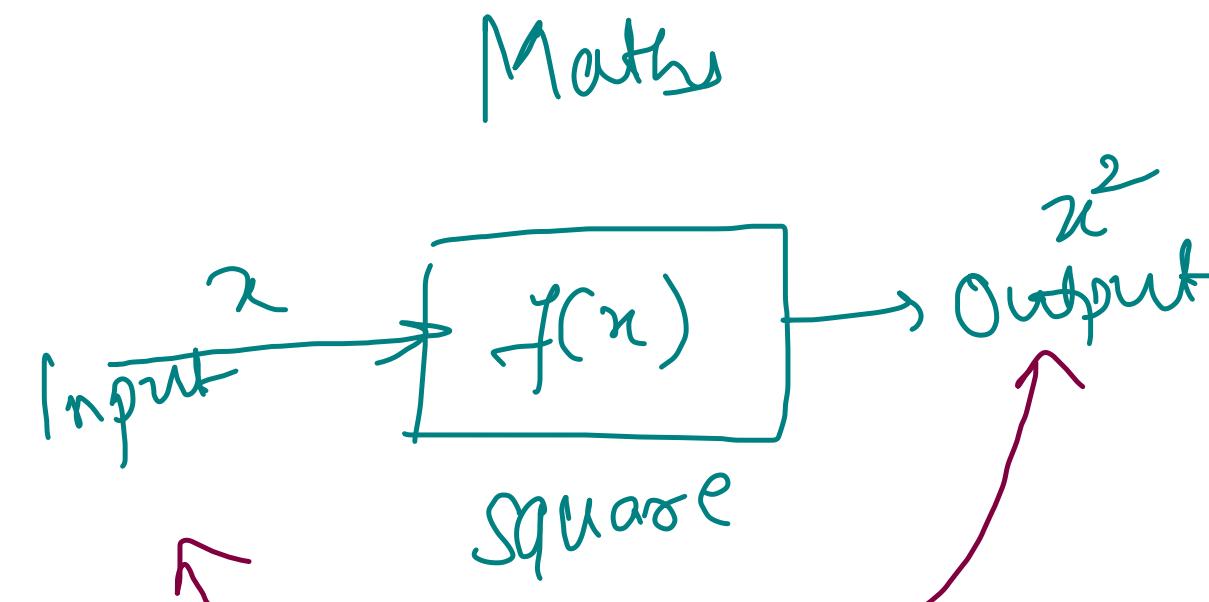
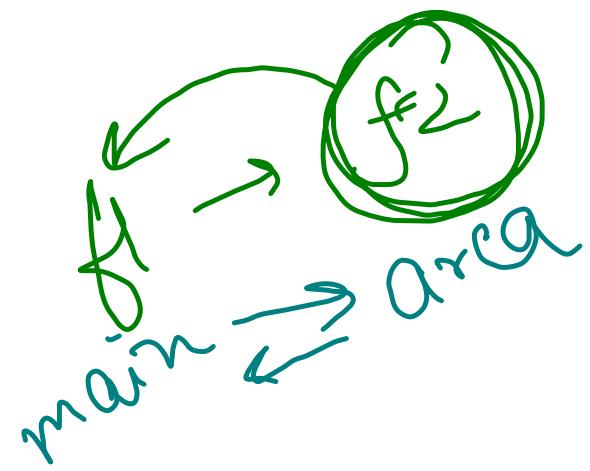
# Time Table { F-JP - 2 }



## My Introduction

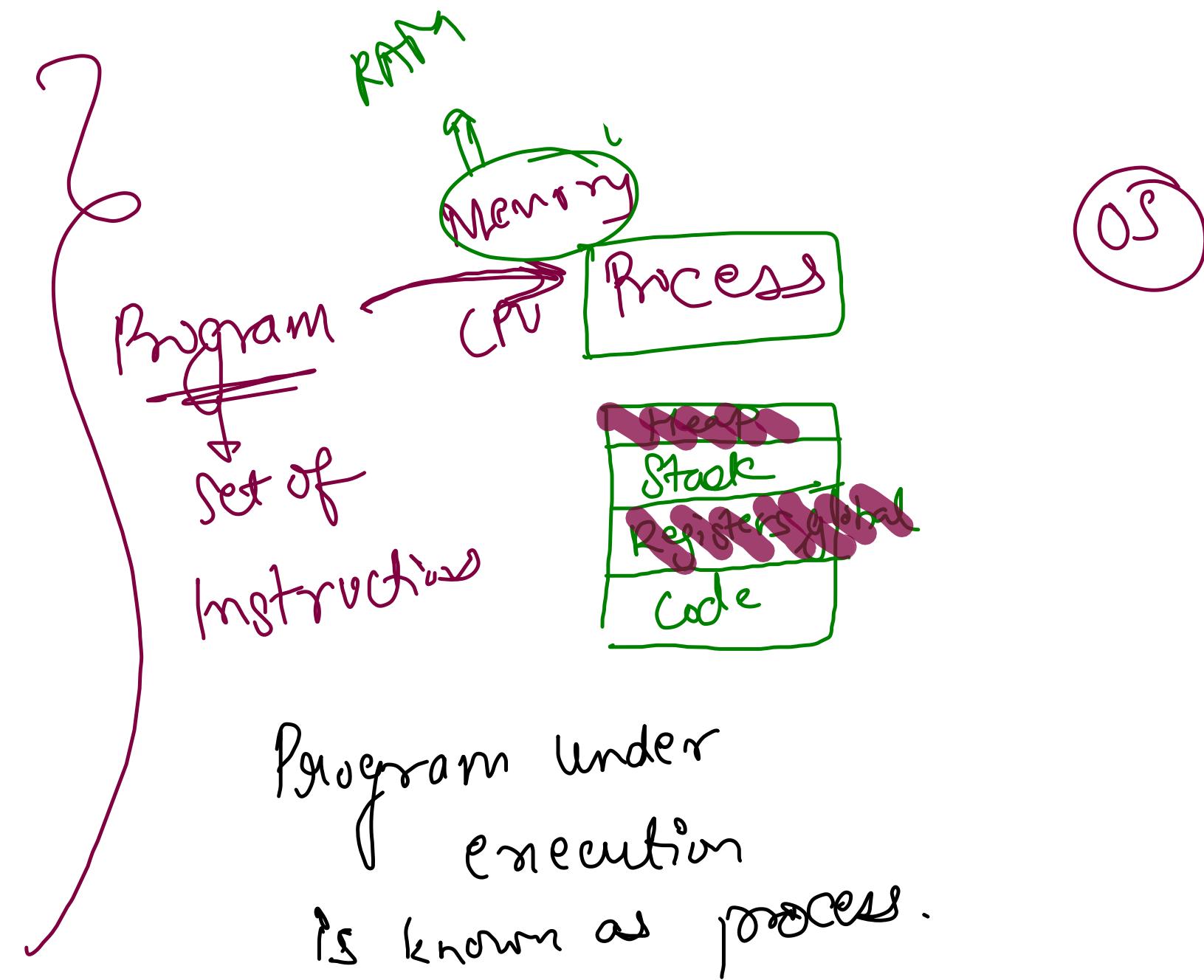
- Pointers
- Programming Articles (30 - 40+)
- Job-Switch program - 4 { 3-4 months }
- GeeksforGeeks
- LinkedIn
- Competitive Programming
- Product Based company
  - Google (3)
  - Salesforce (AMTS) (2)
  - Sprintel (1)

## # Functions



public static int volume( int l, int b, int h ) {  
 datatype function name

}



```

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);

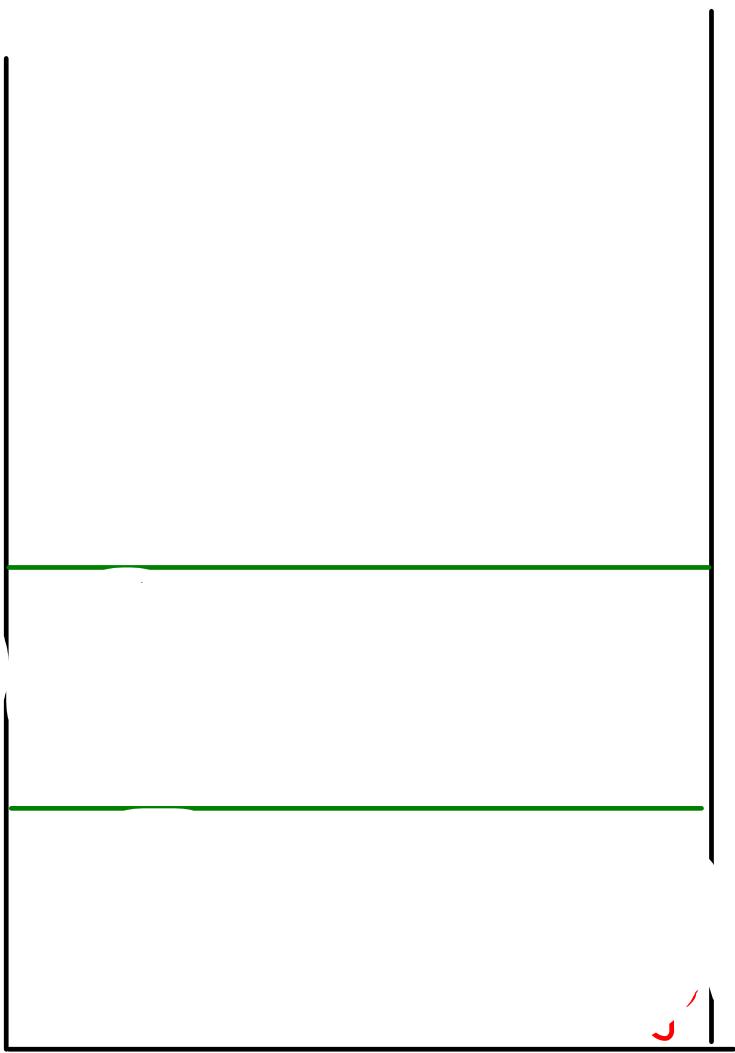
    int n = scn.nextInt();
    int r = scn.nextInt();

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

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

```

function call stack

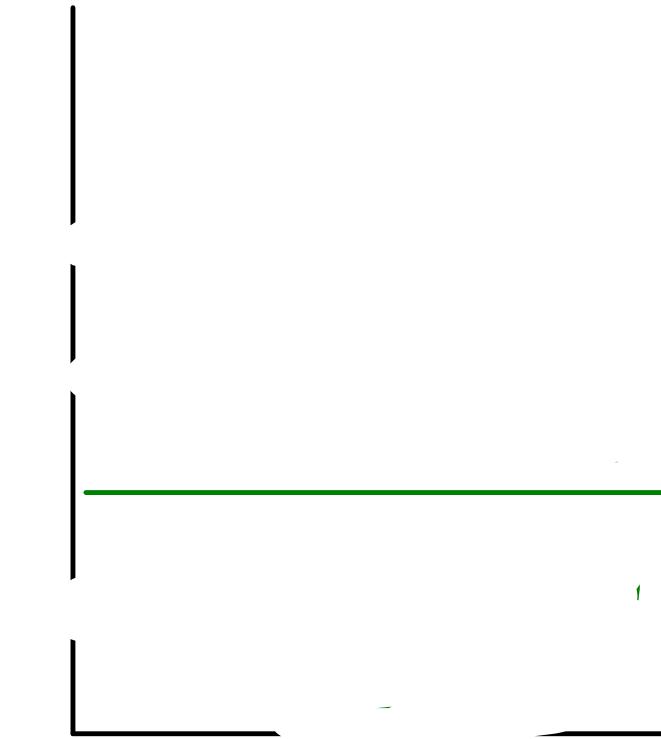


10

int  
void

$n =$ 
  
 $d =$

$\text{freq} =$



```

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;
    }

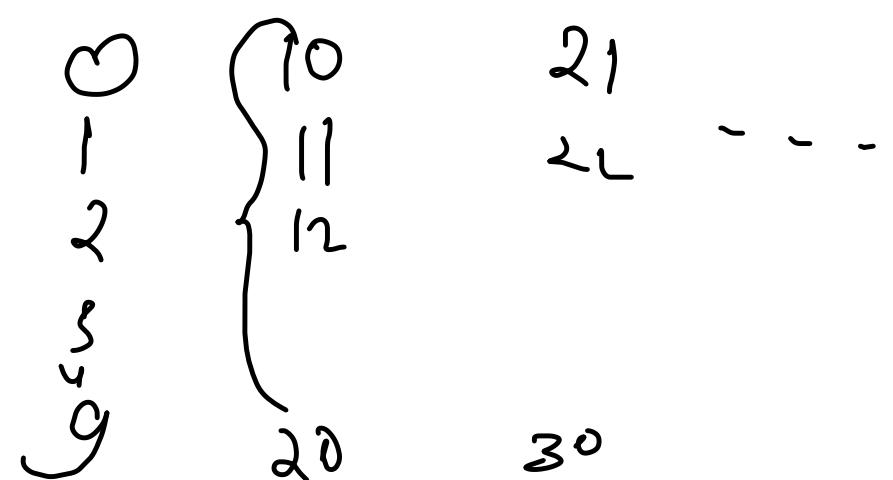
    return freq;
}

```

## Number system

~~#~~ Decimal No

0-9  
10 digits



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

$$100 = 99$$

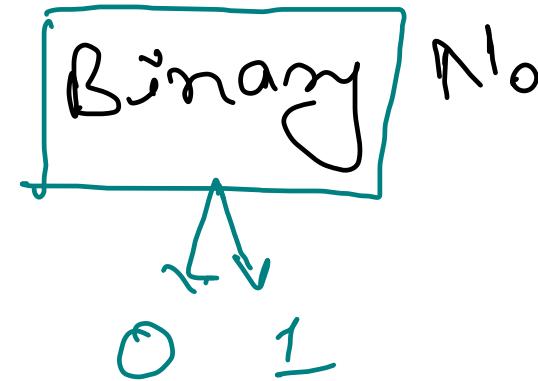
$$100 =$$

25 16.93

$$\begin{aligned}
 &= 6 * 10^2 + 9 * 10^1 + 3 * 10^0 \\
 &\quad + 1 * 10^3 + 5 * 10^4 \\
 &\quad + 2 * 1.5
 \end{aligned}$$

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

#



0	00
1	01
	10
	11

000
001
010
011
100
101
110
111

$2^0$        $2^1$

$$11001011 = 1 * 2^0 + 1 * 2^1 + 0 * 2^2 + 1 * 2^3 + 0 * 2^4 + \dots$$

min 0000  
 $2 * 2 * 2 * 2$   
01 01 01 01 max 1111  
Count  $2^4$

## Octal Nb System

↳ 0, 1, 2, 3, 4, 5, 6, 7

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

$$= 2*8^0 + 1*8^1 + 7*8^2 \\ + 4*8^3 + 3*8^4 + 5*8^5 + 7*8^6$$

6 month +  
↳ electrode L2  
Octal

$8*8*8*8*8$   
 $\underline{d_7\ d_6\ d_5\ d_4}$  ↳ min  $\Rightarrow 0000$   
max  $\Rightarrow 7777$   
total  $\Rightarrow 8^4$

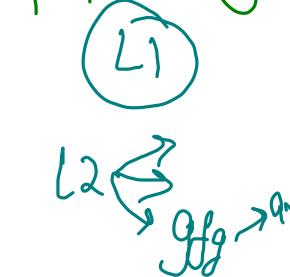
0	10	20	30	40	50	60
1	11	21	31	41	51	61
2	12	22	32	42	52	62
3	13	23	33	43	53	63
4	14	24	34	44	54	64
5	15	25	35	45	55	65
6	16	26	36	46	56	66
7	17	27	37	47	57	67

70	71	72	73	74	75	76	77	78	79
100	101	102	103	104	105	106	107	108	109
777	1000	7777	10000	77777	100000	777777	1000000	7777777	10000000
100000000	777777777	1000000000	7777777777	10000000000	77777777777	100000000000	777777777777	1000000000000	7777777777777
7777777777777	100000000000000	77777777777777	1000000000000000	777777777777777	10000000000000000	7777777777777777	100000000000000000	77777777777777777	1000000000000000000

$$(777)_8 = 1060 \\ 7*8^2 + 7*8^1 + 7*8^0 \\ \underline{1060} \\ 1*8^3$$

$$(gg)_10 = 1060 \\ 9*10^2 + 9*10^1 + 9*10^0 \\ \underline{1060}$$

preceding



service based

tier 3 ↲

tier 1 (3<sup>rd</sup>, 4<sup>th</sup>)

DSA

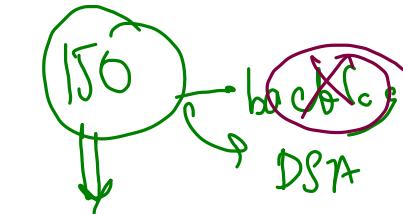


2<sup>nd</sup> Dev

DSA { 600f lectured }  
Dev { 50 + Project }

2-3 hr

8 hr



80-90  
↓  
40-50

# Today's Target

- </> Decimal To Any Base ] → classwork
- </> Any Base To Decimal ]
- </> Any Base To Any Base ] → homework
- </> Any Base Addition ] → classwork
- </> Any Base Subtraction ]
- </> Any Base Multiplication ] → homework

10:20

$$\begin{array}{rcl} \text{dividend} & & \text{Decimal to Any Base} \\ \downarrow & & \\ (\underline{\underline{634}})_{10} & = & (?)_8 \end{array}$$

$$\begin{array}{r}
 & x \\
 \xrightarrow{b} & 634 \\
 \xrightarrow{b} & 79 \\
 \hline
 & 634 \\
 & -79 \\
 \hline
 & 555 \\
 & -79 \\
 \hline
 & 476 \\
 & -79 \\
 \hline
 & 397 \\
 & -79 \\
 \hline
 & 318 \\
 & -79 \\
 \hline
 & 239 \\
 & -79 \\
 \hline
 & 160 \\
 & -79 \\
 \hline
 & 81 \\
 & -79 \\
 \hline
 & 2
 \end{array}$$

multiply  
 10 = 1  
 10 = 1  
 10 = 1  
 10 = 1

$\{0+2\} + 70\} + 100$   
 $= 117$   
 $(1172)g$

$$(634)_{10}$$

$$6 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0$$

→ What ?  
→ how ?  
Why ?

$$79 = 9 \cancel{4} 8 + 7$$

60 + 100

$$= 117$$

```
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;  
}
```

(1172) g

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

$$\begin{aligned}
 (634)_{10} &= 8^4 * (-) + 2 \\
 &= 8^4 * 0 + 8^3 * 1 + 8^2 * 1 + 8^1 * 7 + 8^0 * 2 \\
 79 &= 8^1 * (8^2 * 0 + 8^1 * 1 + 8^0 * 1) \\
 &= 8^3 * 0 + 8^2 * 1 + 8^1 * 1 + 8^0 * 7
 \end{aligned}$$

$$634 = 81 + 79 + 2$$

$$79 = 81 + 9 + 7$$

$$g = g' + (f + 1)$$

$$g = g' + (g' \times 0 + 1) + 1$$

$$g = g^2 * 0 + g^1 * 1 + g^0 * 1$$

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

173

The diagram illustrates the conversion of the decimal number 173 into an octal number. It shows four horizontal lines representing successive divisions by 8. The first division is labeled "21 - 5 \* 0^0". The second division is labeled "2 - 5 \* 0^1". The third division is labeled "0 - 2 \* 10^2". The remainders from each division are 5, 2, and 0 respectively. These remainders are written vertically along the left side of the lines. The quotient from the first division, 21, is written above the first line. The final result, 250, is written above the second line.

$$= 5 + 50 + 200 = (250)_8$$

# Any Base to Decimal

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

1172  
10  
117 - 2 \* 8<sup>0</sup>  
10  
11 - 7 \* 8<sup>1</sup>  
10  
1 - 1 \* 8<sup>2</sup>  
0 - 1 \* 8<sup>3</sup>

$$0 \cdot 2 * 8^0 + 7 * 8^1 + 1 * 8^2 + 1 * 8^3 = (634)_{10}$$

```
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
10	1100 - 1
10	110 - 0
10	11 - 0
10	1 - 1
10	0 - 1

\*  $2^0$   
 \*  $2^1$   
 \*  $2^2$   
 \*  $2^3$   
 \*  $2^4$   
 \*  $2^5$

$$\begin{aligned}
 &= 0 + 2 + 0 + 0 + 16 + 32 \\
 &\quad = 18 + 32 \\
 &\quad = (50)_{10}
 \end{aligned}$$

# Any base to any base

$$(172)_8 = (?)_2$$

Any base to decimal

$$(?)_{10}$$

Decimal to Any base

# Any Base addition

Carry 1

$$\begin{array}{r}
 & 0 & 1 \\
 & \cancel{4} & \cancel{3} \\
 + & (\cancel{7} & \cancel{5})_{10} \\
 \hline
 & 1 & 9 & 0
 \end{array}$$

$$\begin{aligned}
 \text{temp} &= (0 + 6 + 4) \\
 &\xrightarrow[10]{10} 1 \quad \{ \text{quot} \} \\
 &\xrightarrow[10]{10} 0 \quad \{ \text{rem} \}
 \end{aligned}$$

$$1 + 3 + 5 = \frac{9}{10} \quad 0$$

$$0 + 4 + 7 = \frac{11}{10} \quad 1$$

Carry

$$\begin{array}{r}
 1 \\
 0 \\
 0
 \end{array}
 \begin{array}{r}
 1 \\
 3 \\
 5
 \end{array}
 \begin{array}{r}
 0 \\
 8 \\
 8
 \end{array}$$


---


$$\begin{array}{r}
 1 \\
 2 \\
 1 \\
 2
 \end{array}$$

$$236 \rightarrow 23$$

$$236/10 = 23$$

①<sup>st</sup>

$$c + d_1 + d_2$$

$$= 0 + 0 + 4$$

1 (quot)

2 (rem)

②<sup>nd</sup>

$$1 + 3 + 5$$

$$= \frac{9}{8}$$

③<sup>rd</sup>

$$1 + 2 + 7 = \frac{10}{8}$$

④\*

$$1 + 0 + 0 = \frac{1}{8}$$

$$\begin{aligned}
 1212 &= 1000 + 200 \\
 &\quad + 10 + 2
 \end{aligned}$$

$$\left\{ 2 \times 10^0 + 1 \times 10^1 + 2 \times 10^2 \right. \\
 \left. + 1 \times 10^3 \right\}$$

base = 8

```
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;  
}
```

0 1 1 1 0  
0 0 ~~4~~ ~~6~~ ~~3~~  
0 0 ~~7~~ ~~2~~ ~~8~~  
-----  
res → 1 4 9 3  
-----

$$4 \times 100 + 93$$

$$\frac{1}{8} \leftrightarrow 0$$

$$\frac{12}{8} \leftrightarrow 4$$

[1000]  
mult

$$0+8+3=11$$



$$\begin{array}{r} 1 \\ 3 \\ \hline 9 \end{array}$$

## Pseudo Code

carry = 0

res = 0

multiplier = 1

while( $\frac{carry}{10} \parallel n_1 \parallel n_2 > 0$ ) {

$d_1 = n_1 \% 10$

$d_2 = n_2 \% 10$

temp = c + d1 + d2

quot = temp / base

rem = temp % base

carry = quot

res += temp \* multiplier

multiplier \*= 10

$n_1 /= 10$

$n_2 /= 10$

}

quotient will become  
next carry

88

rem will be  
Contributing in  
result.

# Any base subtraction

borrow

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

$$d_1 - d_2 + \text{borrow}$$

$$2 - 6 + 0 =$$

$$\circled{-4}$$

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

$$\circled{6}$$

951

$$\begin{aligned}
 & 2 - 2 + (-1) + 10 \\
 & = -1 + 10 = 9
 \end{aligned}$$

$$\begin{aligned}
 & 1 - 5 + (-1) \\
 & = -4 - 1 = \circled{-5} + 10 \\
 & = 8
 \end{aligned}$$

# Any base subtraction

borrow

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

$-4 + 0$

$\boxed{Q - d_1 + \text{borrow}}$

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

$\approx 6$

$$\begin{aligned}
 2 - 6 &= -4 + 10 \\
 &\approx 6
 \end{aligned}$$

951

$$\begin{aligned}
 2 - 2 + (-1) + 10 \\
 &= -1 + 10 = 9
 \end{aligned}$$

$$\begin{aligned}
 1 - 5 + (-1) \\
 &= -4 - 1 = -5 + 10 \\
 &= 8
 \end{aligned}$$

## Pseudo code

