

Today's agenda

↳ quizzes & learning

Quiz 1:

$\rightarrow \text{int } n = 10;$

Syntax to declare the variable?

- a) type variable-name = value ; X
- b) variable-name type = value ;
- c) variable-name = value ;
- d) None of the above.

Quiz 2:

Declare an int variable with name y and value 20?

a) Int y = 20; xx

$\hookrightarrow \text{int } y = 20;$ (c)

Quiz 3

$\text{int } a = 10;$

$\boxed{10}$
a

$\boxed{20}$
b

$\text{int } b = 20;$

$\text{System.out.println}(a+b); \rightarrow 30$

Quiz 4:

```
int a = 20;
```

```
int b = 30;
```

→ removed from
radar board

```
System.out.println(a + " " + b);
```

↳ 20 30

Quiz 5:

```
int temp: 10;
```

↑
→

```
System.out.println(Temp);
```

↳ error

Quiz 6:

```
System.out.println(a);
```

```
int a = 10;
```

↳ error

Quiz 7:

```
int a = 20;
```

```
int a = 30;
```

```
System.out.println(a);
```

↳ error

you can't initialize

same variable name

twice?

Quiz 8:

```
int n = 20;
```

```
System.out.println(n);
```

```
n = 40;
```

```
System.out.println(n);
```



2040



Quiz 9:

```
int a = 10;
```



10 = 10

```
int b = 20;
```

↳ a = ~~b~~ + 30;

LHS = RHS

```
System.out.println(a);
```

→ 50

LHS $\stackrel{+}{=}$ RHS

↳ Computer \rightarrow = \rightarrow assignment operator

↳ assign right side

value to left side.

* Variable naming rules

1. Name can contain lowercase, uppercase alphabets or digit (0-9) or '\$' (dollar) or '-' (underscore)

2. First letter / character of the name cannot be number.

3. Can't use reserve keywords as variable name:

Reserved word: words which are already predefined in java. ex: Public, Static, void, int etc.

Quiz 10:

int m = 10; ✗

int 1y = 20; ✗

⇒ ans = 1

int y@b = 30; ✗

Q: How many of above are correct variable names?

Quiz 11:

int -y = 10; ✗

int any = 20; ✗

⇒ ans = 2

int or a = 30; ✗

int y\$2 = 45; ✗

Q How many of above are correct variable names?

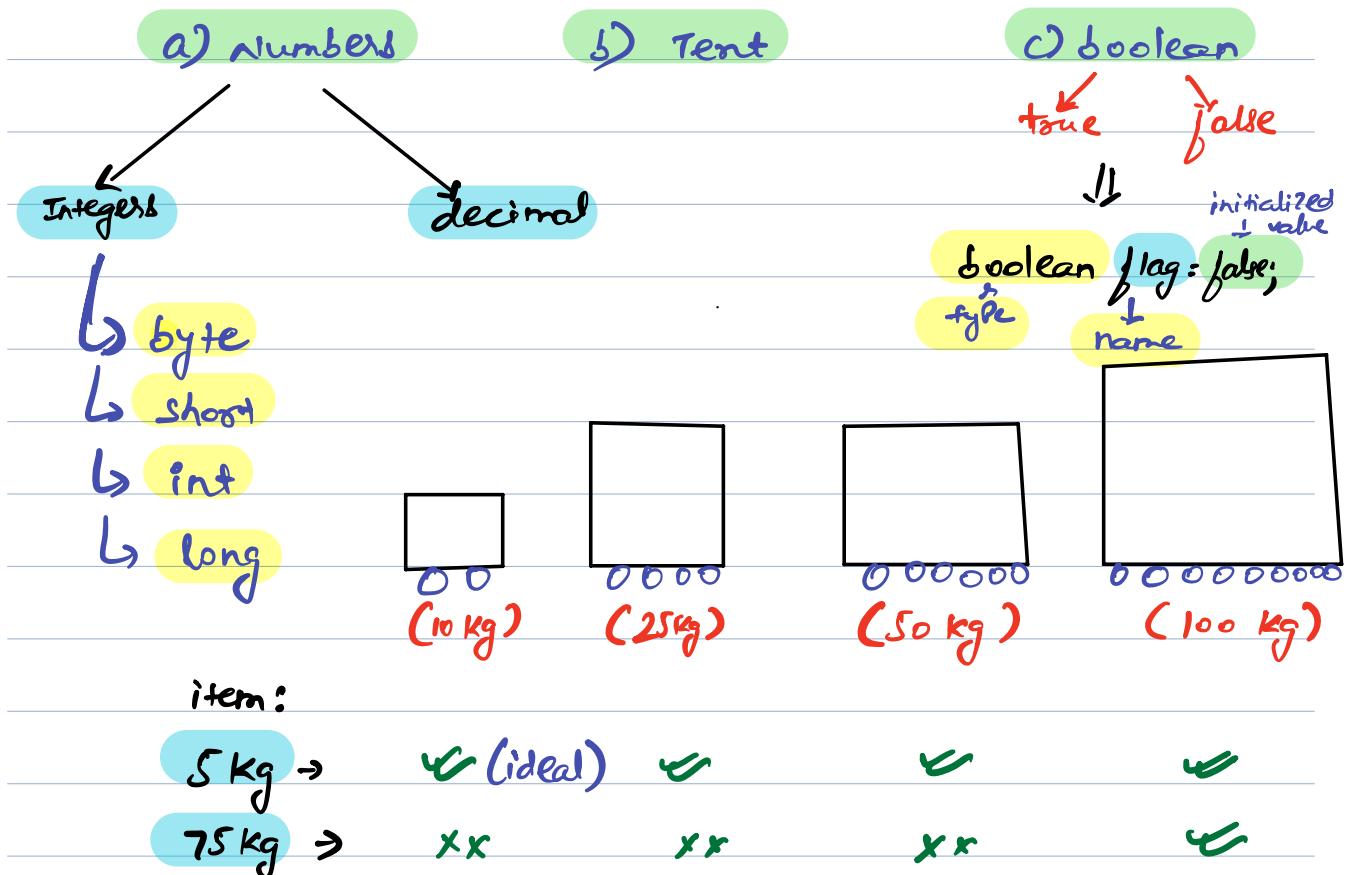
Quiz 12:

int Static = 60;

→ error

System.out.println(Static);

Different Categories of data



Range / capacity

Capacity	byte short int → $\{-2^{31} \dots 2^{31}\}$ long → $\{-2^{63} \dots 2^{63}\}$	$-2^7 \dots 2^7 - 1 : \{-128, 127\}$
range		$-2^{15} \dots 2^{15} - 1 : \{-32768, 32767\}$
are different		$-2^{31} \dots 2^{31} - 1 : \{-214\dots, 214748\dots\}$
		$-2^{63} \dots 2^{63} - 1 : \{ \dots, \dots \}$

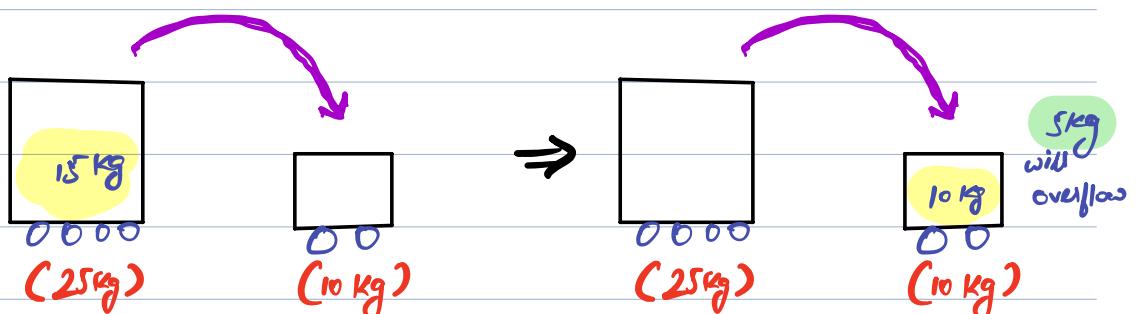
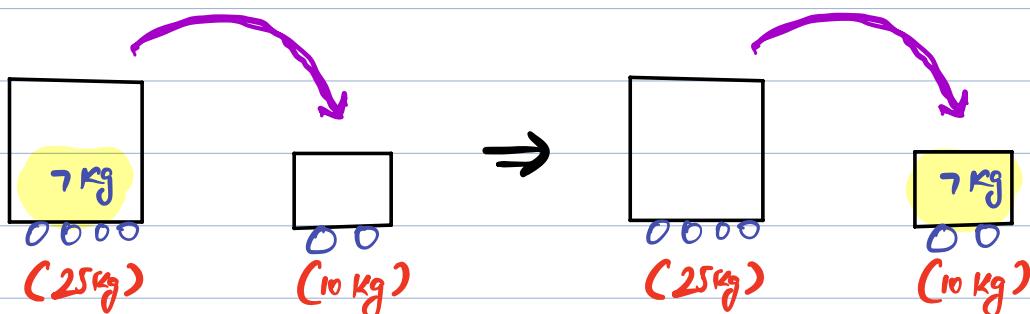
Complete explanation is in Bit manipulation Class.

↳ The default value is int. To initialize long you need long