```

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;
}
  
```

$$n_2 > n_1$$

$$\begin{array}{r}
 0 \quad -1 \quad -1 \quad 0 \\
 (1) \quad 4 \quad 1 \quad 8 \\
 (2) \quad 5 \quad 6 \quad 8 \\
 \hline
 1 \quad | \quad 3 \quad 4
 \end{array}$$

$$\begin{aligned}
 & 1 - 0 + 0 \\
 & = 1
 \end{aligned}$$

$$\begin{aligned}
 & 4 - 2 + (-1) \\
 & = 2 - 1 = 1
 \end{aligned}$$

$$\begin{aligned}
 & 1 - 5 + (-1) \\
 & -4 - 1 = -5 + 8 \\
 & = 3
 \end{aligned}$$

# Any base Multiplication

$$1 \left( \begin{array}{cccc} 1 \\ 2 & 3 & 4 & 0 \\ \hline 1 & 5 & 6 \end{array} \right)_{10}$$

$$\times \left( \begin{array}{cc} 7 & 4 \end{array} \right)_{10}$$


---

$$\left\{ \begin{array}{cccc} 18 & 6 & 2 & 4 \end{array} \right\}$$

$$\left[ \begin{array}{cccc} 15 & 0 & 9 & 2 \end{array} \right] \times$$


---

$$\underline{15 \ 9 \ 5 \ 4 \ 4}$$

Any base  
Addition

$$7+2+1 = \frac{15}{10} \rightarrow 15$$

$$\frac{24}{10} \rightarrow 2 \quad 4$$

$$7+6 = \frac{42}{10} \rightarrow 4$$

$$\frac{39}{10} \rightarrow 3 \quad 9$$

$$5+4+2 = \frac{22}{10} \rightarrow 2$$

$$4+1+2 = \frac{6}{10} \rightarrow 0$$

$$\frac{8}{10} \rightarrow 0$$

$$\frac{10}{10} \rightarrow 1$$

Algorithm

while {  $n_1 > 0$  } {

- ① Extract digits of  $n_1$  :  $d_1$
- ② multiply  $d_1$  with  $n_2$
- ③  $res = \text{anybaseAddition}(res, \text{temp} * \text{multiplier})$   
multiplier  $\times = 10$

getProductWithDigit()

*pencil  
eraser*

{

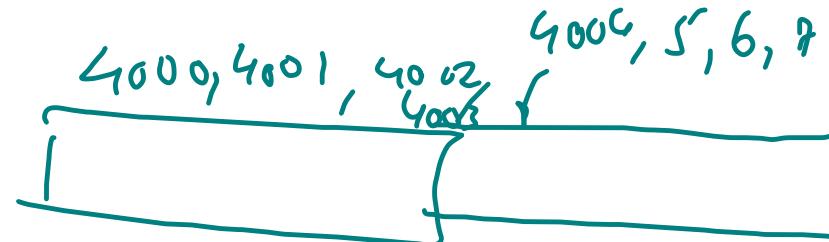
$$\begin{array}{r}
 & 1 & 1 & 5 & 5 & 6) \\
 & (2 & 1 & 5 & 7 & 8 \\
 & \times & (7 & ) & 4) \\
 \hline
 & 1 & 0 & 6 & 7 & 0) \\
 & (1 & 7 & 4 & 0 & 2 \\
 \hline
 & 2 & 0 & 4 & 7 & 1 & 0 \\
 \hline
 & 2106 * 74
 \end{array}$$

$$= 2156 * 4 + \textcircled{+} \quad \text{Total}$$

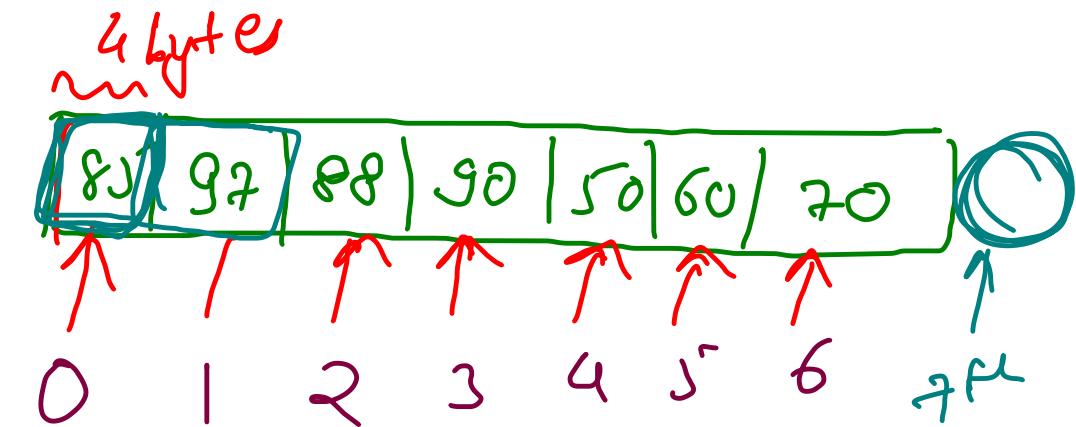
$$\begin{array}{r}
 24 \over 8 \rightarrow 3 \\
 20 + 3 = 23 \over 8 \rightarrow 2 \\
 6 \over 8 \rightarrow 0 \\
 40 \over 8 \rightarrow 5 \\
 40 \over 8 \rightarrow 5 \\
 8 \over 8 \rightarrow 1 \\
 1 \over 8 \rightarrow 0 \\
 15 \over 8 \rightarrow 1 \\
 1 \over 8 \rightarrow 0
 \end{array}$$

## Data Structure

int m1 = 95;  
int m2 = 97;  
int m3 = 82;



## Arrays



contiguous  
memory  
allocation

7 size  
0 - 6

$$7 \times 4 = 28 \text{ bytes}$$

## Memory mapping of arrays

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

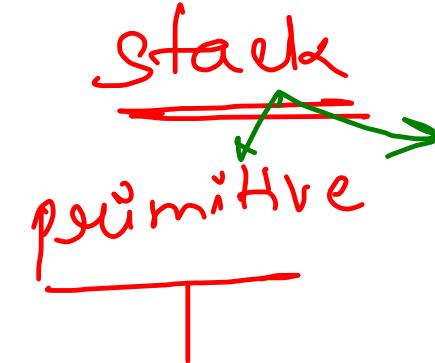
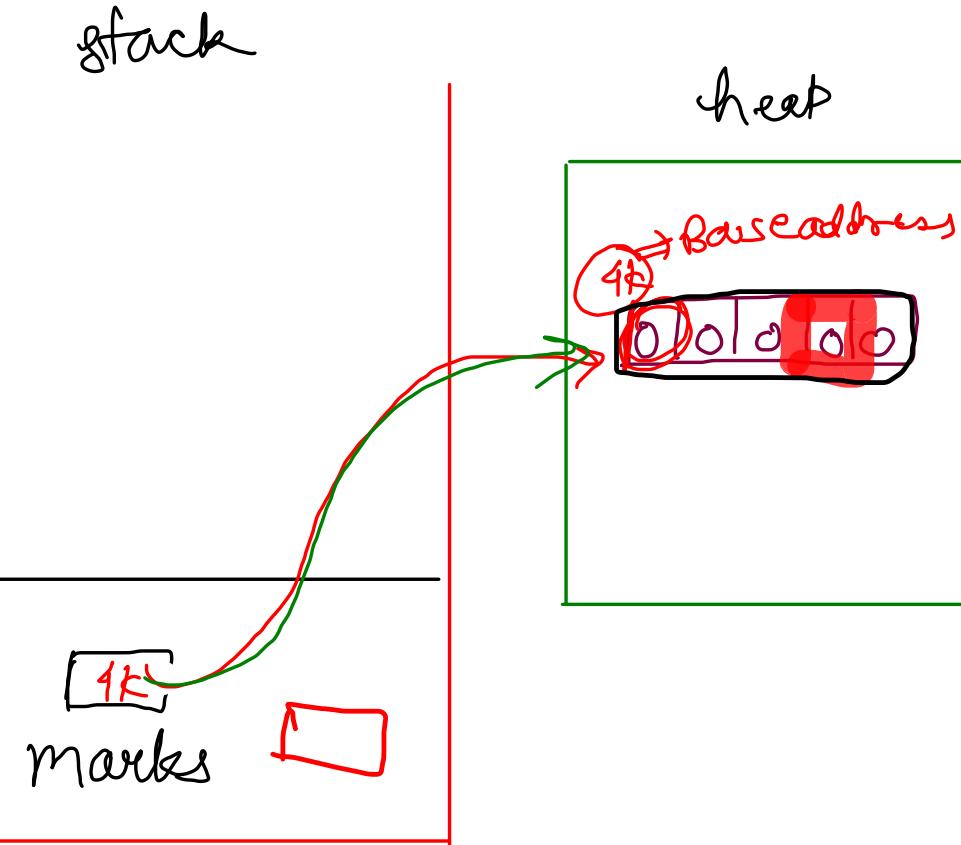
{Static array}

marks[1]

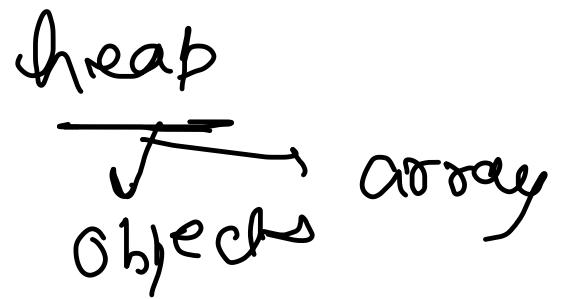
base address +  $i^{th}$  block  
of array  
 $\times$   
4 bytes

Random access {constant operation}

$\Rightarrow$  BA + index \* size of one block



referencing  
if



marks[0]

$$4k + 0 \times 4 = 4k$$

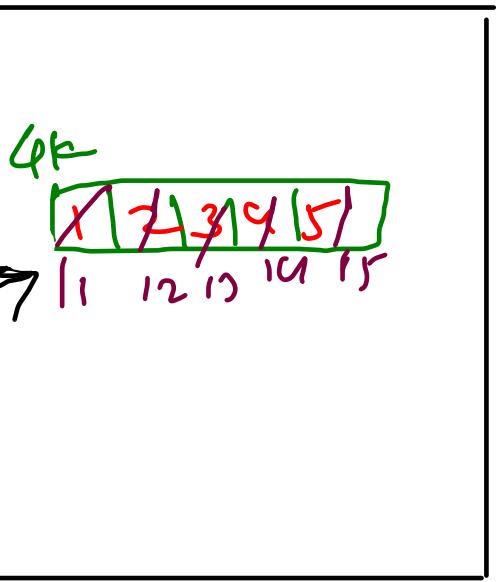
marks[3]

$$4k + 3 \times 4 = 4k12$$

## Passing array to function

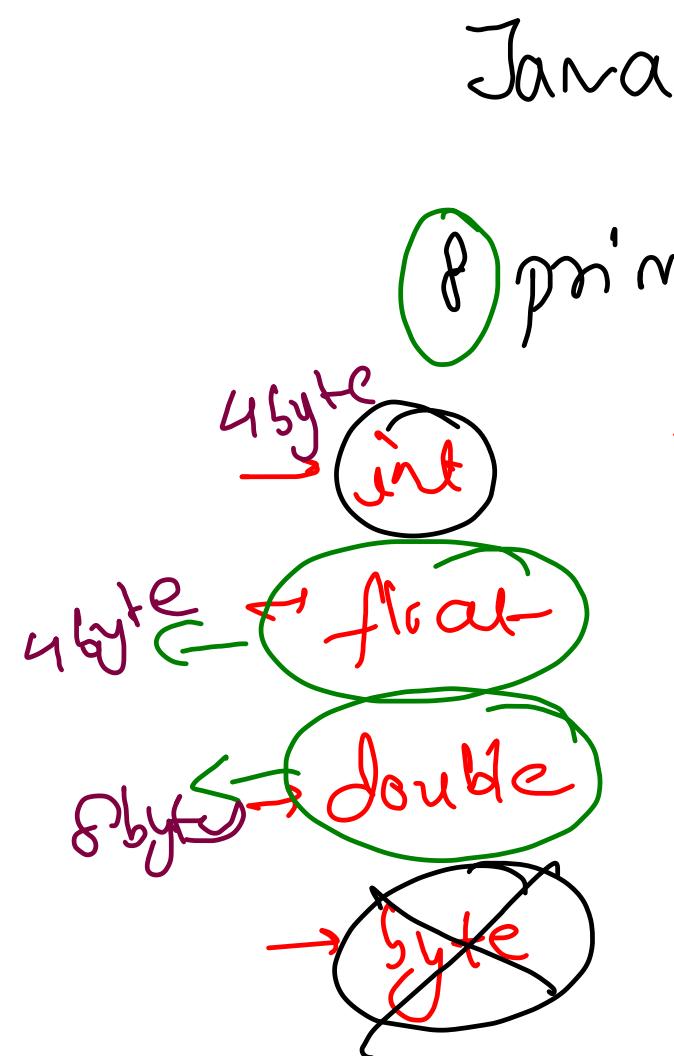
"Changes in the heap always persist"  
Shallow copy permanent

```
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);  
}
```

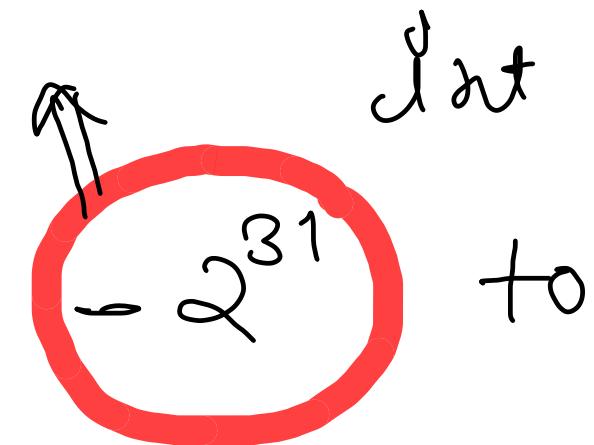


"Java is always

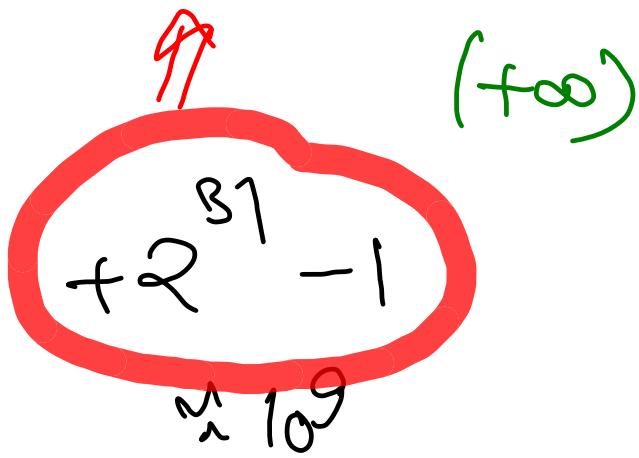
pass by value  
Shallow copy



(-2<sup>31</sup>) Integer.MIN-VALUE



Integer.MAX-VALUE



long

$$-2^{63}$$

to

$$2^{63} - 1$$

$\downarrow$								
63	78	32	46	26	92	16	104	2

$$\text{Span} = \underline{\max^m} - \underline{\min^m}$$

$$\max^m = -\infty \cancel{63} \cancel{78} \cancel{92} 104$$

$$\min^m = +\infty \cancel{63} \cancel{32} \cancel{26} 162$$

```

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);

```

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

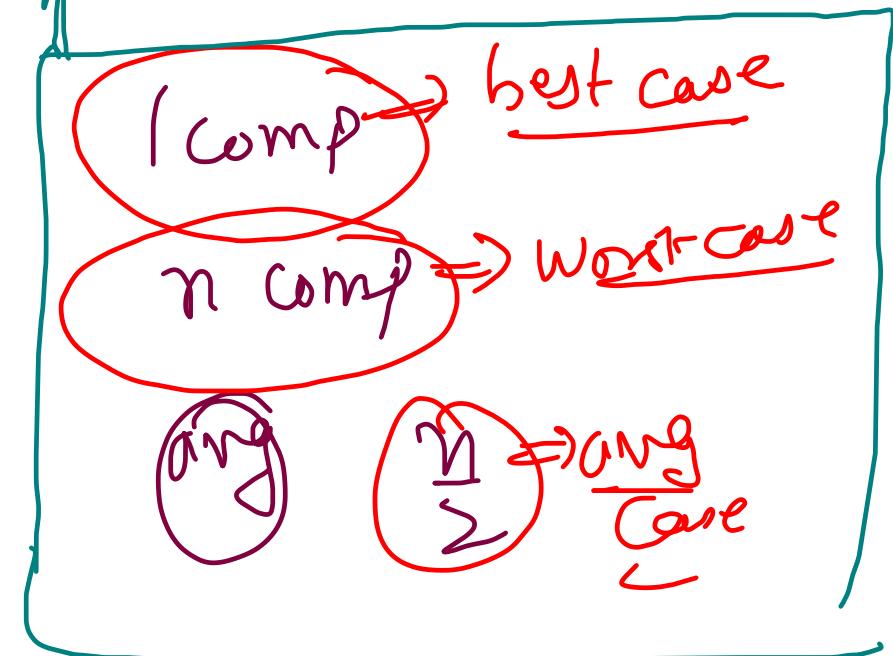
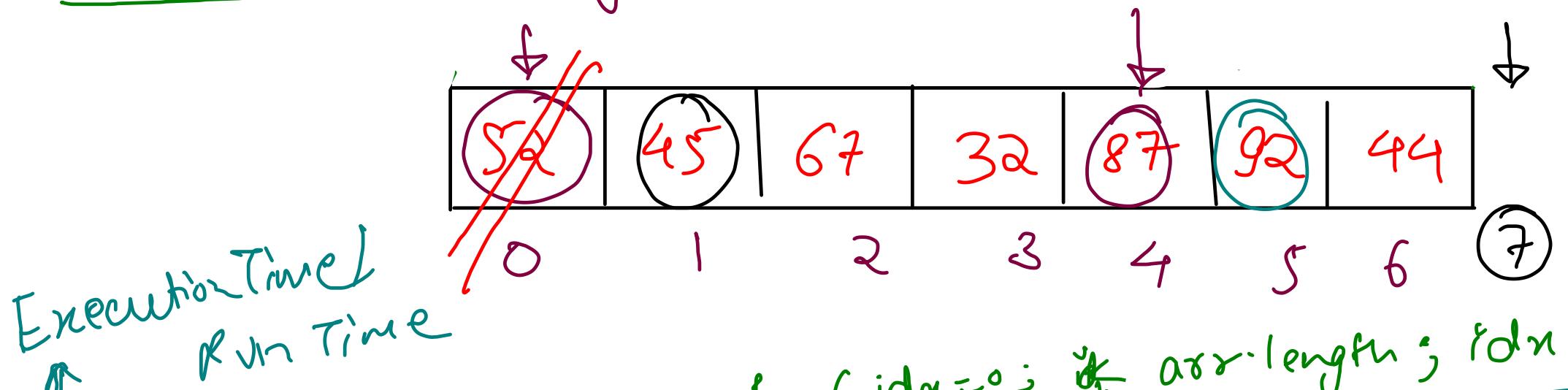
$x + 0 = x$

## Today's Questions

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

→ Linear Search Algorithm

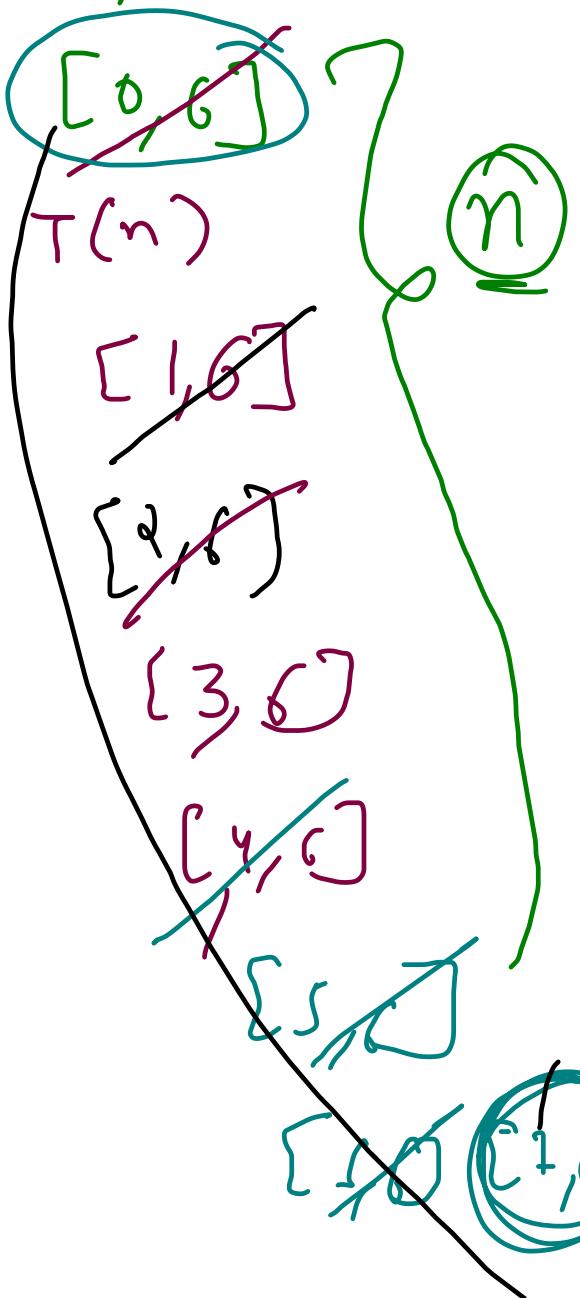
## Linear Search Algorithm



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

ele = ~~87~~ ~~45~~ ~~67~~ 56

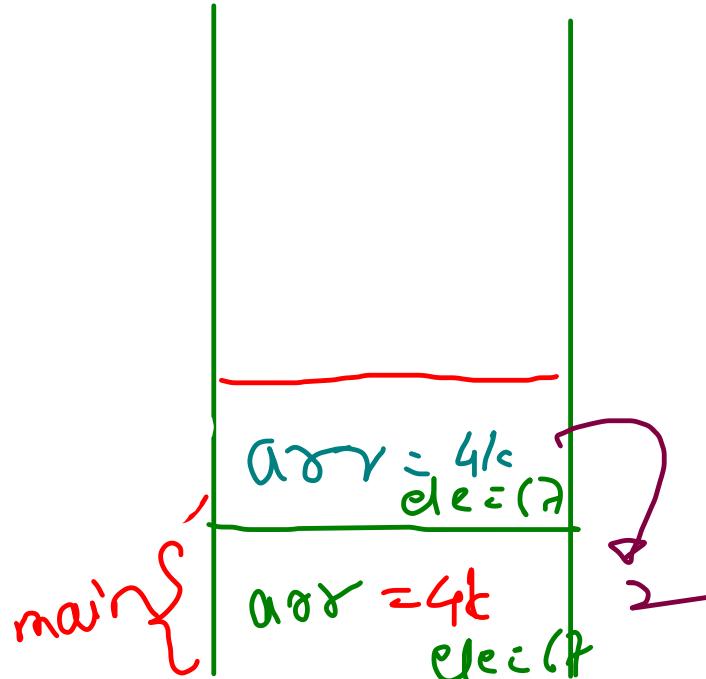
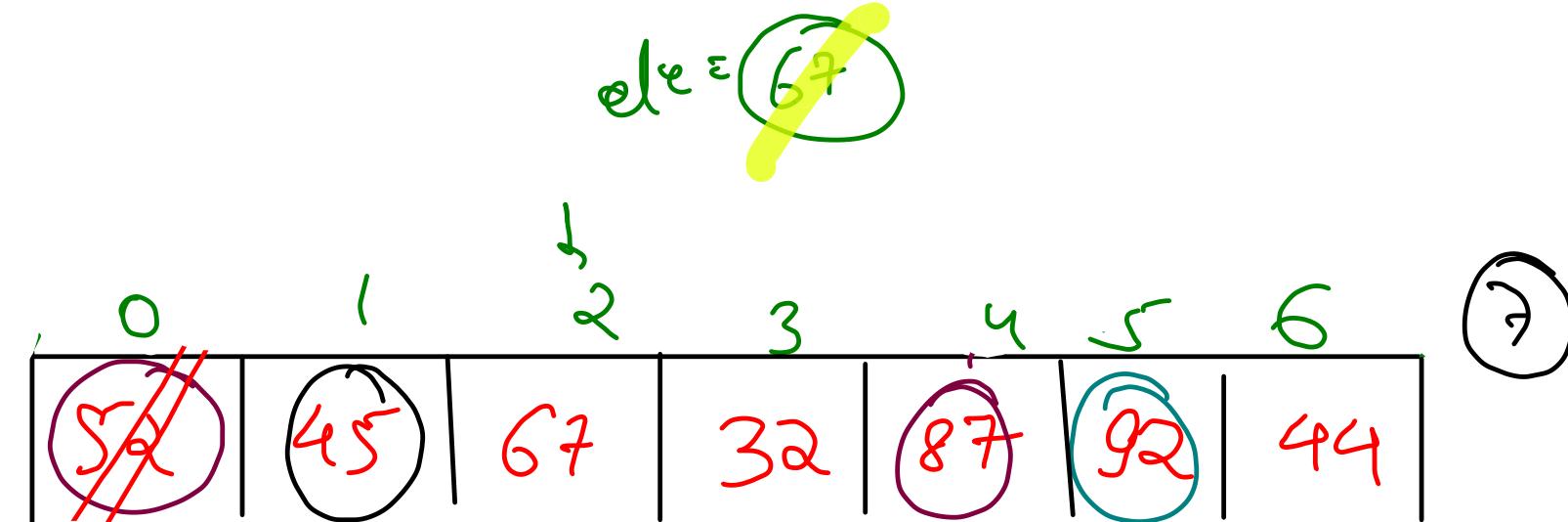
search space



```

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)
}

```



```

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

    int[] arr = new int[n];

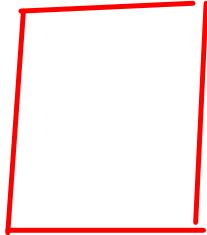
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    int ele = scn.nextInt();

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

```

## Notes

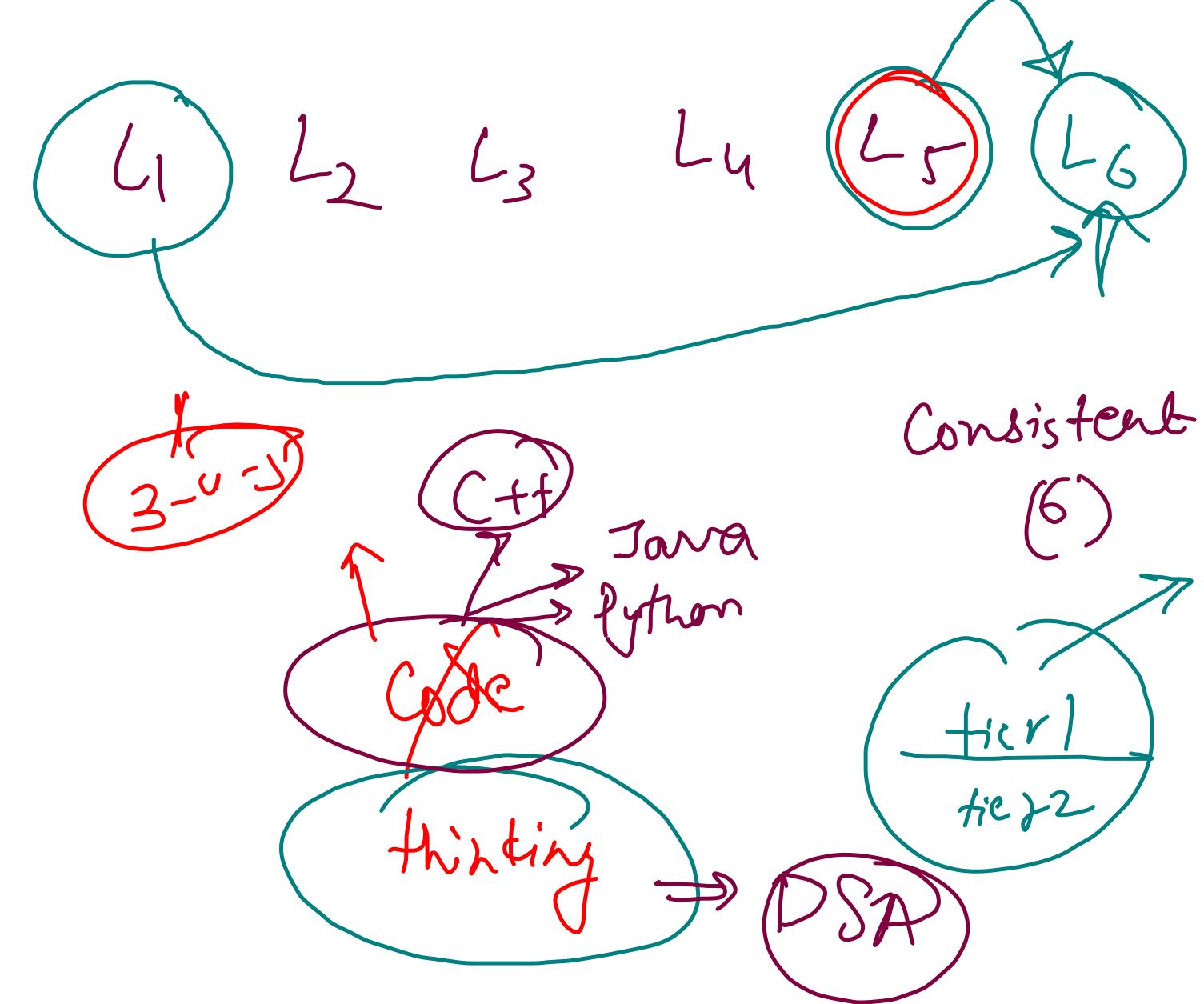


Question name  
1 liner logic

Important points



## Revision



DSA + Dev + Core

Hypercoding  
leetcode  
level 2  
MB  
Virtual context

SB-C.  
(3-6)

Aptitude  
Level 1

20-30%

Small Startups  
(6-12)

Level 1

Dev → 50-60%  
Core

big startups  
(12-30)

Level 1

+2  
Dev → 30-40%  
Core

70-80

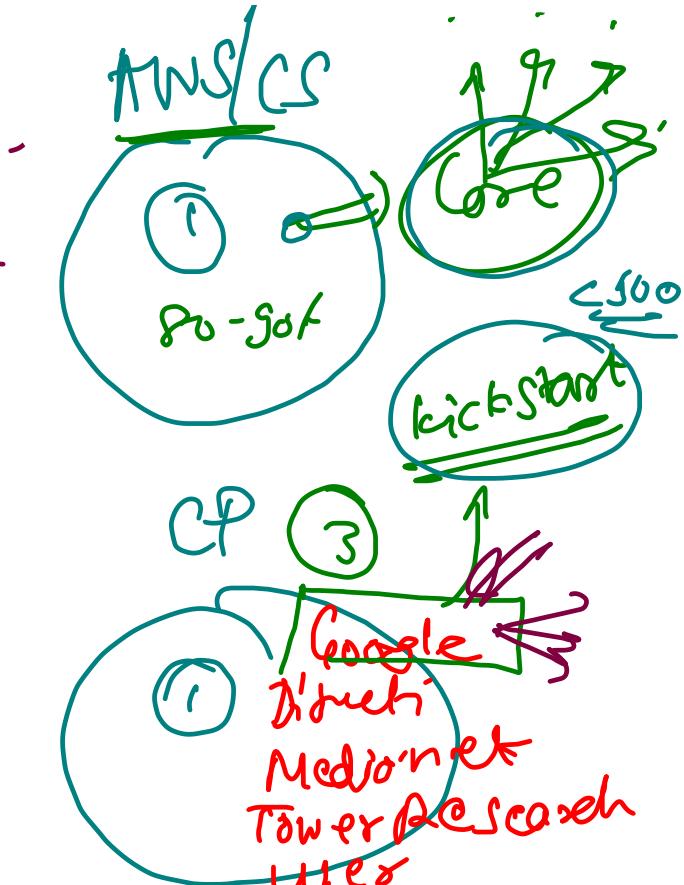
PBC  
SOF

1  
+2

new  
Core

tech  
giant

{FAANG}



10-30

Codechef  
Bit/stdc++ DP/graphs

Tree, LL, STQ  
leetcode

fin-tech  
Aptitude  
Math level 1/2  
Core

# Bar Chart

{ 2, 1, 0, 2, 5 }

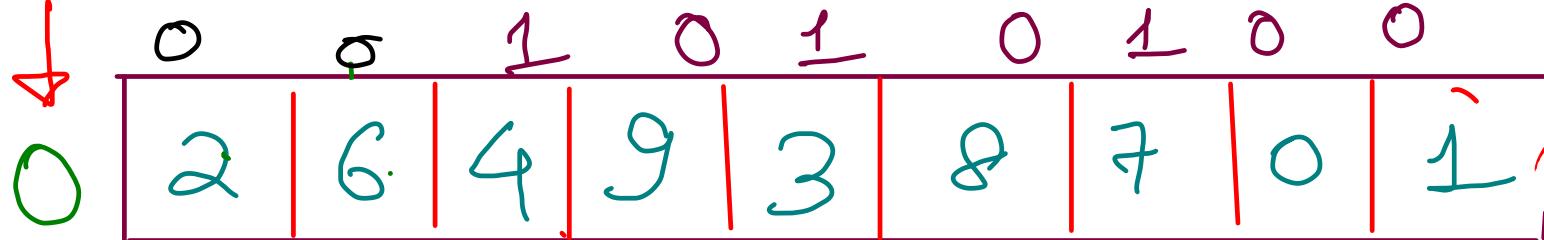
```
int totalCols = n;
int totalRows = maxEle(arr);
for(int i=totalRows - 1; i>=0; i--){
    for(int j=0; j<totalCols; j++){
        if(*arr[j] >= i)
            System.out.print("*\t");
        else {
            System.out.print("\t");
        }
    }
    System.out.println();
}
```

6	5	4	3	2	1
*	*	*	*	*	
*	*	*	*	*	
*	*	*	*	*	
*	*	*	*	*	
*	*	*	*	*	
*	*	*	*	*	

$i \geq arr[j]$  : Space  
else  $i < arr[j]$  : Star

# Sum of two arrays

$$i1 = n1 - 1$$



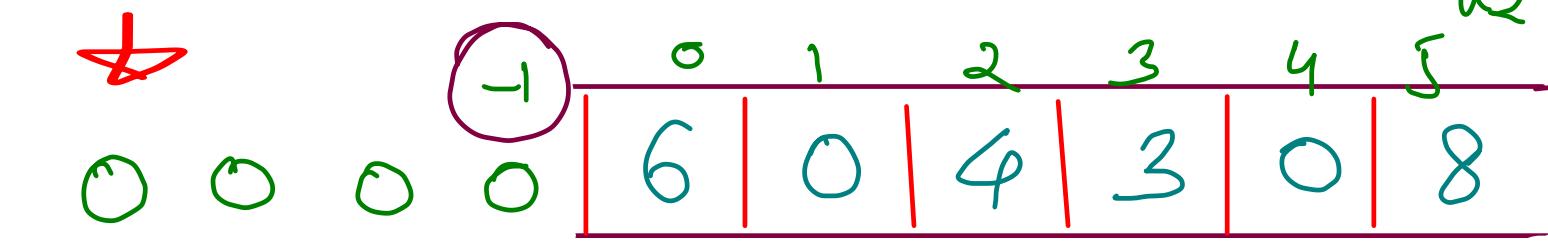
$$d_1 = \text{arr1}[i1]$$

long {8 bytes}

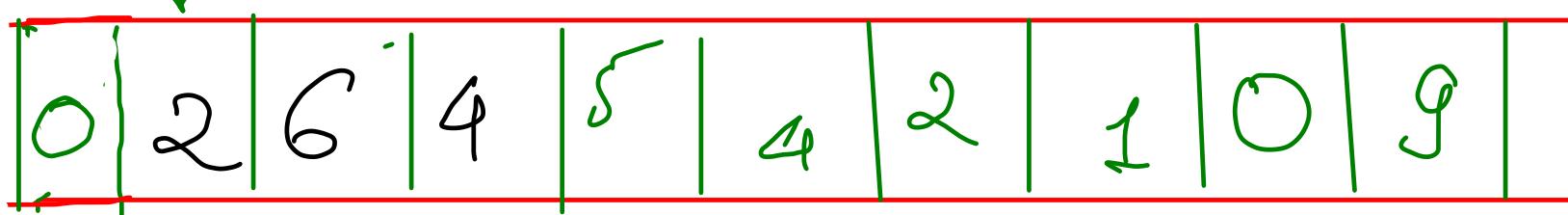
long → 2<sup>63</sup> - 1

→ 10<sup>18</sup>

$$\text{carry} = 0$$



$$\text{id}_n = 0 \quad \text{id}_n = 1$$



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

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

$$i) = n3 - 1$$

$$\text{man}(n_1, n_2) + 1$$

$$n = \underline{\underline{10^9}}$$

> 10<sup>18</sup>

→ Arrays

BigInteger

factorial of  
c n'

## Pseudo Code

```
synt []res = new int { man(n1, n2) + r ]
```

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

while (      i3 ≥ 0      ) {

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

temp = c + 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]

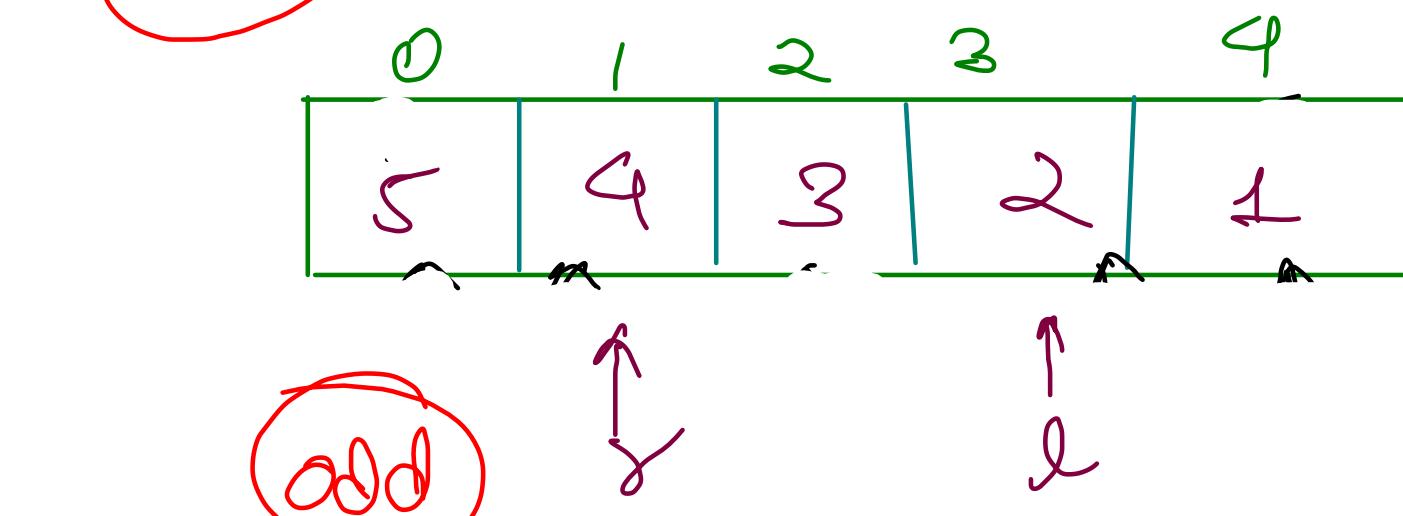
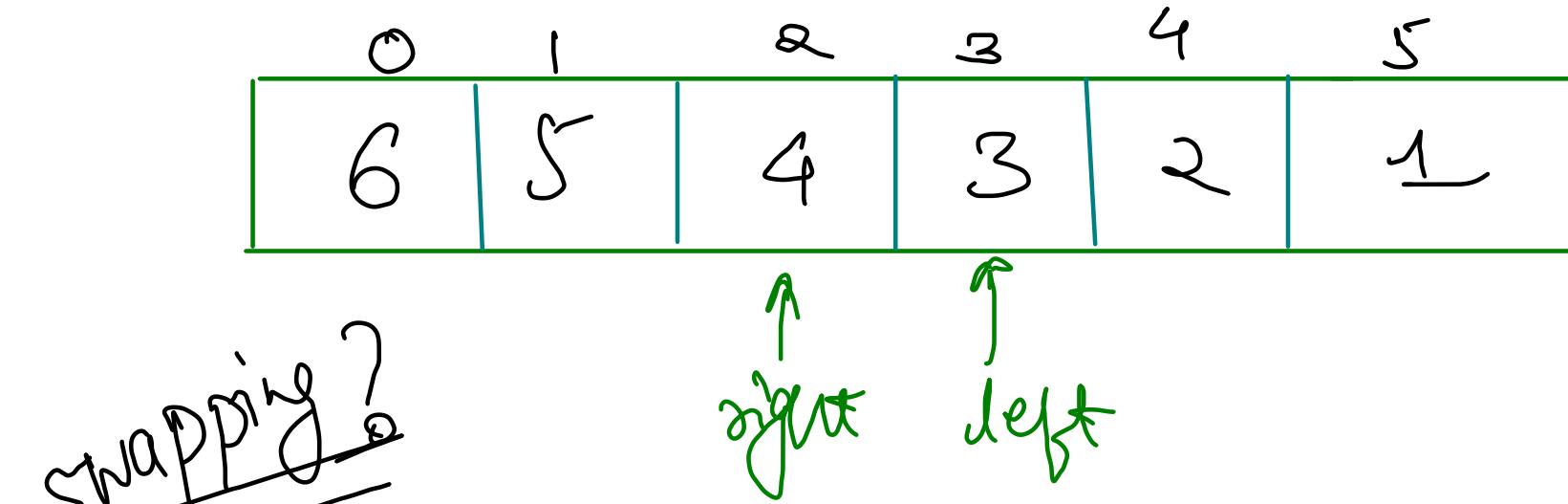
if (i2 < 0)      d2 = 0

else      d2 = arr2[i2]

## Today's Questions

- </> Reverse An Array
- </> Rotate An Array
- </> Inverse Of An Array
- </> Subarray Problem
- </> Subsets Of Array

## Reverse Array



even

$l > r$

$l > r$

swap (arr, left, right);  
temp  $\leftarrow$  arr[left]  
arr[left] = arr[r]  
arr[r] = temp

```
public static void swap(int[] arr, int left, int right){  
    int temp = arr[left];  
    arr[left] = arr[right];  
    arr[right] = temp;  
}  
  
public static void reverse(int[] a){  
    int left = 0, right = a.length - 1;  
  
    while(left < right){  
        swap(a, left, right);  
        left++; right--;  
    }  
}
```

①

```
public static void swap(int[] arr, int left, int right){  
    int temp = arr[left];  
    arr[left] = arr[right];  
    arr[right] = temp;  
}
```

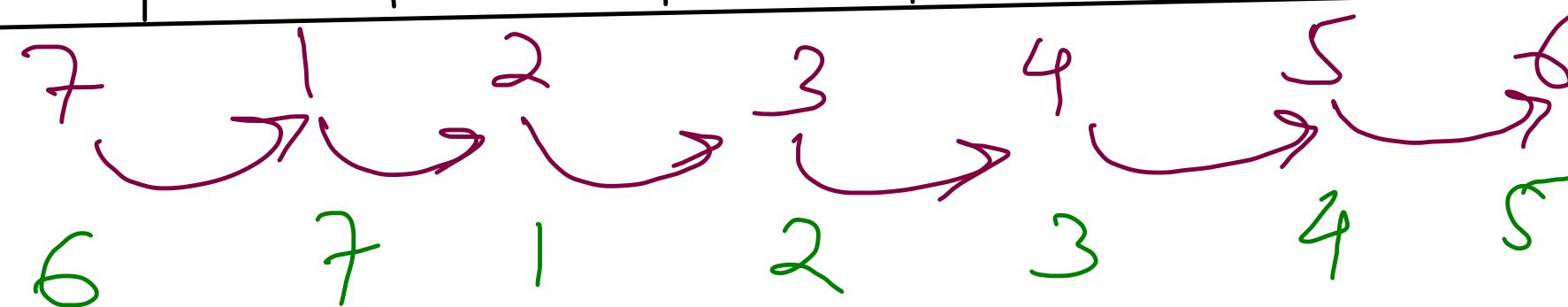
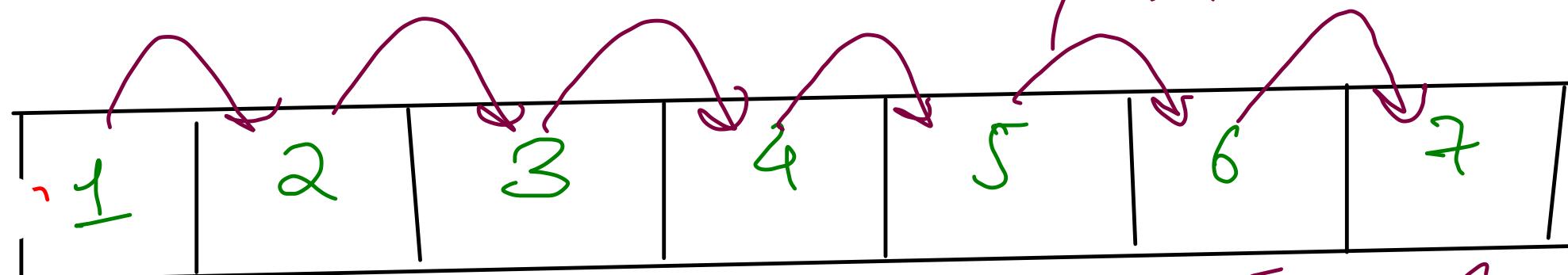
heap  
change

②

```
public static void swap(int val1, int val2){  
    int temp = val1;  
    val1 = val2;  
    val2 = temp;  
}
```

stack  
change

## Rotate an Array

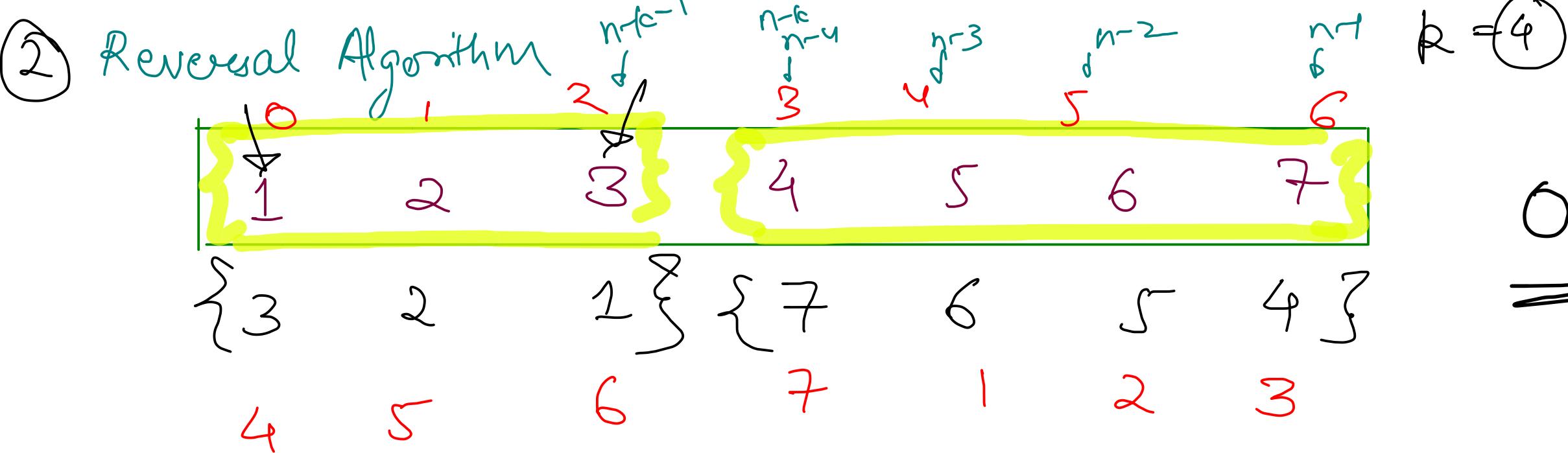


① Brute force  
    { Rotate Array  $k$  times }  
    by Inplace  
    Naïve Approach

$k = 18$   
[0 7 -]  
 $R = 1 \cdot n$   
if ( $k < 0$ )  
 $k = k + n$

4

$O(k * n)$



~~$O(n)$~~  (Swaps in worst case)

- ① Reverse (arr, 0,  $n-k-1$ )  $\xrightarrow{(n-k)/2}$
- ② Reverse (arr,  $n-k, n-1$ )  $\xrightarrow{k/2}$
- ③ Reverse (arr, 0,  $n-1$ )  $\xrightarrow{n/2}$

$$\frac{n-k}{2} + \frac{k}{2} + \frac{n}{2}$$

```

public static void reverse(int[] a, int left, int right){
    while(left < right){
        swap(a, left, right);
        left++; right--;
    }
}

public static void rotate(int[] a, int k){
    int n = a.length;

    // Handle Larger K Values
    k = k % n;

    // Handle Negative K Values
    if(k < 0){
        k = k + n;
    }

    // 1. Reverse First Block |
    reverse(a, 0, n - k - 1);

    // 2. Reverse Second Block
    reverse(a, n - k, n - 1);

    // 3. Reverse Entire Array
    reverse(a, 0, n - 1);
}

```

6 participants raised hand

Why ?

$$A = A_1 + A_2$$

$$\text{Res} = A_2 + A_1 \quad \text{↙}$$

$$(A_1' + A_2') \quad \text{↑ } \quad \text{↑ }$$

$$\underline{\underline{321 + 7654}}$$

$$A_2 + A_1$$

## Inverse of a No

a[i]	0	1	2	3	4	5	6
	2	4	1	3	6	0	5

b[j]	0	1	2	3	4	5	6
	5	2	0	3	1	6	4

inplace { ✓ / ✗ }

new resultant array

```

public static int[] inverse(int[] a){
    int n = a.length;
    int[] b = new int[n];

    for(int i=0; i<n; i++){
        // a[i] -> i
        int j = a[i];
        b[j] = i;
    }

    return b;
}

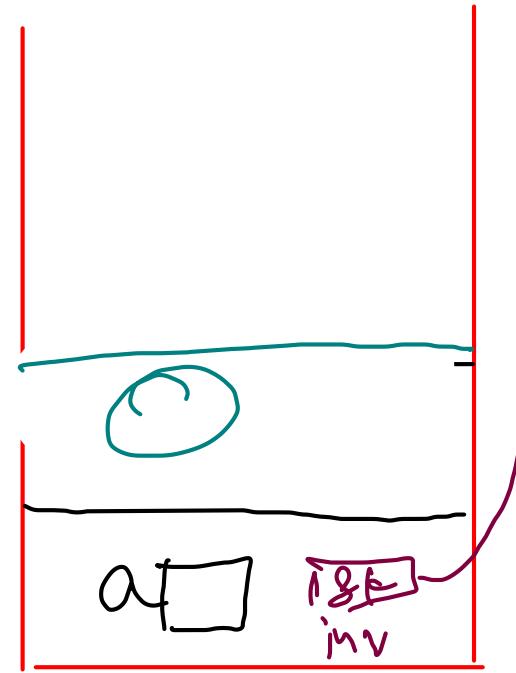
```

## Mock interview {for inplace} Algo

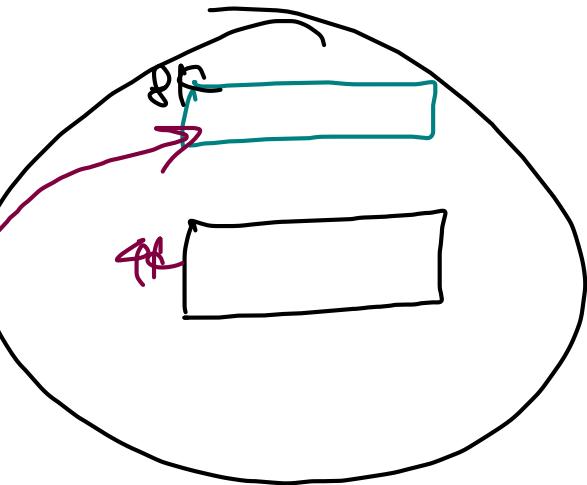
$$a[i] \rightarrow j$$

$$b[j] \rightarrow i$$

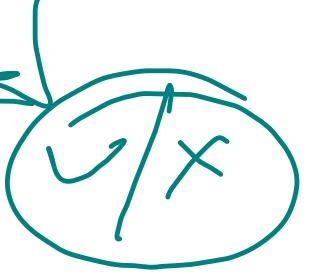
Stack



heap



## Subset / Subsequence

Each element  
2 choices  




$\{1\}$     $\{2\}$     $\{3\}$     $\{4\}$     $\{5\}$   
 $\{1, 2\}$     $\{1, 3\}$     $\{1, 4\}$     $\{1, 5\}$   
 $\{2, 3\}$     $\{2, 4\}$     $\{2, 5\}$   
 $\{3, 4\}$     $\{3, 5\}$   
 $\{4, 5\}$   
**total subset**  
 $2^n$

Subarray / Substring

- Order must be same
- Contiguous subpart of array

$\{1\}$     $\{2\}$     $\{3\}$     $\{4\}$     $\{5\}$   
 $\{1, 2\}$     $\{2, 3\}$     $\{3, 4\}$     $\{4, 5\}$   
 $\{1, 2, 3\}$     $\{2, 3, 4\}$     $\{3, 4, 5\}$   
 $\{1, 2, 3, 4\}$     $\{2, 3, 4, 5\}$   
 $\{1, 2, 3, 4, 5\}$

total subarray  
 $= \frac{n * n + 1}{2}$

# Today's Questions

- ① subarrays } code
  - ② sub sets
  - ③ 2D array demo
- ↓  
Decoder & Initialization  
Input & Output  
Memory mapping

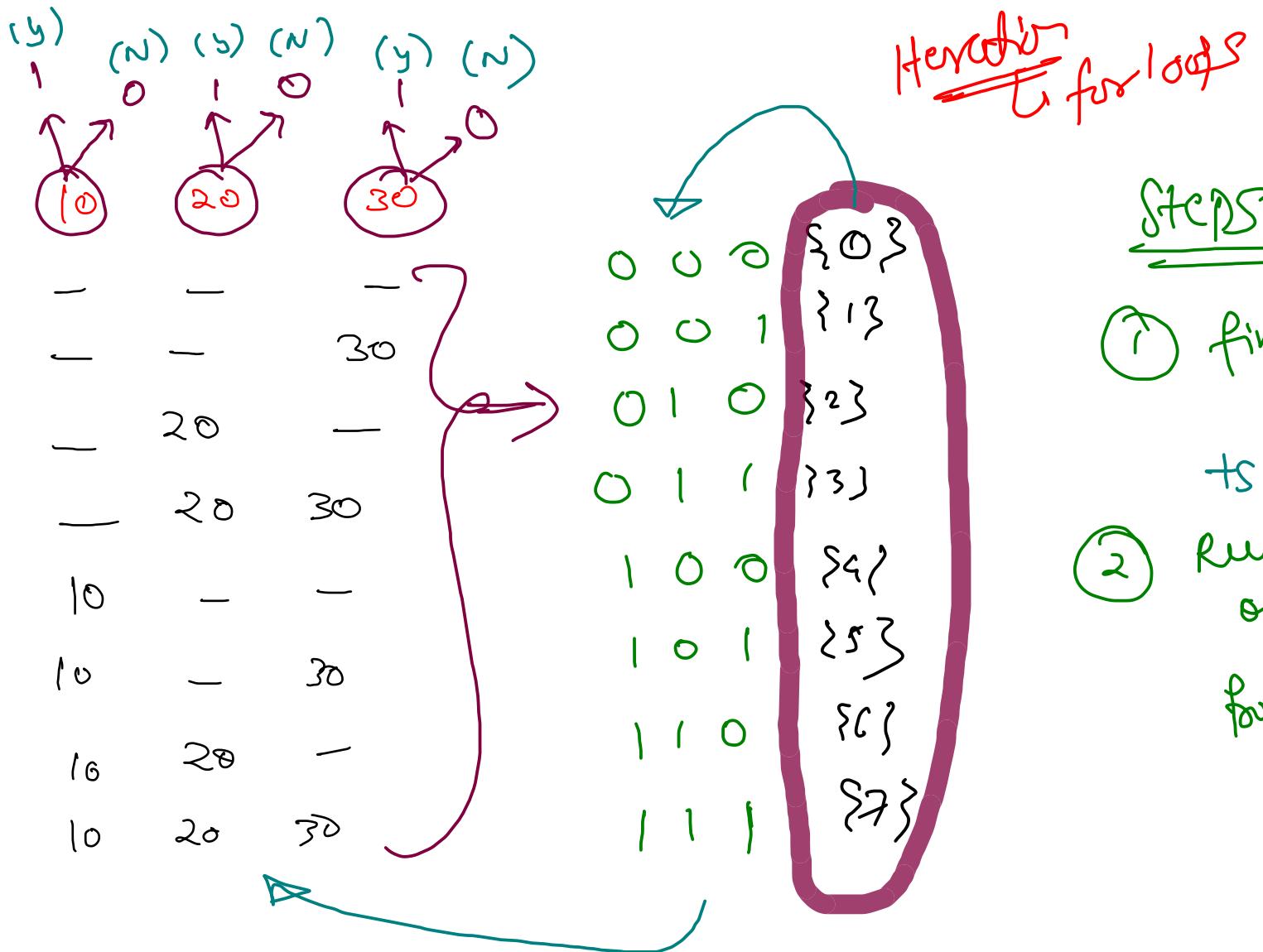
10       $s=0, e=0$   
 10      20       $s=0, e=1$   
 10      20      30       $s=0, e=2$   
 10      20      30      40       $s=0, e=3$   
 10      20  
 20  
 20      30  
 20      30      40  
 20  
 30  
 30      40  
 30  
 40  
 40

```

10      20      30      40

for(int st=0; st<n; st++){
  for(int end=st; end<n; end++){
    for(int i=st; i<=end; i++) {
      System.out.print(arr[i] + "\t");
    }
    System.out.println();
  }
}
  
```

Subsets



~~Iteration for loops~~

Steps

- ① find total Subsets  
 $TS = 2^n$
- ② Run a loop on decimal no from 0 to  $TS-1$

(binary / power)  $\times 10$

$$(011010 / 10^5) = 0 \cdot 1 \cdot 1 \cdot 0 \cdot 0 = 0$$

$$(011010 / 10^4) = 01 \cdot 1 \cdot 0 = 1$$

$$(011010 / 10^3) = 011 \cdot 1 \cdot 0 = 1$$

$$(011010 / 10^2) = 0110 \cdot 1 \cdot 0 = 0$$

$$(011010 / 10^1) = 01101 \cdot 1 \cdot 0 = 1$$

$$(011010 / 10^0) = 011010 \cdot 1 \cdot 0 = 0$$

$$(011010 / 10^{-1}) = 011010 \cdot 1 \cdot 1 = 1$$

$$(011010 / 10^{-2}) = 011010 \cdot 1 \cdot 1 = 1$$

$$(011010 / 10^{-3}) = 011010 \cdot 1 \cdot 1 = 1$$

$$(011010 / 10^{-4}) = 011010 \cdot 1 \cdot 1 = 1$$

$$(011010 / 10^{-5}) = 011010 \cdot 1 \cdot 1 = 1$$

$$(011010 / 10^{-6}) = 011010 \cdot 1 \cdot 1 = 1$$

③ Convert decimal to its binary equivalent

```

int totalSubsets = (int)Math.pow(2, n);

for(int dec = 0; dec < totalSubsets; dec++){

    int binaryNo = decimalToBinary(dec);

    int power = (int)Math.pow(10, n - 1);
    // convert binaryNo to equivalent subset
    for(int i=0; i<n; i++){

        int bit = (binaryNo / power) % 10;

        if(bit == 1){
            System.out.print(arr[i] + "\t");
        } else {
            System.out.print("-\t");
        }

        power /= 10;
    }
    System.out.println();
}

```

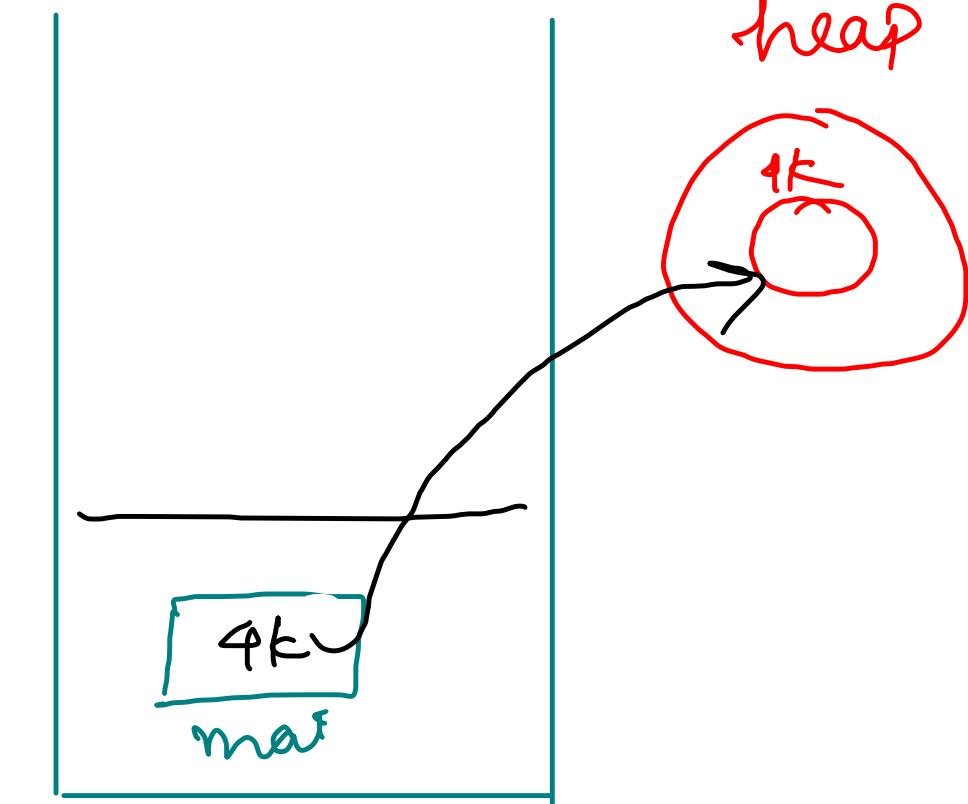
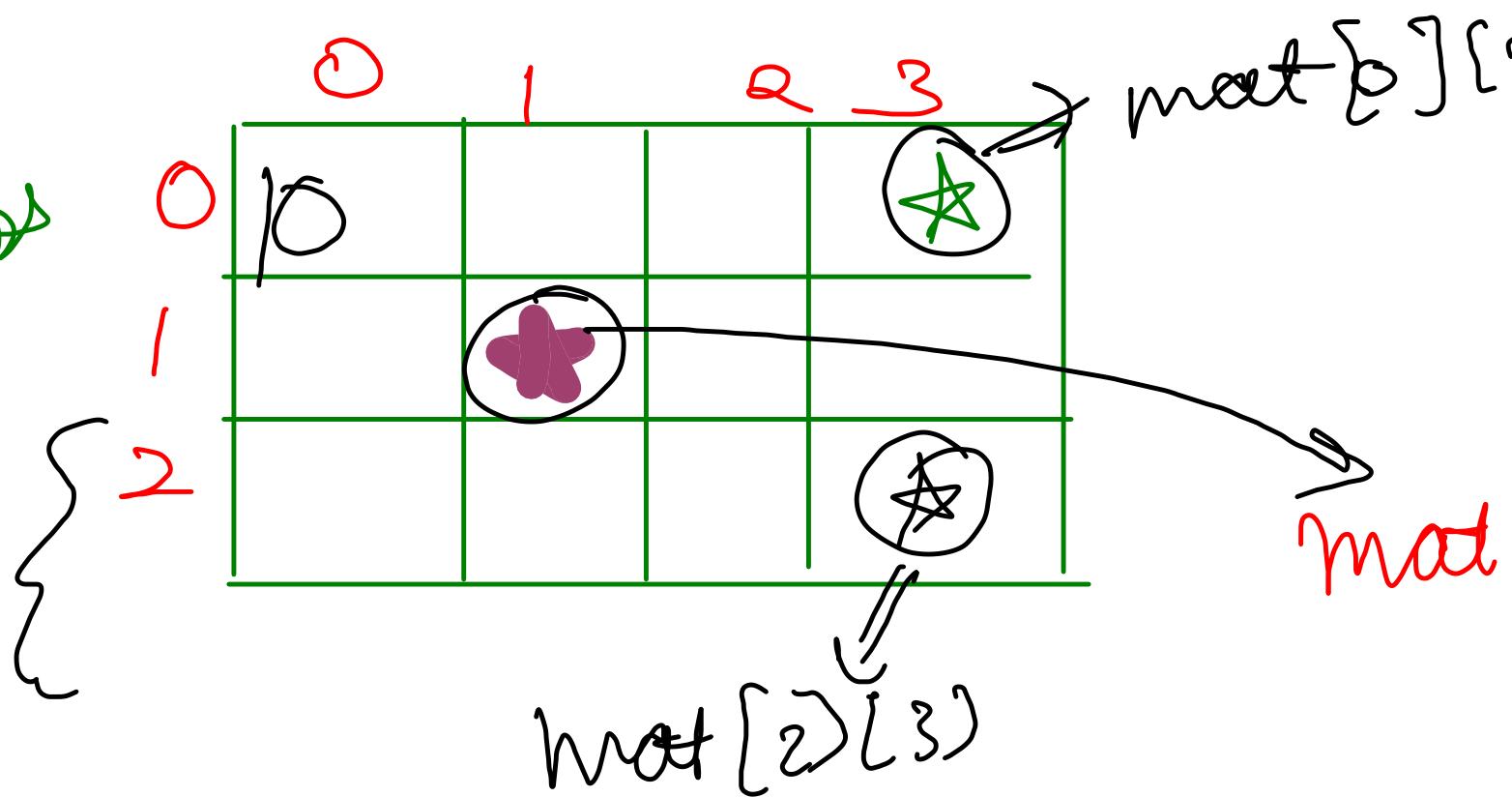
		10	20	30
0	000	—	—	—
1	001	—	—	30
2	010	—	20	—
3	011	—	20	30
4	100	10	—	—
5	101	10	—	30
6	110	10	20	—
7	111	10	20	30

2D arrays / matrix  
↓  
row column

`int [][] mat = new int [3][4];`

random access

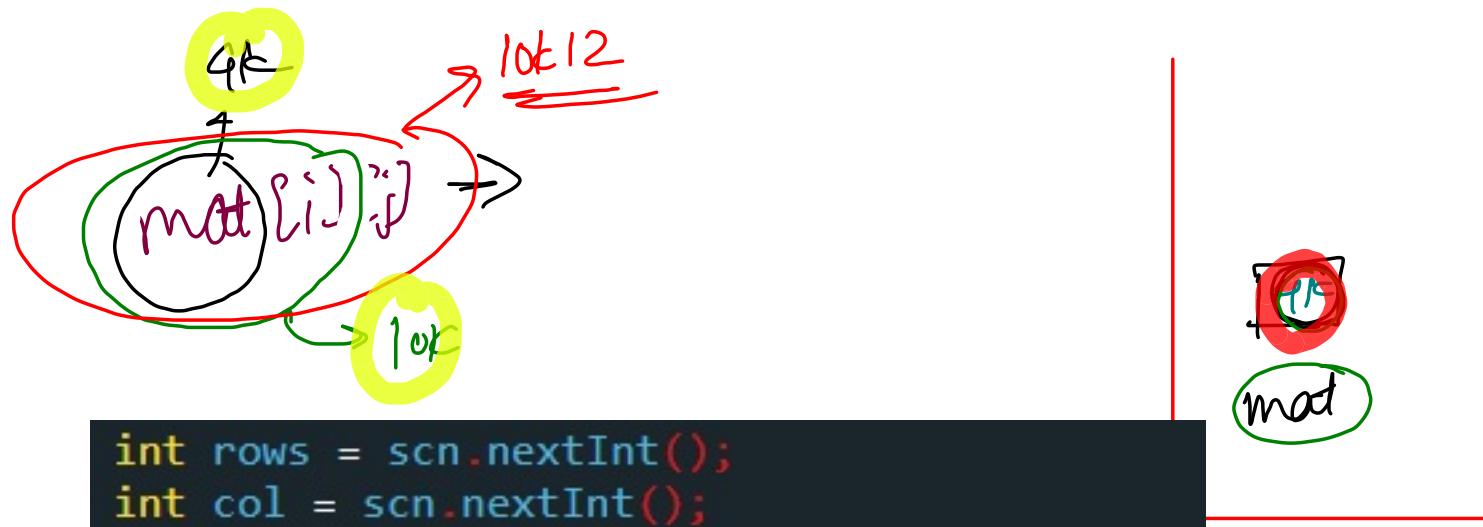
logical view



$$\underline{\underline{mat[2][3]}} = 10$$

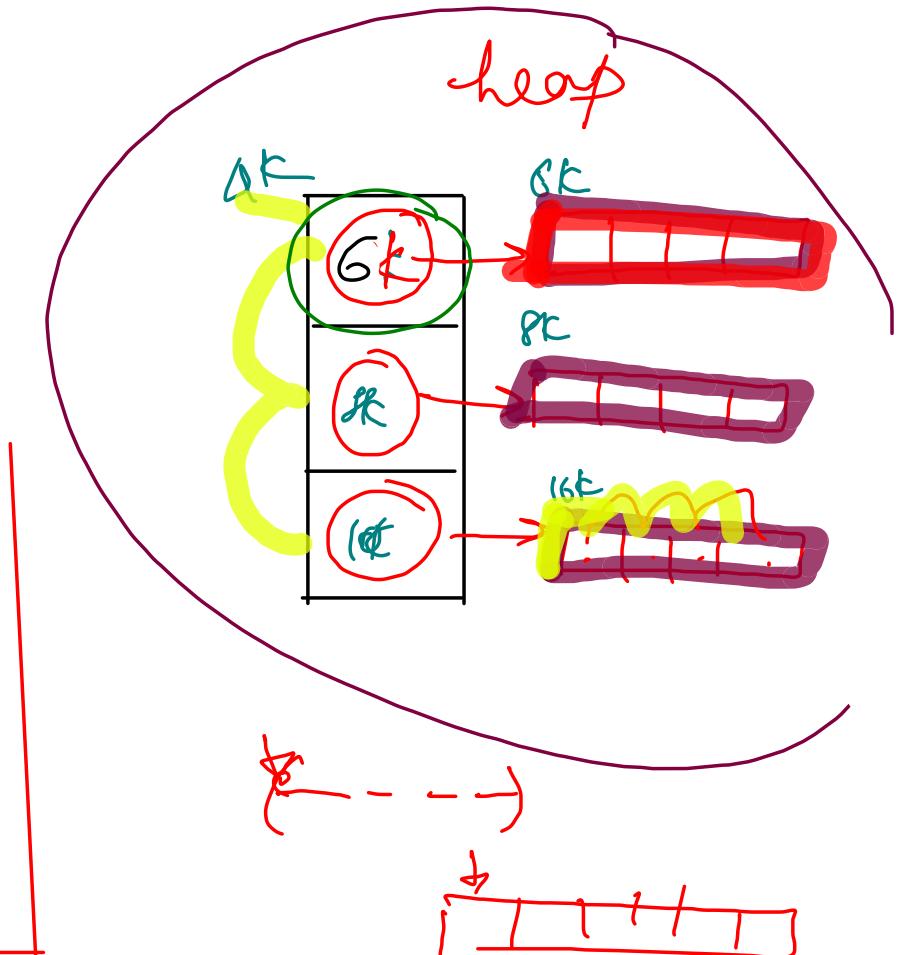
## Memory Allocation

```
int [][] mat = new int [3][4];
```

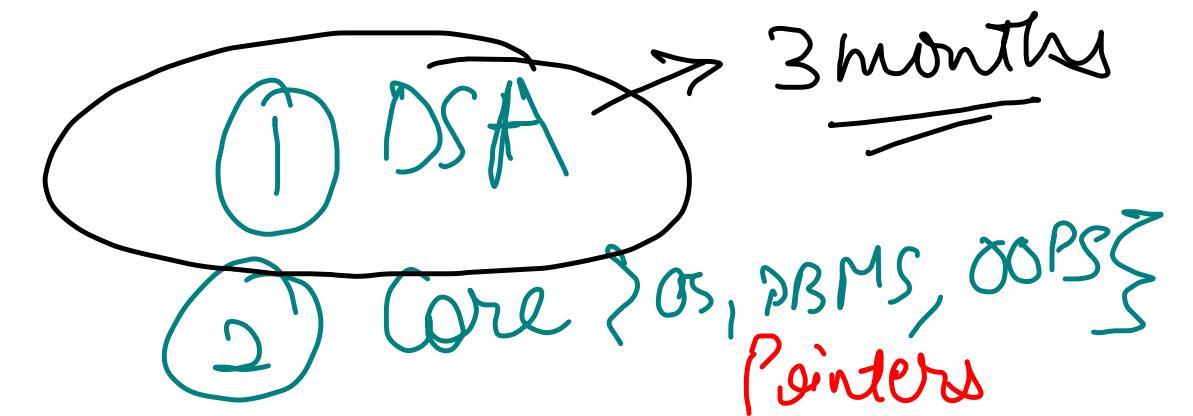


```
int rows = scn.nextInt();
int col = scn.nextInt();
```

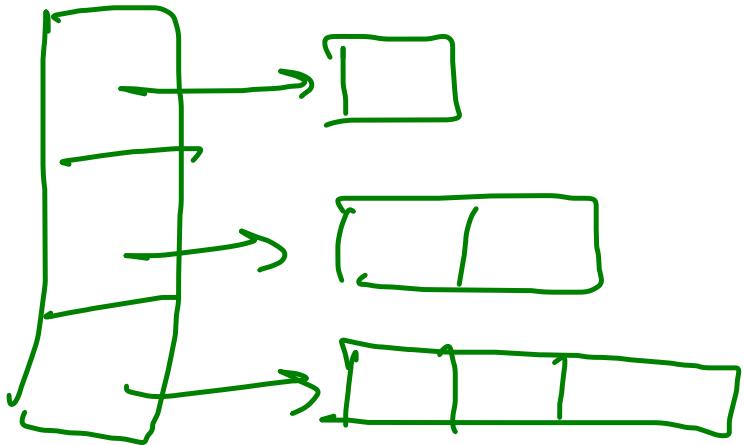
```
int [][] mat = new int [rows][col];
for(int i=0; i<rows; i++){
    for(int j=0; j<col; j++){
        mat[i][j] = scn.nextInt();
    }
}
for(int i=0; i<rows; i++){
    for(int j=0; j<col; j++){
        System.out.print(mat[i][j] + " ");
    }
    System.out.println();
}
```



## Stack & Queue



```
int [3][3] mat = new int[3];  
for( int i=0; i<3; i++ ) {  
    mat[i] = new int[ i+1 ];  
}
```



## Today's Questions

- ① State of Wakanda - I { zig-zag Order }
- ② State of Wakanda - II { Diagonal Order }
- ③ spiral Display
- ④ End Point

wave Traversal

## To morrow's Questions

- ① Matrix Multiplication
- ② Rotate by  $90^\circ$

- ③ Ring Rotate
- ④ Saddle Rice

DSA

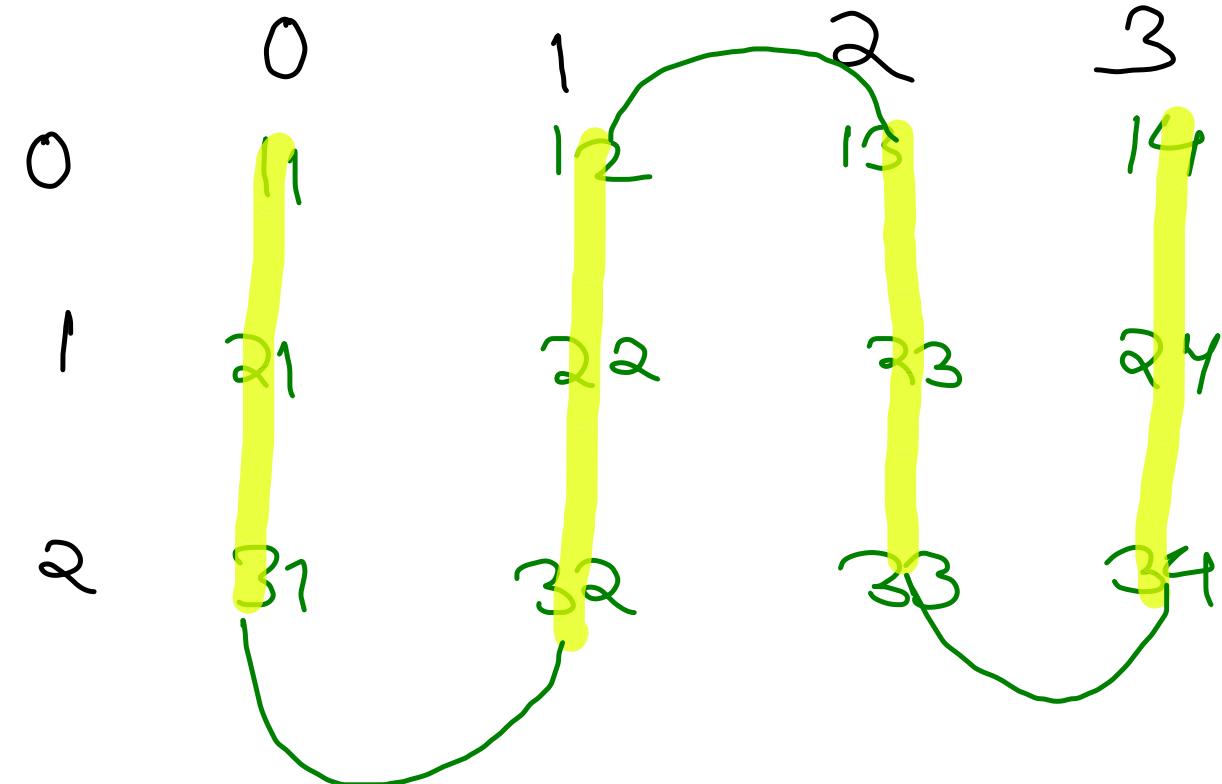
- ① Level 1 :  $\rightarrow$  3 months
- ② Level 2 :  $\rightarrow$  4 months
- ③ Level 3 : - 1 month

9 - 11:30  
2.5 hrs  
M, T, W, F

# State of Wakanda - 1

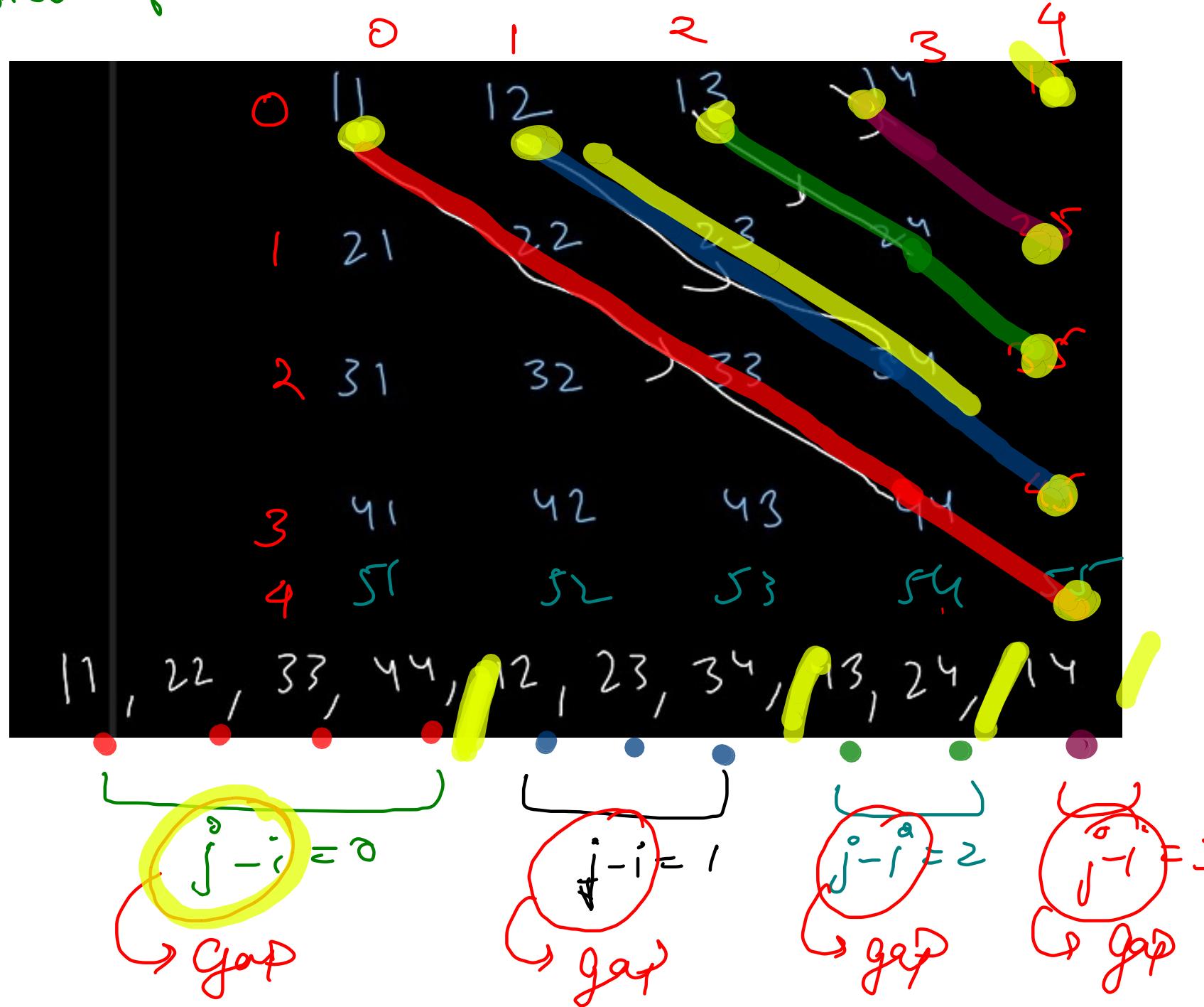
11 21 31 32 22 12

13 23 33 34 24 14

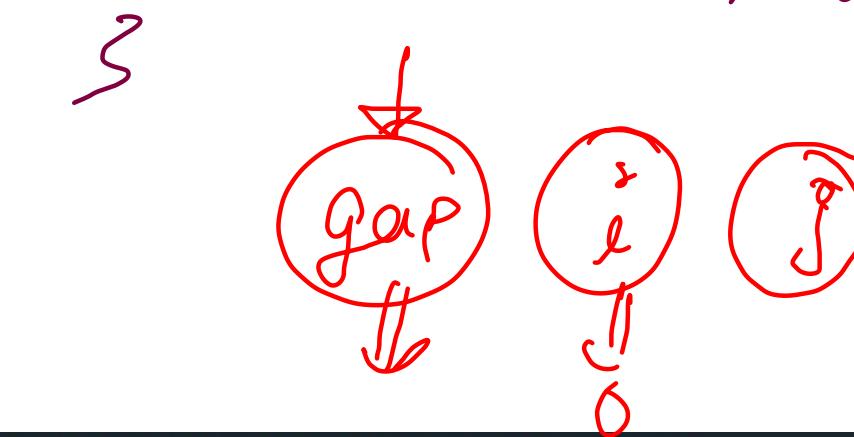


```
public static void waveTraversal(int[][] mat){  
    int n = mat.length; // rows  
    int m = mat[0].length;  
  
    // column by column print  
    for(int j=0; j<m; j++){  
        if(j % 2 == 0){  
            for(int i=0; i<n; i++){  
                System.out.println(mat[i][j]);  
            }  
        } else {  
            for(int i=n-1; i>=0; i--){  
                System.out.println(mat[i][j]);  
            }  
        }  
    }  
}
```

# state of Wakanda-II



diagonals  
cols.  
first gap = 0; gap < m; gap++;  
for (inner)  $\rightarrow i, j, i+j$



```
// outer loop -> diagonals -> gap
for(int gap = 0; gap < mat[0].length; gap++){
    // traversing the diagonal
    // gap -> j - i
    for(int i = 0, j = gap; j < mat[0].length; i++, j++){
        System.out.println(mat[i][j]);
    }
}
```

gap strategy

Phase - I

Algo ✗

Code ✗

Phase - II

Algo ✓

Code ✗

Phase - III

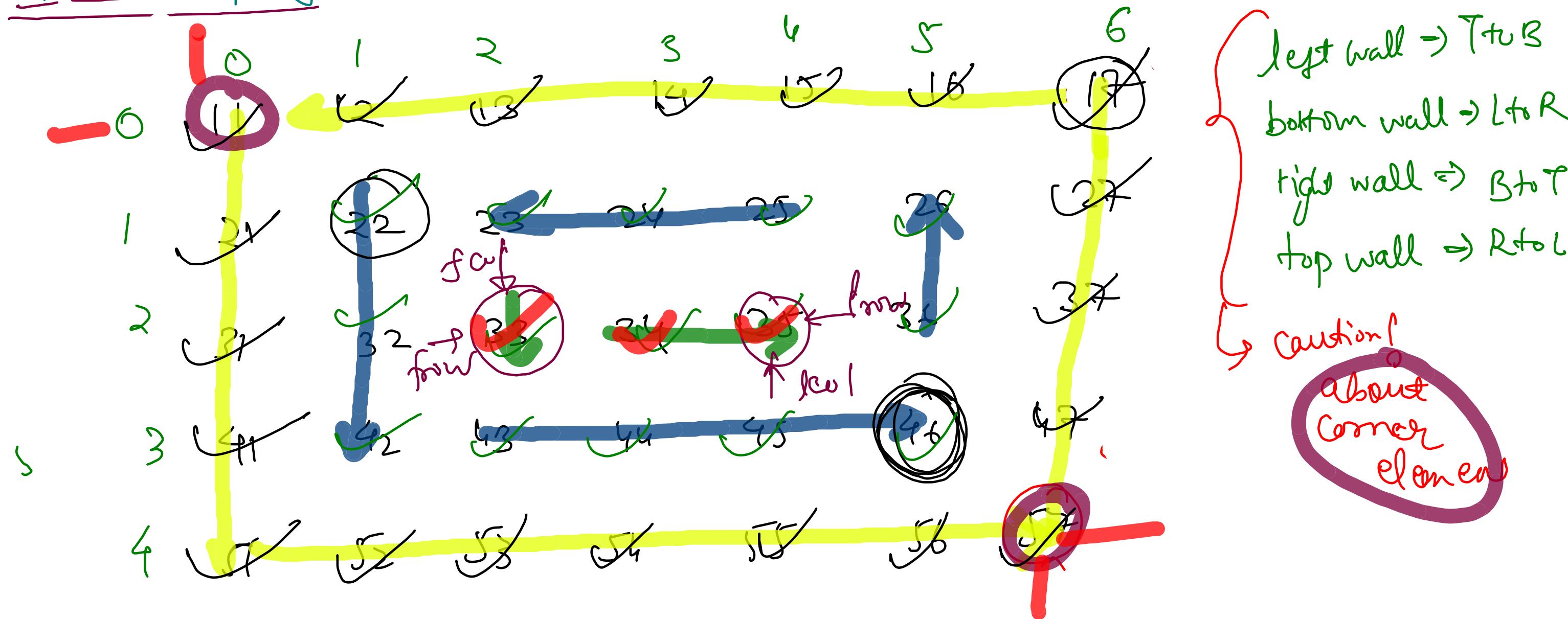
Algo ✗

Code ✓

Phase IV

Algo ✓  
Code ✓

## Spiral Display

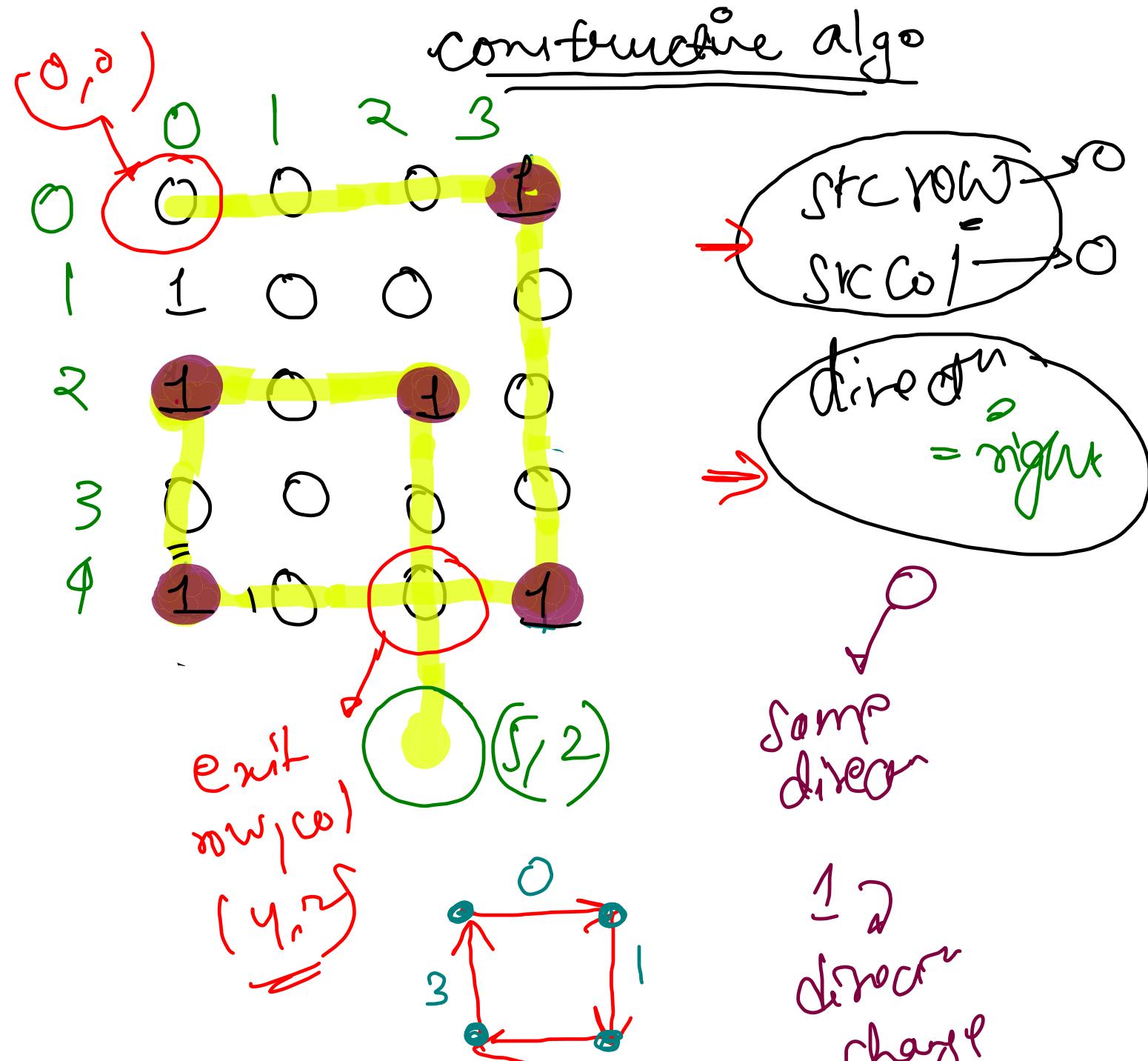
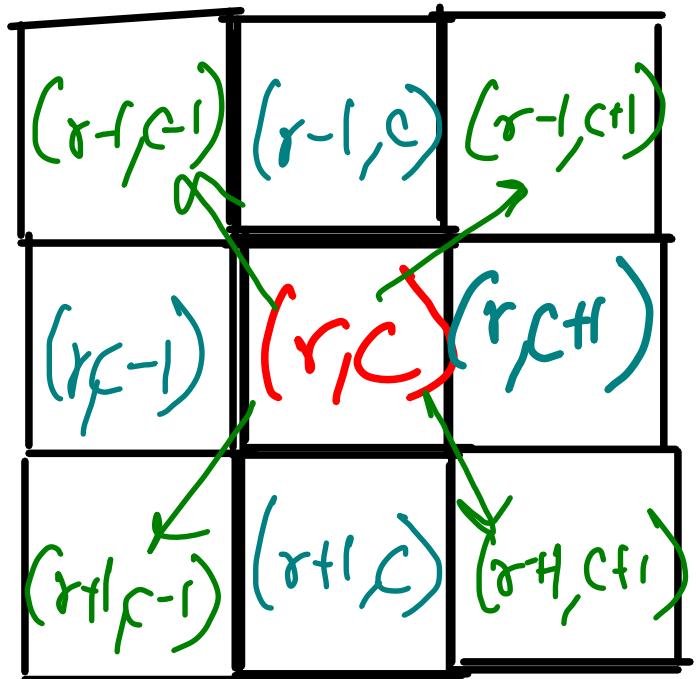


Exit Point

{ binary matrix }

# there is always

a  
exit point



```
int currRow = 0, currCol = 0;
int prevRow = 0, prevCol = 0;

// directions
// right -> 0, down -> 1, left -> 2, top -> 3
int direction = 0;

// left wall se exit, top wall se exit, bottom wall se exit, right wall se exit
while((currCol >= 0) && (currRow >= 0) && (currRow < n) && (currCol < m)) {

    if(mat[currRow][currCol] == 1){
        // change direction -> 90 degree clockwise
        direction = (direction + 1) % 4;
    }

    prevRow = currRow;
    prevCol = currCol;

    if(direction == 0){
        // right
        currCol++;
    } else if(direction == 1){
        // down
        currRow++;
    } else if(direction == 2){
        // left
        currCol--;
    } else {
        // top
        currRow--;
    }
}

System.out.println(prevRow);
System.out.println(prevCol);
```

Press Esc to exit full screen



## Today's Questions

- ① Matrix Multiplication
- ② Rotate by  $90^\circ$

- ③ Ring Rotate
- ④ Saddle Rice

Rotate by  $90^\circ$

	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34
3	41	42	43	44

$90^\circ$   
clockwise

	0	1	2	3
0	41	31	21	11
1	42	32	22	12
2	43	33	23	13
3	44	34	24	14

In-place

```
public static void swapColumns(int[][] arr){  
    int leftCol = 0, rightCol = arr[0].length - 1;  
    while(leftCol < rightCol){  
        for(int i=0; i<arr.length; i++){  
            swap(arr, i, leftCol, rightCol);  
        }  
        leftCol++; rightCol--;  
    }  
}
```

Transpose

$a[i][j] \Leftrightarrow$

0	11	1	21
1	12	2	22
2	13	3	23
3	14	4	24

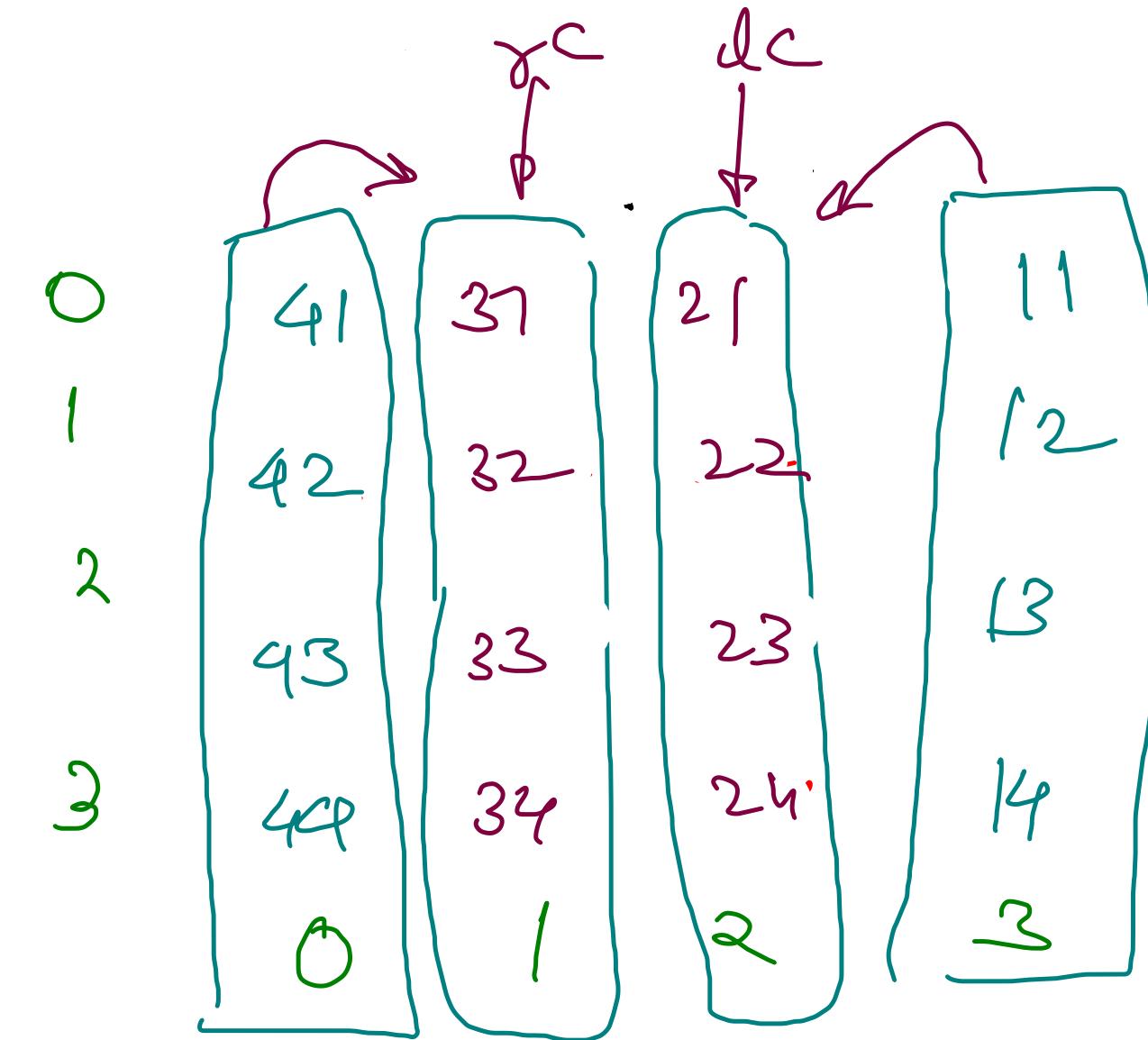
$a[j][i]$

0	31	1	21
1	32	2	22
2	33	3	23
3	34	4	24

swap columns [OR] Reverse rows

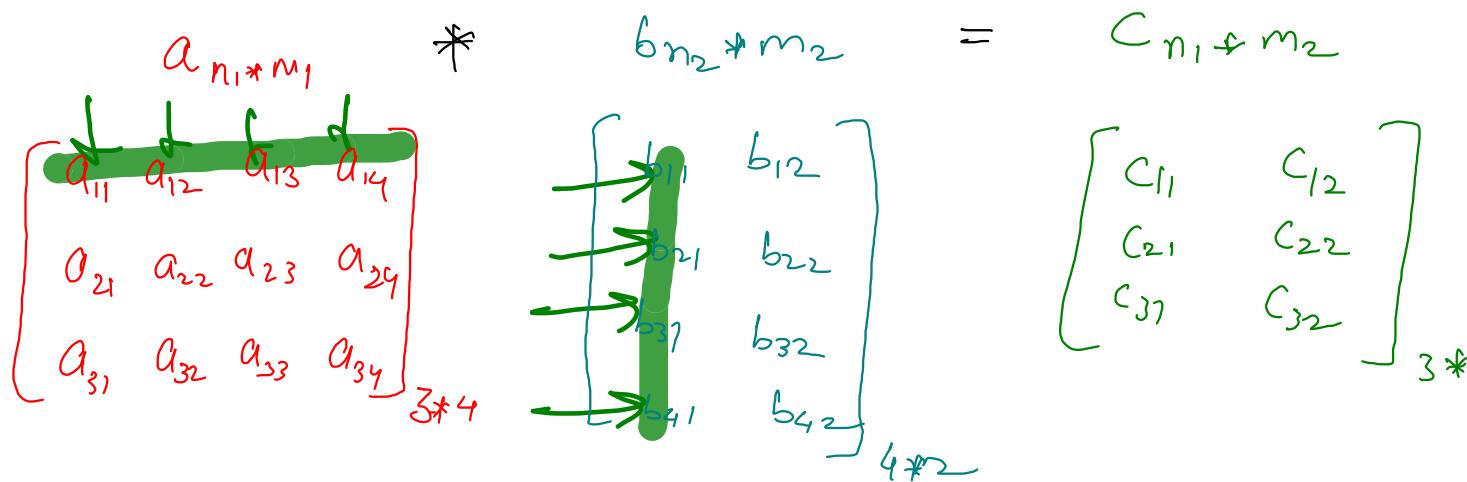
```
public static void transpose(int[][] arr){  
    for(int i=0; i<arr.length; i++){  
        for(int j = i + 1; j < arr[0].length; j++){  
            swap(arr, i, j);  
        }  
    }  
}
```

```
public static void swapColumns(int[][] arr){  
    int leftCol = 0, rightCol = arr[0].length - 1;  
    while(leftCol < rightCol){  
        for(int i=0; i<arr.length; i++){  
            swap(arr, i, leftCol, rightCol);  
        }  
        leftCol++; rightCol--;  
    }  
}
```



## # Matrix Multiplication

$$m_1 = n_2$$



```
int[][] res = new int[n1][m2];
```

```
for(int i=0; i<res.length; i++){
    for(int j=0; j<res[0].length; j++){
        for(int k=0; k<m1; k++) { // m1 == n2
            res[i][j] += (mat1[i][k] * mat2[k][j]);
        }
    }
}
```

$$\left\{ \begin{array}{l} c_{11} = a_{11} * b_{11} + a_{12} * b_{21} + a_{13} * b_{31} + a_{14} * b_{41} \\ c_{12} = a_{11} * b_{12} + a_{12} * b_{22} + a_{13} * b_{32} + a_{14} * b_{42} \\ c_{21} = a_{21} * b_{11} + a_{22} * b_{21} + a_{23} * b_{31} + a_{24} * b_{41} \\ c_{22} = a_{21} * b_{12} \quad a_{22} * b_{22} \quad a_{23} * b_{32} \quad a_{24} * b_{42} \end{array} \right.$$

```
if(m1 != n2){
    System.out.println("Invalid input");
    return;
}
```

$$\begin{aligned} i=0, j=0, & \quad r = p/r \neq 3 \\ & \quad \text{true}[0][0] \end{aligned}$$

~~$$C_{i,j} = \sum_{k=0}^{m_1} a_{i,k} * b_{k,j}$$~~

$$\begin{aligned} c_{11} &= a_{11} * b_{11} \\ &+ a_{12} * b_{21} \\ &+ a_{13} * b_{31} \\ &+ a_{14} * b_{41} \end{aligned}$$

level ①

breadth of  
knowledge

level ②

depth of  
knowledge

level ③

Problem solving

Saddle Point

	0	1	2	3	# All elements are unique/distinct
0	34	42	30	59	
1	50	85	44	97	# Saddle point - minimum in the row
2	42	16	25	36	maximum in the column
3	15	21	05	50	

① 0 saddle point (Y/N)

② >1 saddle point (Y/N)

(44)

Example  
Proof

```

for(int i=0; i<mat.length; i++){
    // finding minimum of ith row
    int minCol = 0;
    for(int j=0; j<mat[0].length; j++){
        if(mat[i][j] < mat[i][minCol]){
            minCol = j;
        }
    }

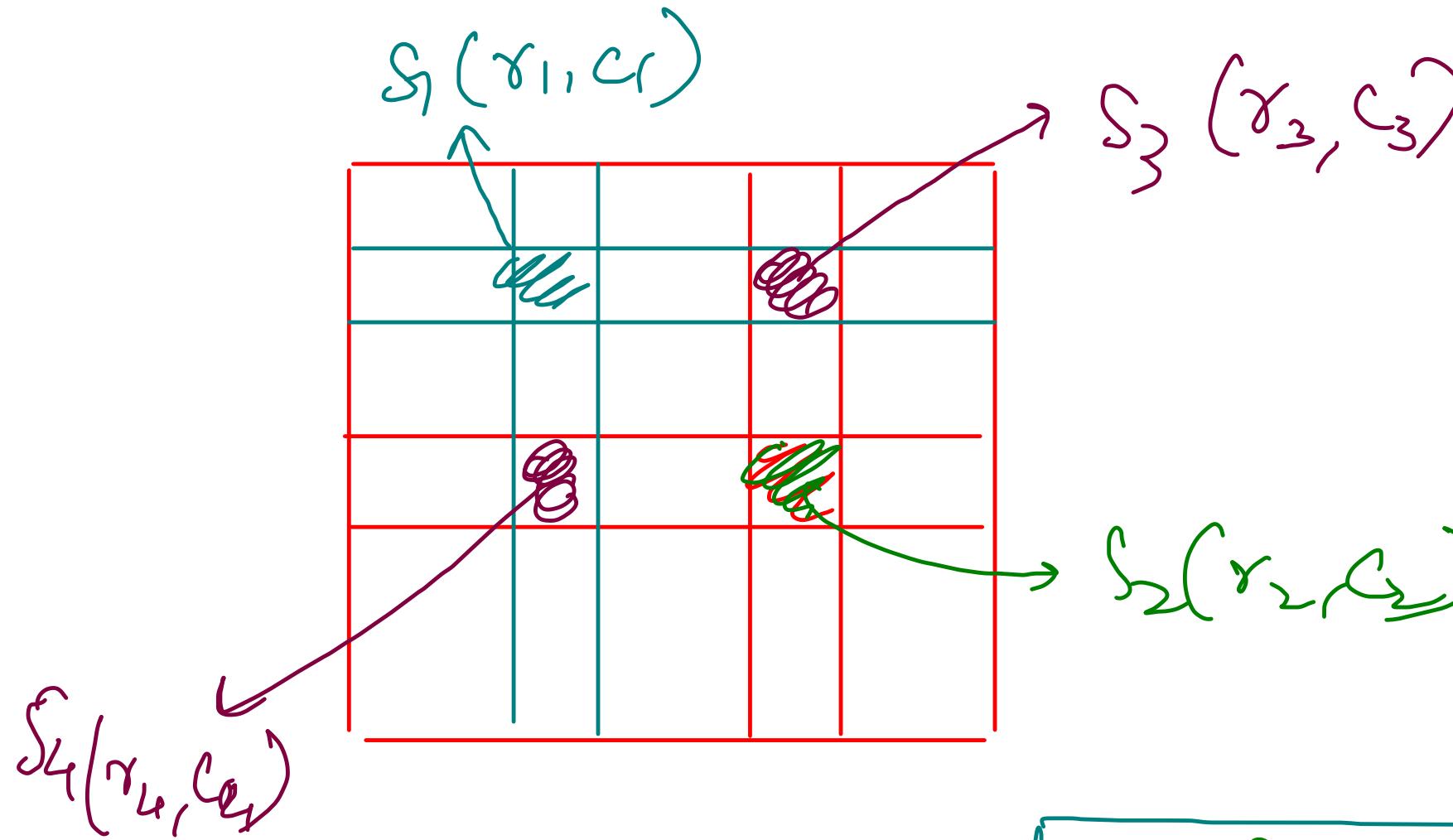
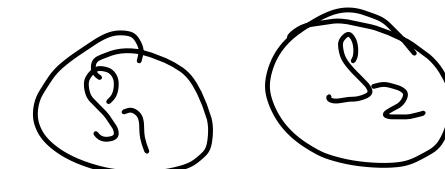
    int maxRow = 0;
    for(int k=0; k<mat.length; k++){
        if(mat[k][minCol] > mat[maxRow][minCol]){
            maxRow = k;
        }
    }

    if(maxRow == i){ // min in Row -> minCol is also true
        System.out.println(mat[maxRow][minCol]);
        return;
    }
}

```

Assumption

② saddle point



$$\left. \begin{array}{l} ①^o S_1 < S_3 \\ ②^o S_1 > S_4 \end{array} \right\} S_1$$

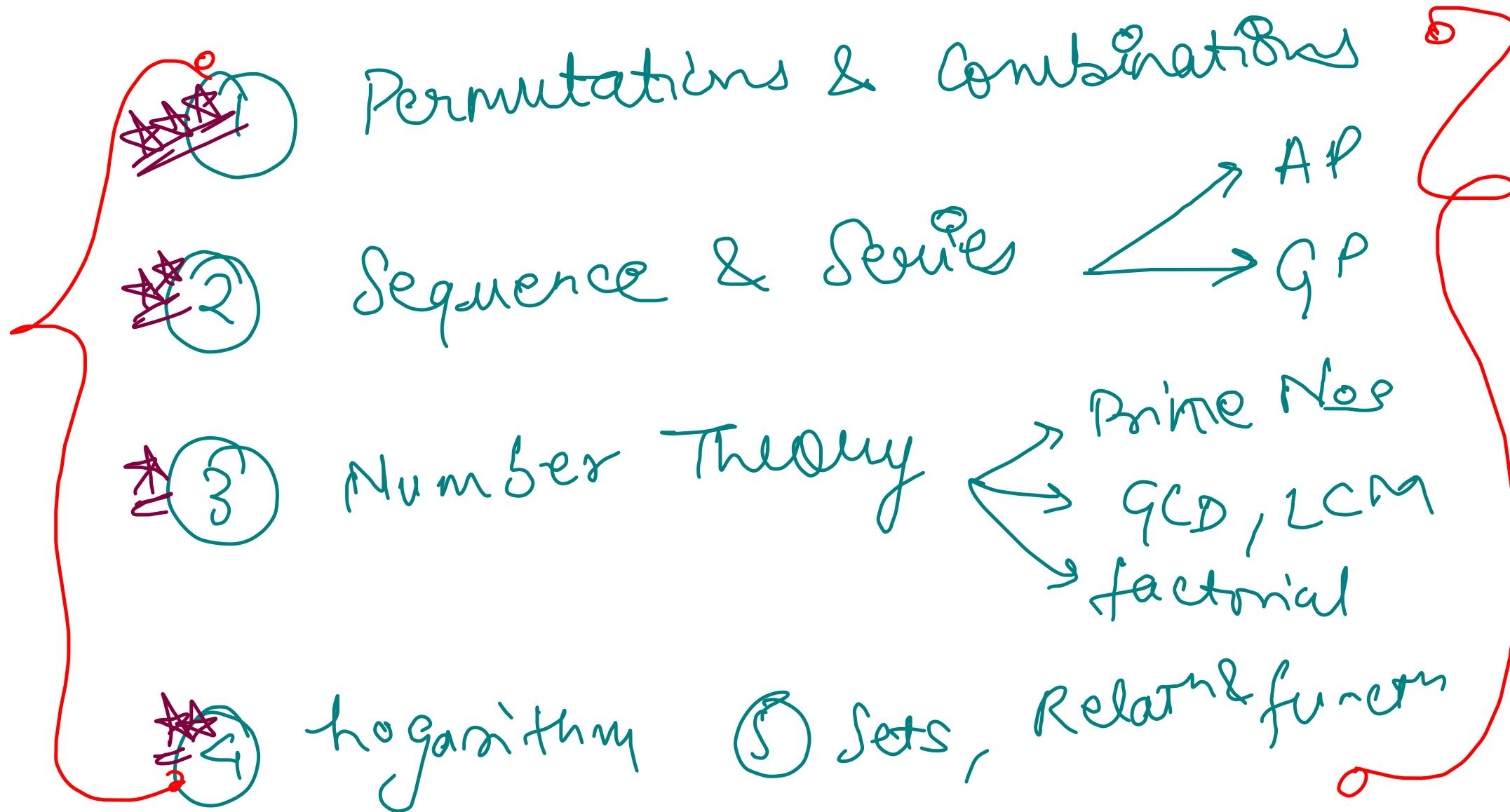
$$\left. \begin{array}{l} ③^o S_2 < S_4 \\ ④^o S_2 > S_3 \end{array} \right\} S_2$$

$$\boxed{S_1 < S_3 < S_2 \Rightarrow S_1 < S_2 \rightarrow \text{assumption wrong}}$$

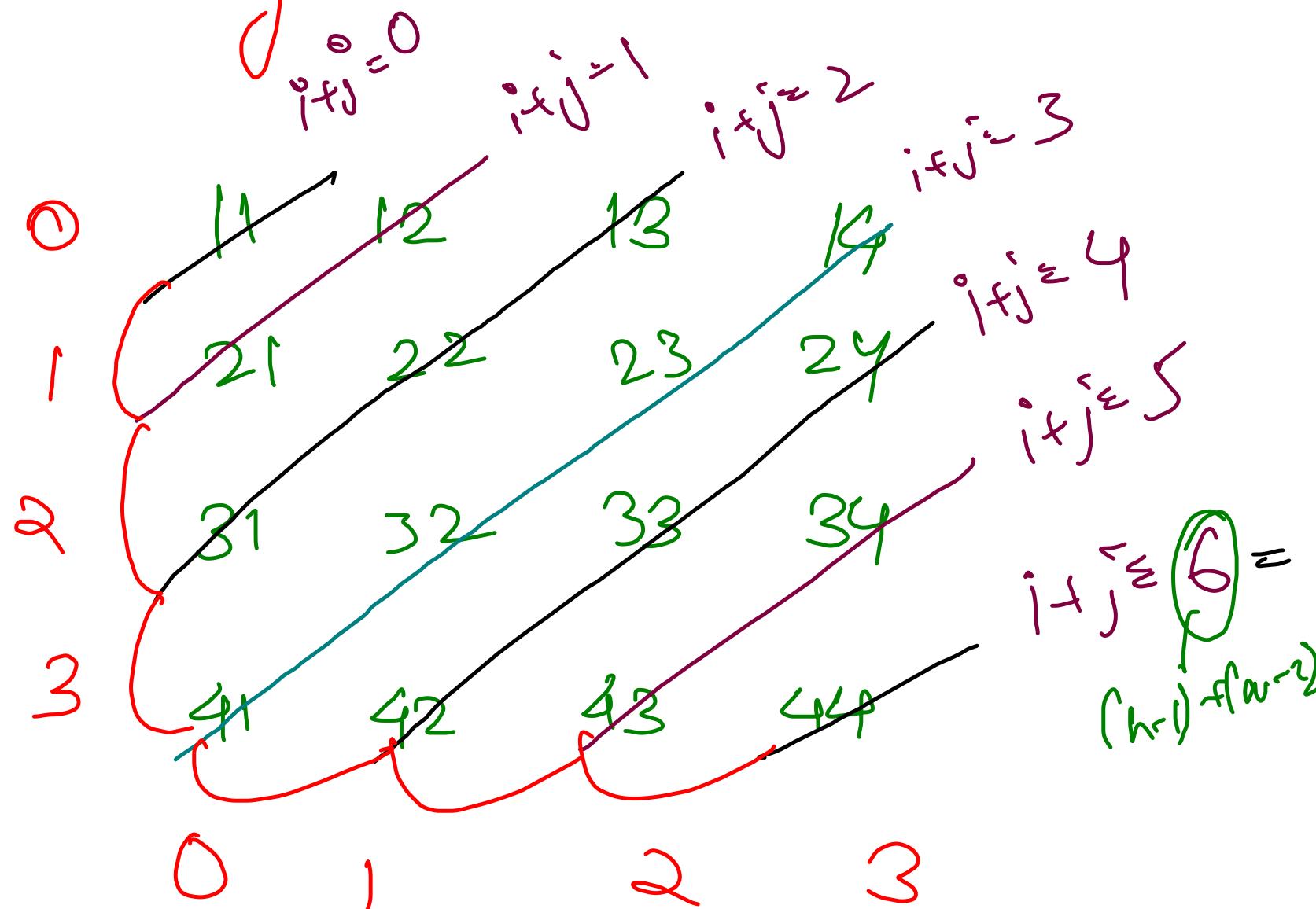
$$\boxed{S_1 > S_4 > S_2 \Rightarrow S_1 > S_2}$$

There will be at most 1 saddle point -

# Maths required for DSA



# Diagonal Traversal - II



```

for(int sum = 0; sum <= (n - 1) + (m-1); sum++){
    int i = 0, j = 0;
    if(sum <= (n - 1 + m - 1) / 2){
        // lower triangle
        j = 0;
        i = sum - j;
    } else {
        i = n - 1;
        j = sum - i;
    }

    for(; i >= 0 && j < m; i--, j++){
        System.out.print(mat[i][j] + " ");
    }
    System.out.println();
}
    
```

String

→ Array of characters

- ↳ lowercase { 'a', 'b', 'c', 'd', ... }
- ↳ uppercase { 'A', 'B', 'C', ... }
- ↳ digits { '0', '1', '2', '3', ... }
- ↳ null character { '\0' }



## ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	©	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	.	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	-	127	7F	[DEL]

length()

{ 'h', 'e', 'l', 'l', 'o' }  
 0 1 2 3 4  
 ↓

"hello"

65 - 90 (A - Z)

97 - 122 (a - z)

48 - 57 (0 - 9)

$$'A' - 'a' = -32$$

$$'a' - 'A' = 32$$

$$A \rightarrow a$$

$$a \rightarrow A$$

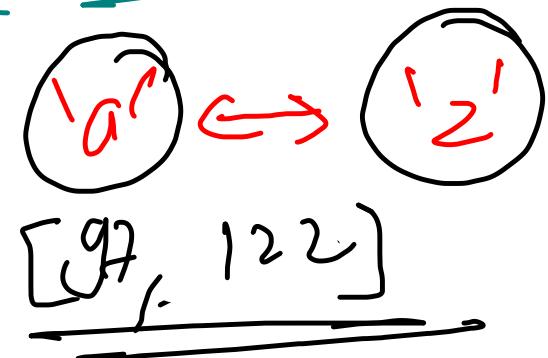
$$b \rightarrow B$$

$$B \rightarrow b$$

pepCODiG

for(int i=0) i < ? j++ {

if(iflowerCase)



toUpperCase()

$$'p' - 'p' = 'a' - 'A'$$

$$'p' = 'p' - 'a' + 'A'$$

$$'p' = 'p' - 97 + 65$$

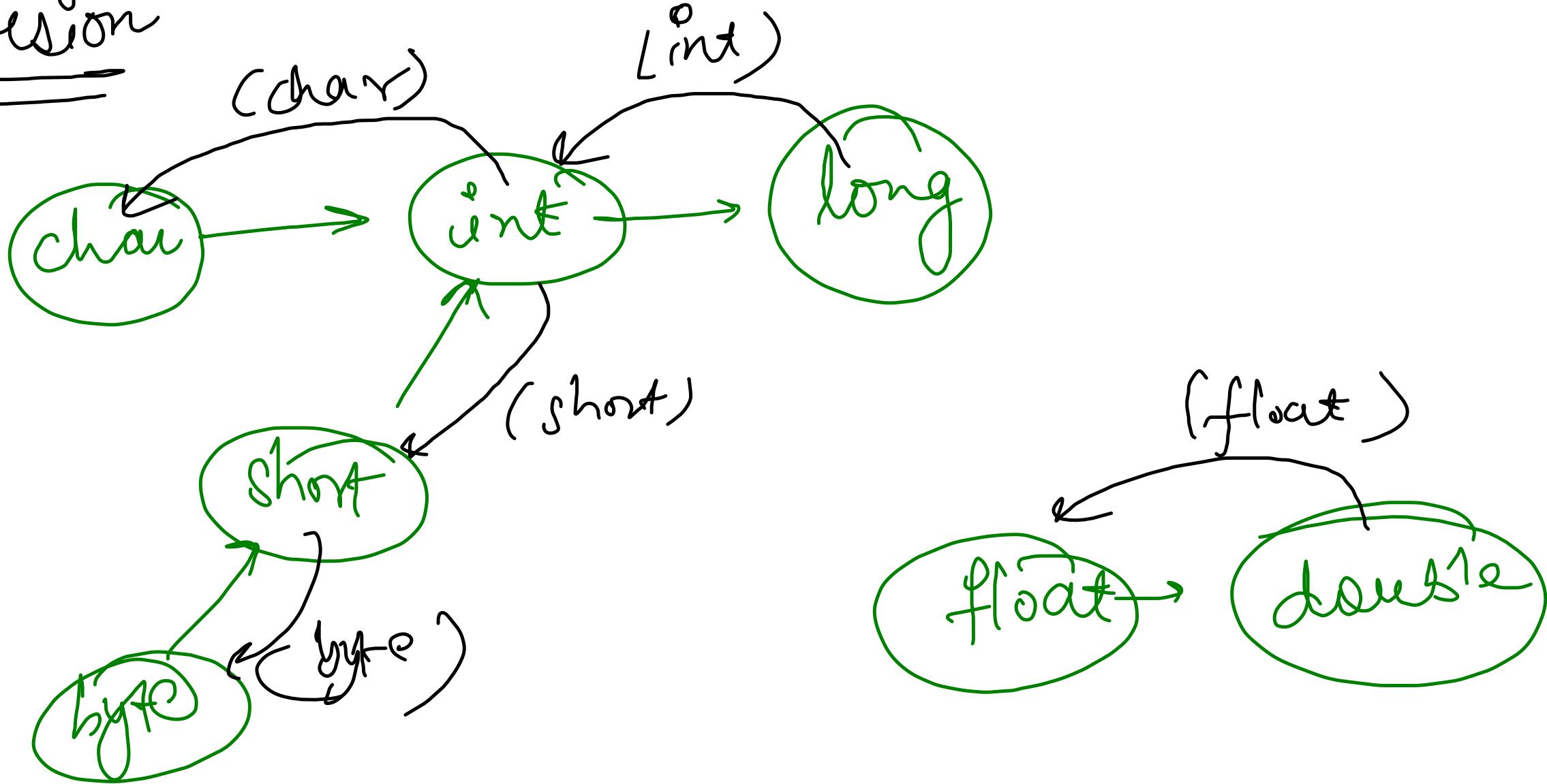
$$'p' = 'p' - 32$$

else

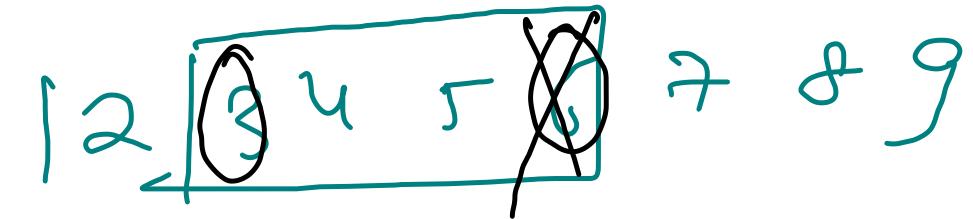
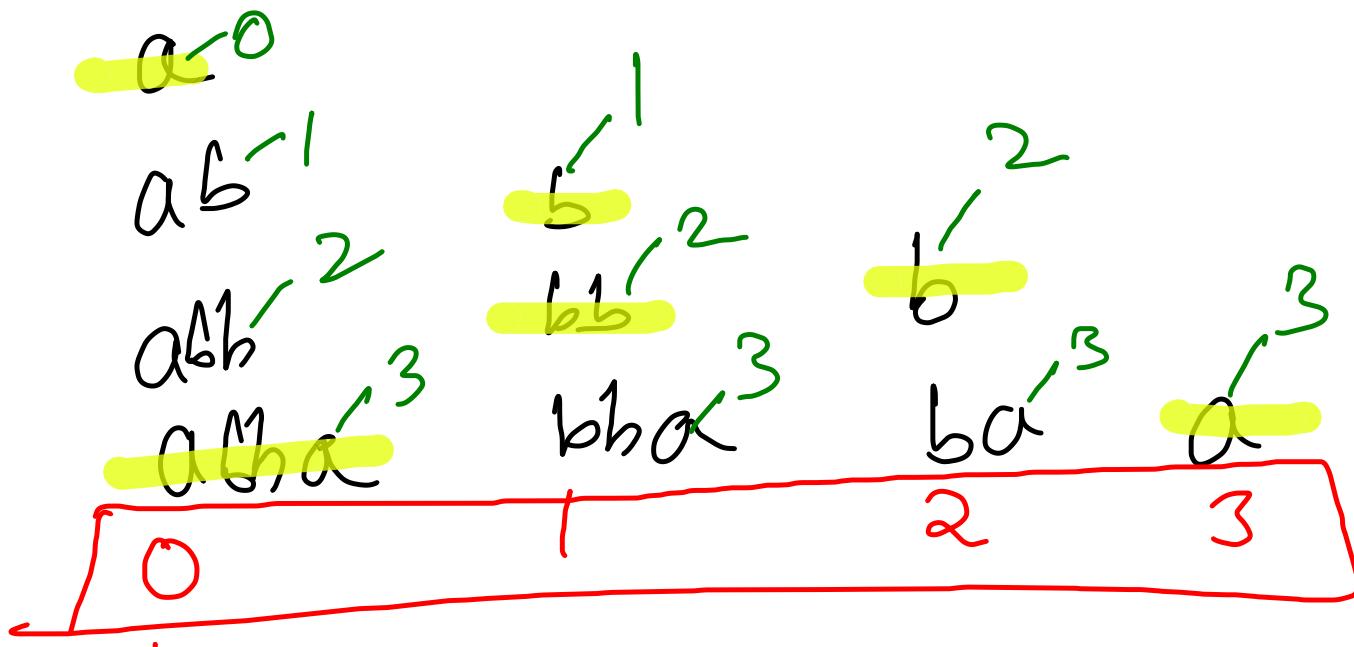
toLowercase()

}

## Type conversion



"abba"  
0 1 2 3



6 ->

{ l, r )  
r - l

Str. substring( startingIndex , endingIndex )

## Strings

### Memory Mapping

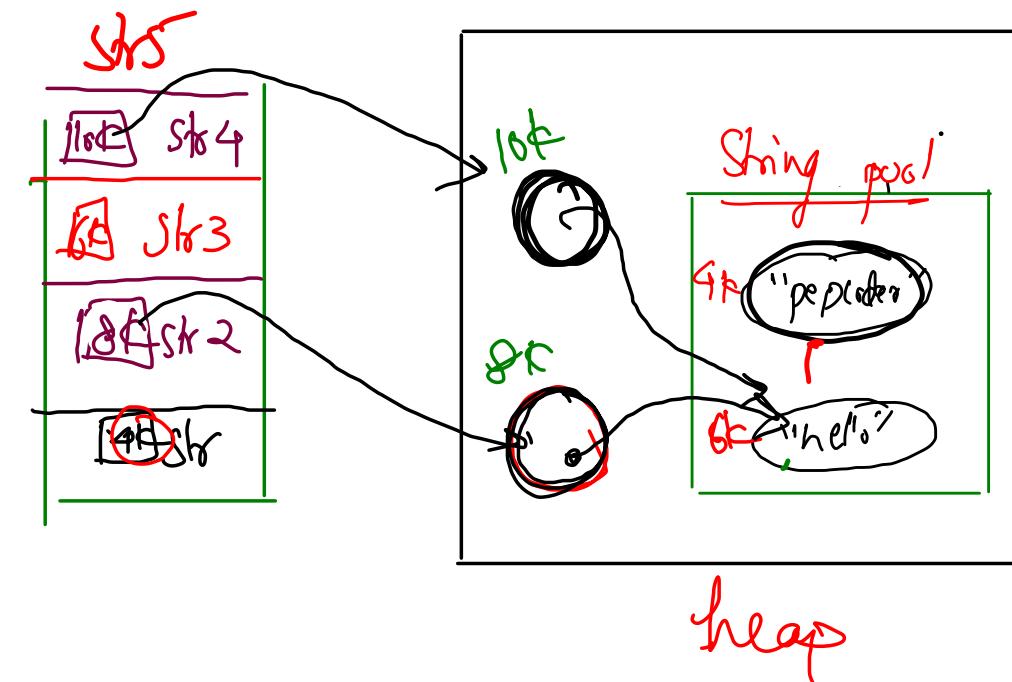
① String str = "peacock";

② String str2 = new String("hello");

③ str3 = "hello"

④ str4 = new String("hello");

⑤ str5 = "hefli";



- ① length()
- ② substring()
- ③ charAt()

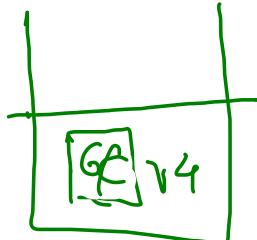
- 1 str2 = str3 (F)
  - 2 str3 = str5 (T)
  - 3 str2 == str4 (F)
  - 4 str2.equals(str3) (T)
  - 5 str3.equals(str5) (T)
  - 6 hello (=> hefli)
  - 7 str2.equals(str4) (T)
  - 8 str.equals(str2) (F)
- Toggle case  
↓  
ASCII code

## String Interning

Advantage → Memory wastage  
Implications

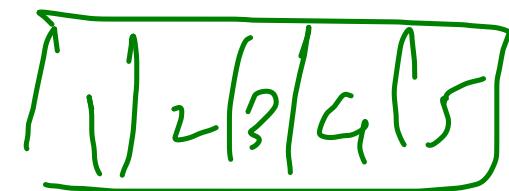
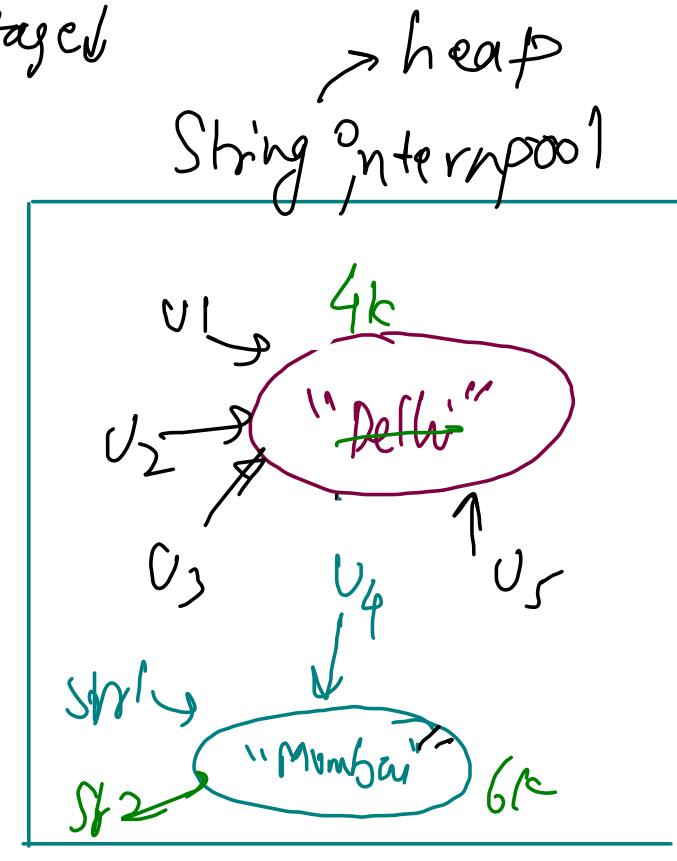
String str = "Mumbai";

String str2 = "Mumbai";



① v4 = "Hello"

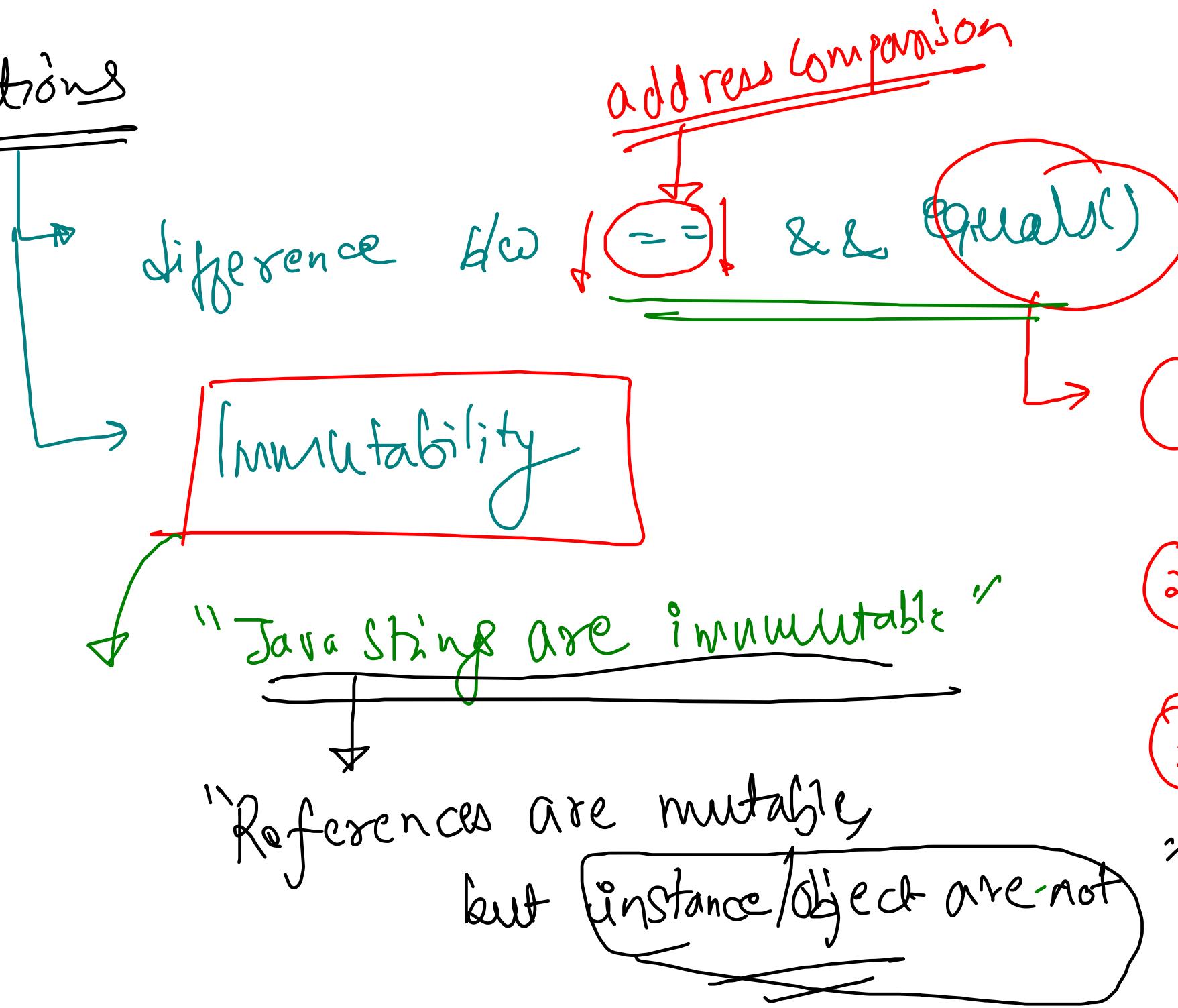
v1 = "Mumba"



arr[2] = 30

~~String.charAt(2) = 'H'~~  
→ setchar X

## Implications



String  $s = "0";$

for (int  $i=1; i < n; i++)$

$s = s + i;$

Concatenation

① Distracted → Social Media  
Nodes

② College

1 + 2 + 3

+ ... 1000

$$= \frac{1000 * 1001}{2} = \frac{n * n + 1}{2}$$

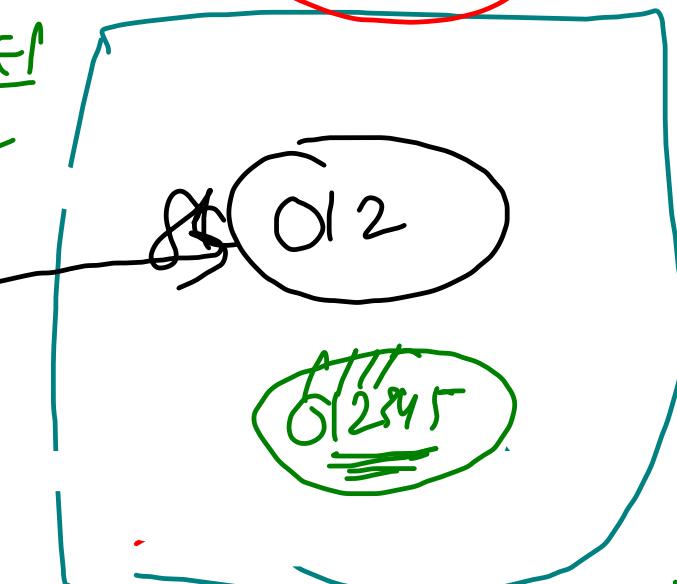
"0123456 - ... 1000"

998

012345

01234

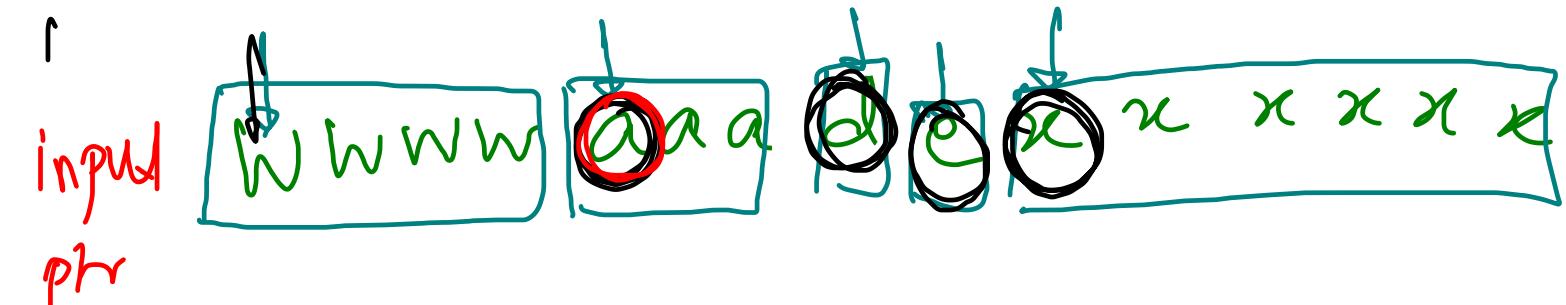
0123



0123  
0123  
0123

## String Compression

wwwwwwaaadexxxxxx  
w4a3dex6 wadex



(comp) wadex

Comp: w4a3dex

freq of prev group  
char = group starting

append last group's  
frequency  
after loop ends

```

String output = "";
int freq = 0;

for(int idx=0; idx<input.length(); idx++){
    // is character first element of it's group
    if(idx == 0 || input.charAt(idx) != input.charAt(idx - 1)){
        if(idx > 0 && freq > 1){
            output = output + freq;
        }
        output = output + input.charAt(idx);
        freq = 0;
    }
    freq++;
}

if(freq > 1){
    output = output + freq;
}

return output;

```

~~Input~~ ~~cwawaw a`aaade xx x x~~

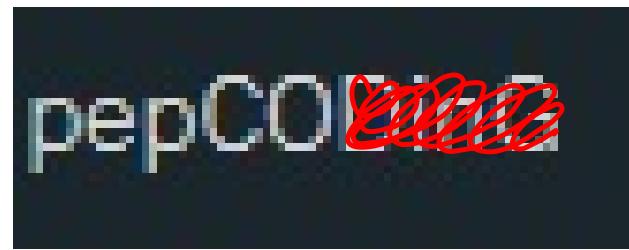
~~Output~~ ~~w4 a3de~~x~~3~~

freq = ~~∅~~ / ~~2~~ / ~~3~~ / ~~4~~

~~∅~~ / ~~2~~ / ~~3~~ / ~~4~~  
~~∅~~ / ~~2~~ / ~~3~~ / ~~4~~  
~~∅~~ / ~~2~~ / ~~3~~ / ~~4~~

③

## String With Difference Of Every Two Consecutive Characters



P -11 e 11 P -45 C 12 0

# Topics to be covered today

①

StringBuilder

Initializat& declaratn

Insert { at last, at given index }

Update { setCharAt }

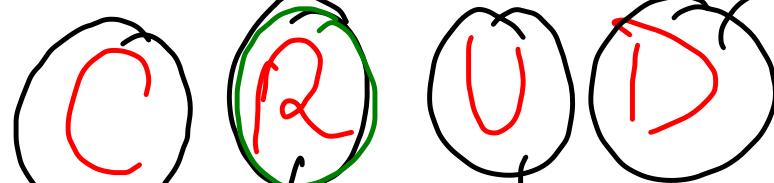
Delete { deleteCharAt }

②

ArrayList

↳ size()

remove()



get

Set

(Q) Remove Peimes

{HW} (Q)  
Discussion

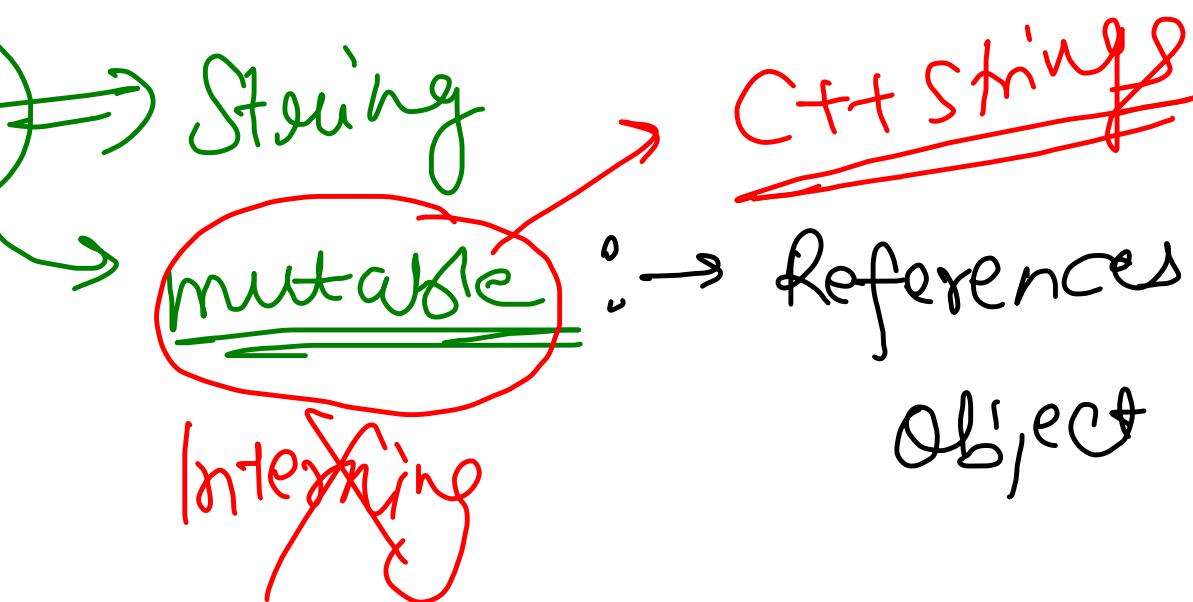
Ring Rotate

Declaratn  
& Initializatn

hello

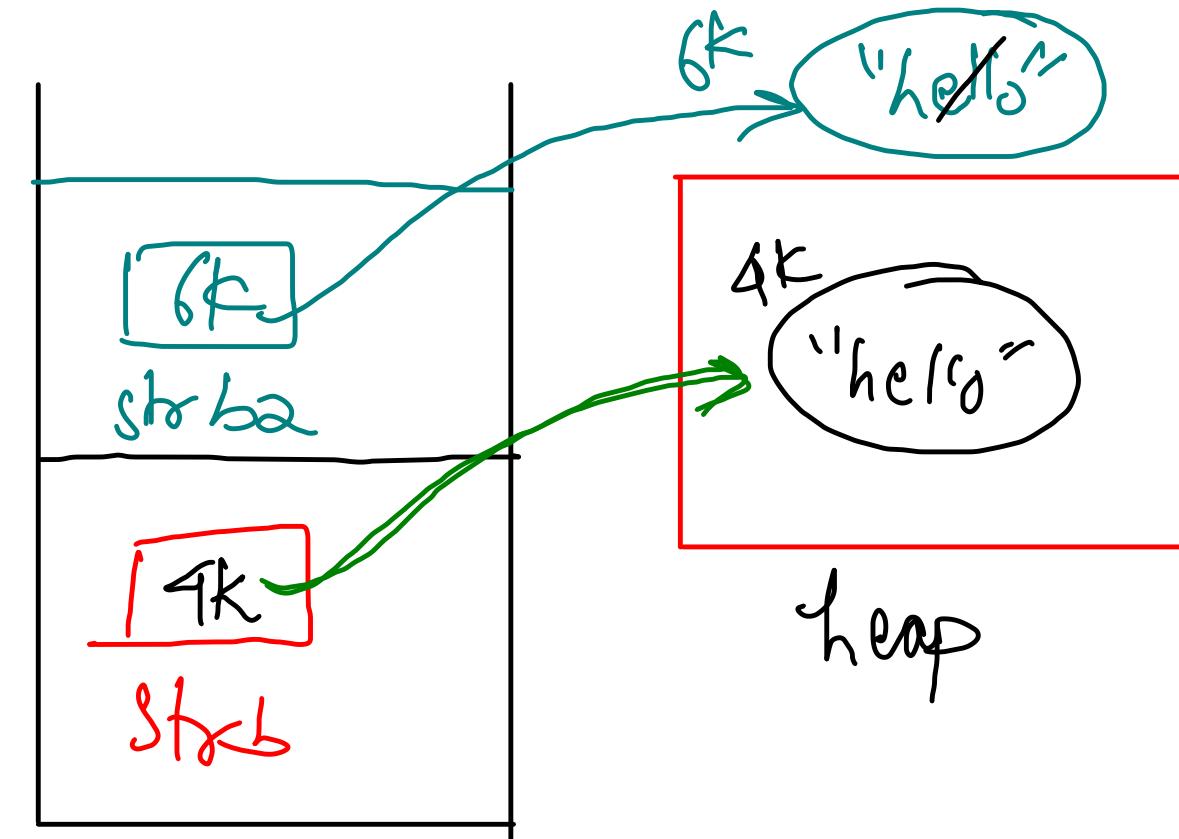
charAt

String Builders



String Builder strb = new SB("hello");

String Builder strb2 = new SB(~~"Hello"~~);



```

// Update -> Constant Operation
strb.setCharAt(3, 'd');
System.out.println(strb);

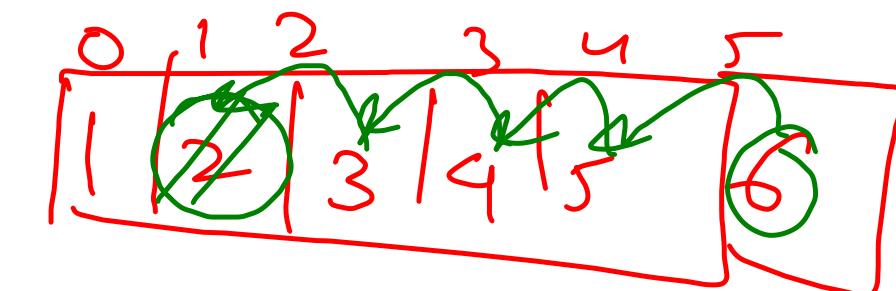
// Delete
strb.deleteCharAt(1);
System.out.println(strb);

// Insert
strb.insert(2, 'e');
System.out.println(strb);

// Append (Insert at Last) -> Constant Operation
strb.append('s');
System.out.println(strb);

```

~~a~~  
 b c d e f g h i j k l ?  $\rightarrow O(N)$



a B C d e f g h i j k l  $\Rightarrow O(N)$

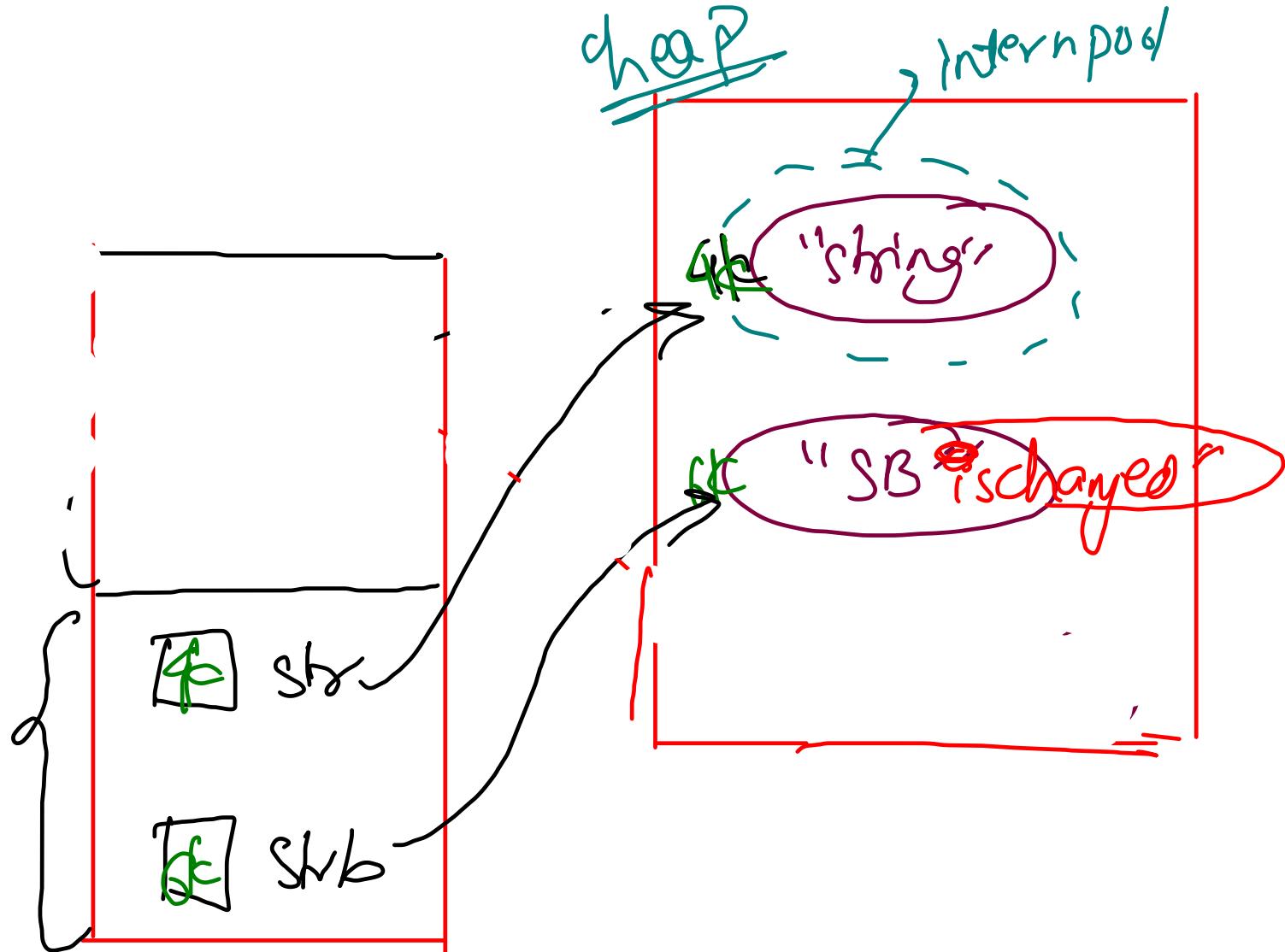
```

public static void fun(String x, StringBuilder y){
    x = x + " is changed";
    y.append(" is changed");
}

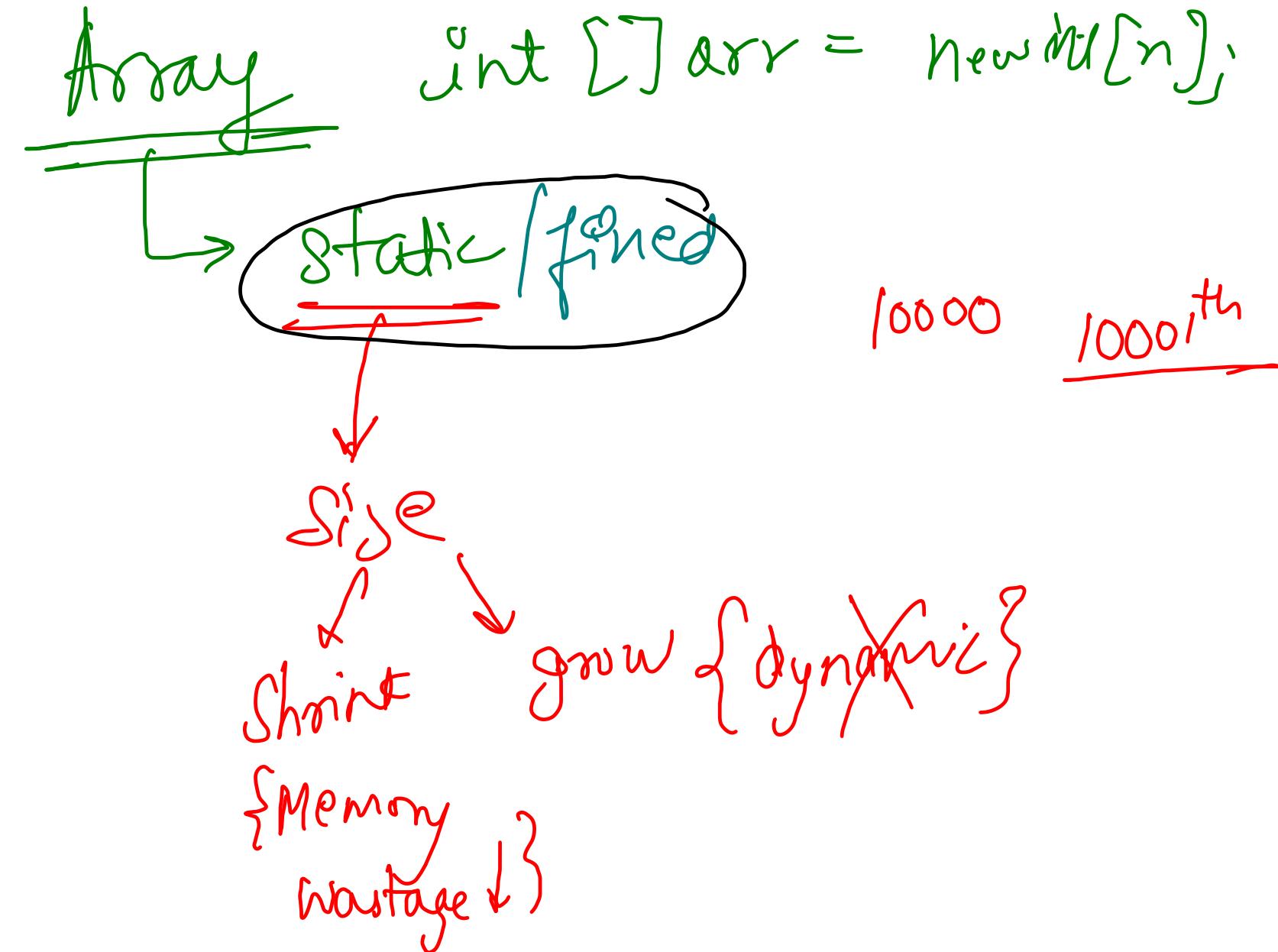
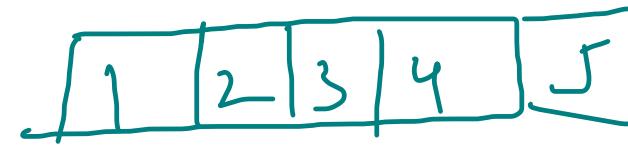
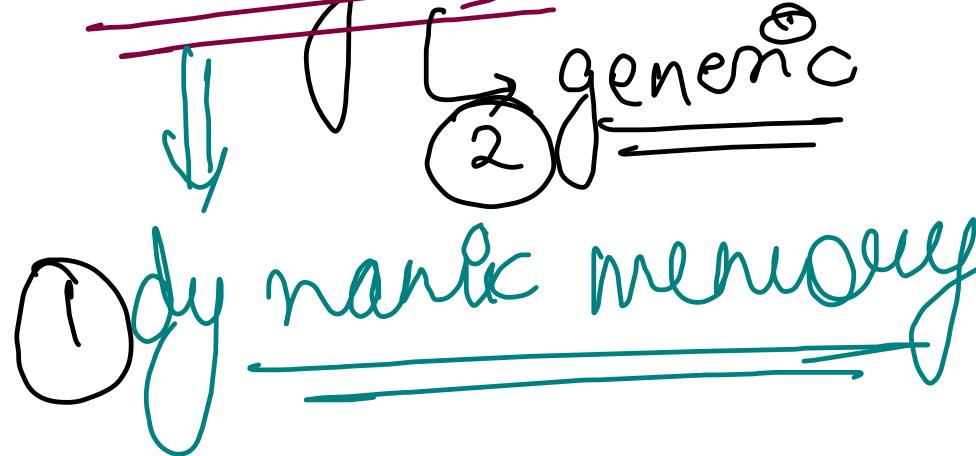
public static void main(String[] args) {
    String str = "string";
    StringBuilder strb = new StringBuilder("stringbuilder");
    fun(str, strb);
    System.out.println(str);
    System.out.println(strb);
}

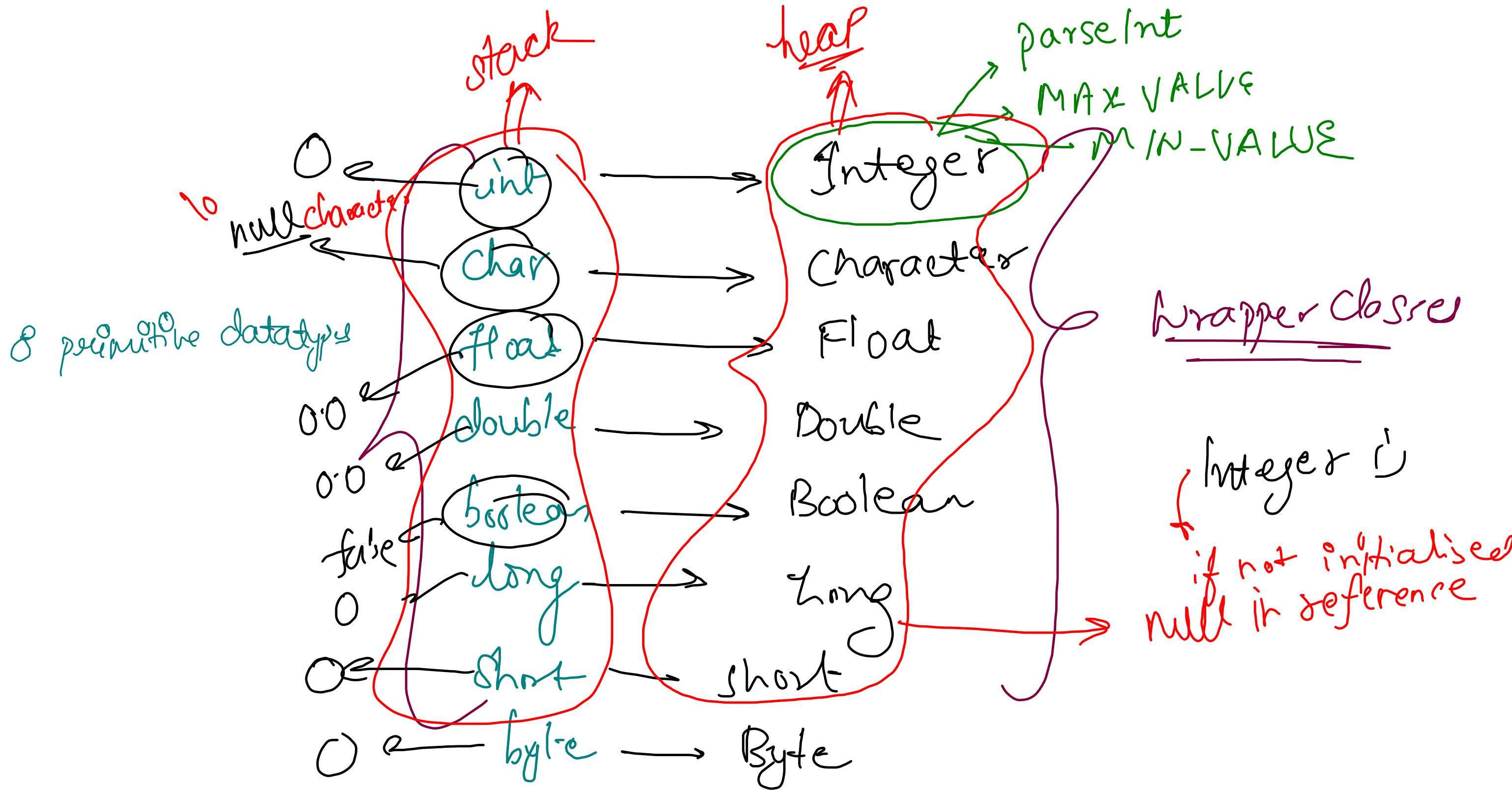
```

main



~~if ArrayList~~





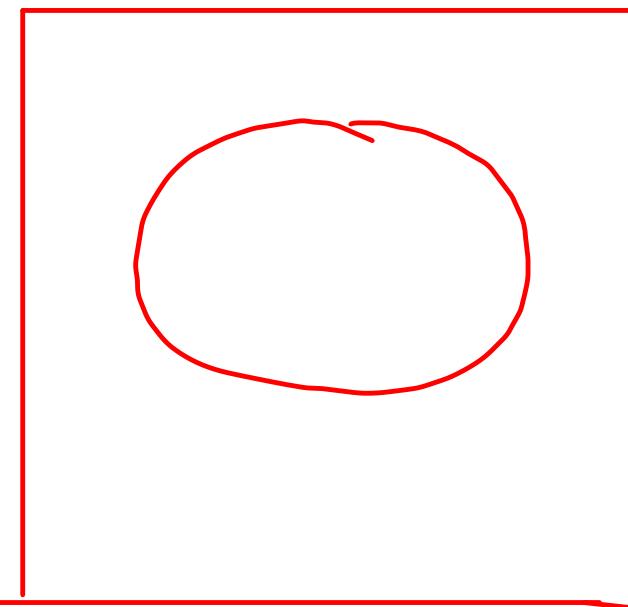
```
ArrayList < Integer > arr = new ArrayList < >();
```

```
arr.add(1)
```

```
arr.add(2)
```

4k arr

heap

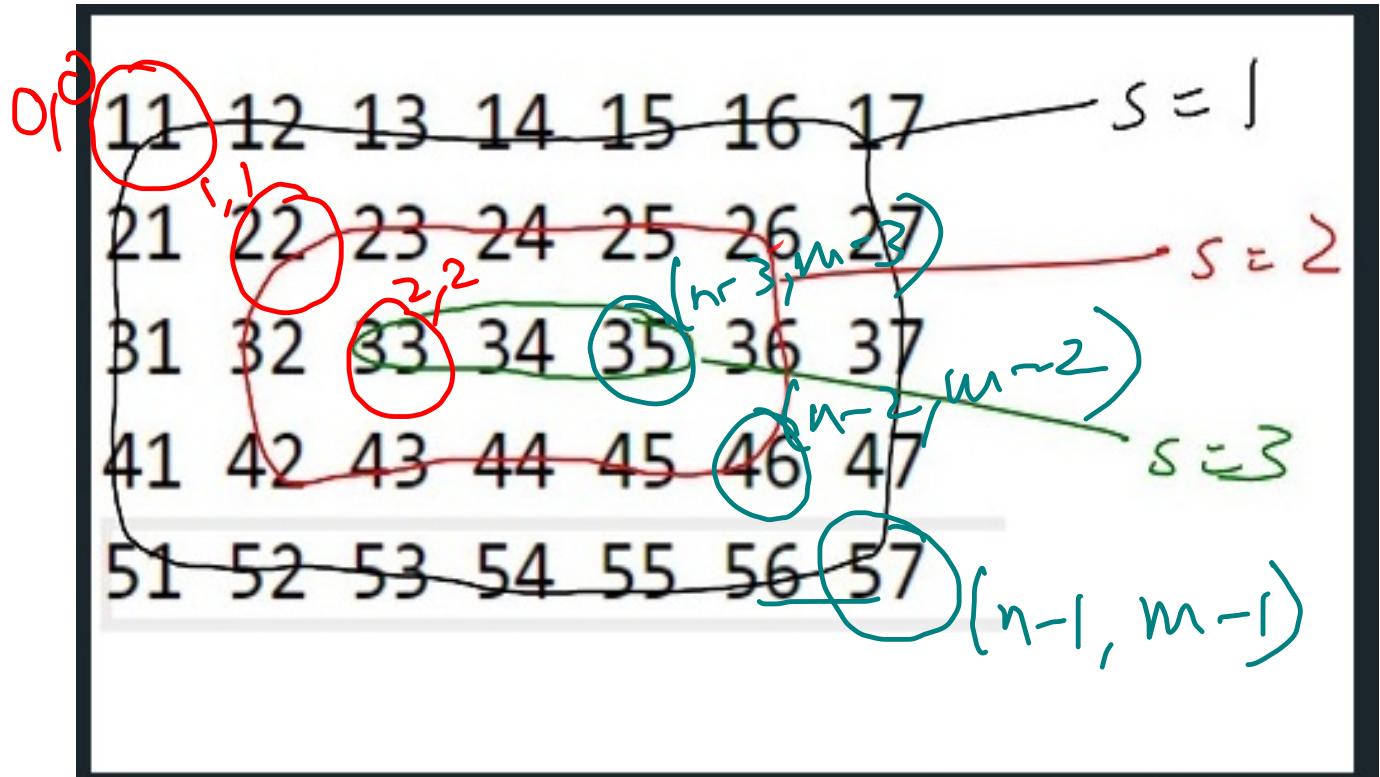


## Remove Prunes

3	4	6	13	9
0	1	2	3	4

↑  
id n + f

## Ring Rotate



shell = ②    & ②

$s^{\text{th}}$  shell  
→ top left :  $(s-1, s-1)$   
→ bottom right :  $(n-s, m-s)$

① fill shell in 1D array

② Rotate 1D array { reversal Algo }

③ Fill Rotated array back  
int the shell of  
matrix

# Strings {Contest → OS}

## Challenges



- Pangrams**  
Success Rate: 90.24% Max Score: 10 Difficulty: Easy [Solve Challenge](#)
- Maximum Frequency Character**  
Success Rate: 89.13% Max Score: 10 Difficulty: Easy [Solve Challenge](#)
- Funny String**  
Success Rate: 89.74% Max Score: 10 Difficulty: Easy [Solve Challenge](#)
- Remove Consecutive Duplicates**  
Success Rate: 97.73% Max Score: 15 Difficulty: Easy [Solve Challenge](#)
- CamelCase**  
Success Rate: 100.00% Max Score: 15 Difficulty: Easy [Solve Challenge](#)
- Gemstones**  
Success Rate: 74.07% Max Score: 20 Difficulty: Easy [Solve Challenge](#)
- Game of Thrones - I**  
Success Rate: 88.24% Max Score: 20 Difficulty: Easy [Solve Challenge](#)

## (Q1) PANGRAM

$s = \text{'The quick brown fox jumps over the lazy dog'}$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

① Freq char : 26

② Iterate on input & update freq

③ Iterate on freq array & check if any cell is 0,  
then false else true

```

public static String pangrams(String s) {
    int[] freq = new int[26];

    // Fill Frequency Array
    for(int i=0; i<s.length(); i++){
        if(s.charAt(i) >= 'A' && s.charAt(i) <= 'Z'){
            freq[s.charAt(i) - 'A']++;
        } else if(s.charAt(i) >= 'a' && s.charAt(i) <= 'z'){
            freq[s.charAt(i) - 'a']++;
        }
    }

    // Check if string is pangram: Iterating on frequency array
    for(int i=0; i<26; i++){
        if(freq[i] == 0){
            return "not pangram";
        }
    }
    return "pangram";
}

```

A' B' C' D'

$$D' : 65 + 3 = 68$$

D' - A'

$$68 - 65 = 3$$

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
0	0	1	1	12										1	12	1									

```

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

    int[] freq = new int[26];

    for(int i=0; i<s.length(); i++){
        freq[s.charAt(i) - 'a']++;
    }

    int maxFreq = 0;
    char maxFreqChar = 'a';
    for(int i=0; i<26; i++){
        if(freq[i] > maxFreq){
            maxFreqChar = (char)('a' + i);
            maxFreq = freq[i];
        }
    }

    System.out.println(maxFreqChar);
}

```

"pepcoder"

maxFreq = ~~x~~'e'

~~0~~ 2

(Q2) Maximum  
Frequency  
char

# (Q3) Funny String

"abc"  
11

"cba"  
rr

```
public static String funnyString(String s) {  
    // Write your code here  
    StringBuilder rev = new StringBuilder(s);  
    rev.reverse();  
  
    for(int i=0; i<s.length()-1; i++){  
        int val1 = Math.abs(s.charAt(i + 1) - s.charAt(i));  
        int val2 = Math.abs(rev.charAt(i + 1) - rev.charAt(i));  
  
        if(val1 != val2) return "Not Funny";  
    }  
  
    return "Funny";  
}
```

(Q1) Remove Consecutive Duplicates

aabbccccc

{ String Compression - I }

```
public static int camelcase(String s) {  
    // Write your code here  
    int count = 0;  
    for(int i=0; i<s.length(); i++){  
        if(i == 0 || (s.charAt(i) >= 'A' && s.charAt(i) <= 'Z')){  
            count++;  
        }  
    }  
    return count;  
}
```

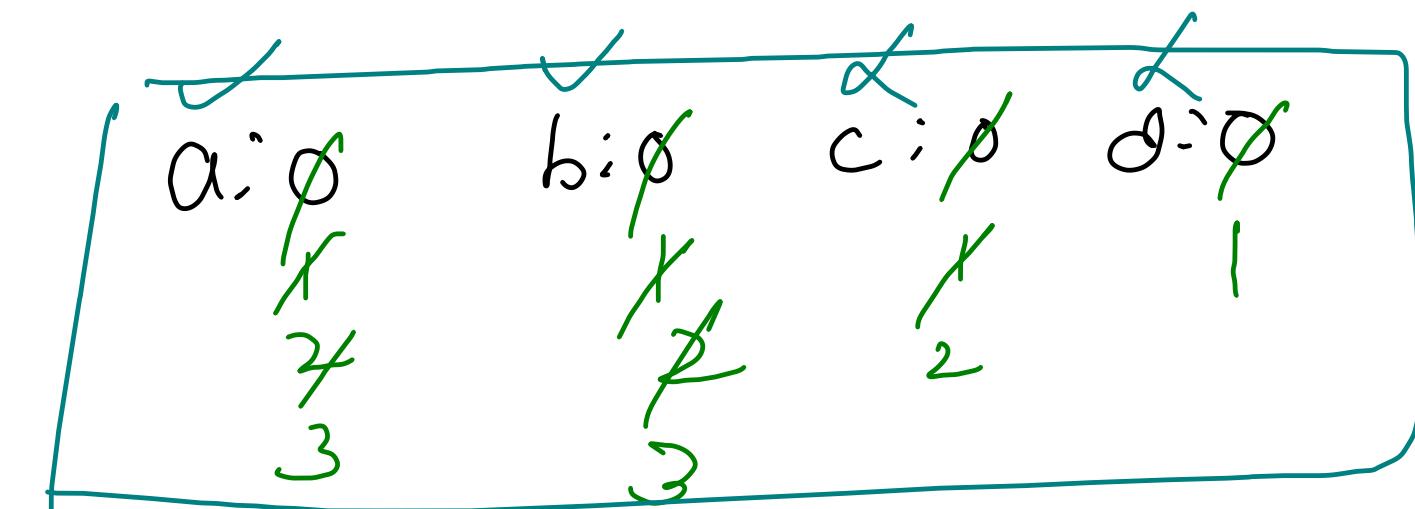
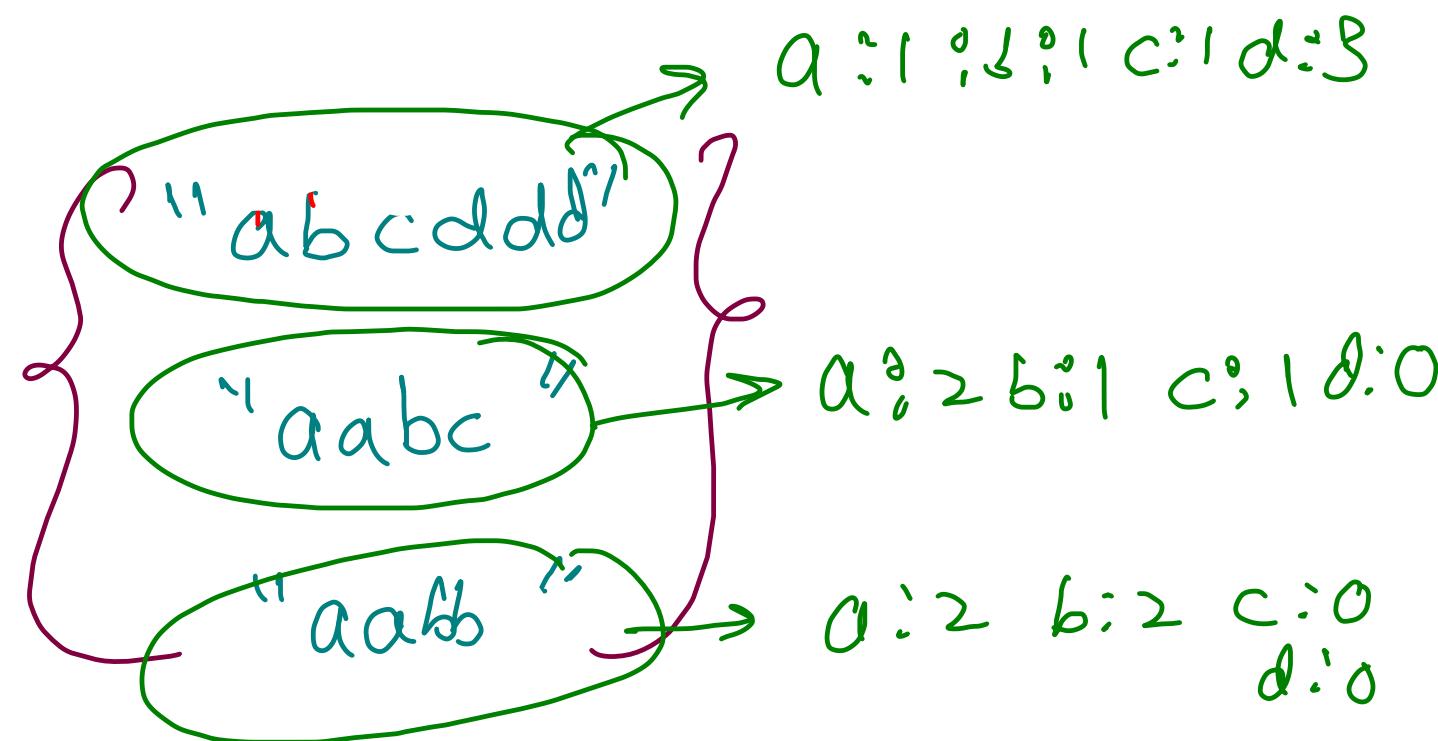
## (Q5) Gemstones

```
// Frequency Array of Minerals -> LowerCase Alphabets
int[] freq = new int[26];

// Loop on Array of Strings
for(int i=0; i<arr.size(); i++){
    // Loop on Character of String
    int[] tempFreq = new int[26];
    for(int j=0; j<arr.get(i).length(); j++){
        char ch = arr.get(i).charAt(j);
        tempFreq[ch - 'a']++;
    }

    for(int j=0; j<26; j++){
        if(tempFreq[j] > 0){
            freq[j]++;
        }
    }
}

int countOfGemStones = 0;
for(int i=0; i<26; i++){
    // is Mineral present in all the stones
    if(freq[i] == arr.size()){
        countOfGemStones++;
    }
}
```



# (QG) Game of Thrones { Palindromic Permutation }

"aabbaadd"

even

if all  
characters  
have  
even freq

a: 4  
b: 2  
d: 2

1000 --- 00 000

abad / dab a

"ābāc dāb cād dād b"

a: 3 b: 4 c: 2 d: 4

abbcddaddcbbba

Odd length

Only one character will have odd frequency

remaining should have even

```
public static String gameOfThrones(String s) {  
    // Write your code here  
    int[] freq = new int[26];  
    for(char ch: s.toCharArray()) {  
        freq[ch - 'a']++;  
    }  
  
    int oddCount = 0;  
    for(int i=0; i<26; i++) {  
        if(freq[i] % 2 == 1) {  
            oddCount++;  
        }  
    }  
  
    if(oddCount <= 1) return "YES";  
    else return "NO";  
}
```

① R&B

② S&S

Time Comp

Space Comp

③ LL

④ OOPS

⑤ S&Q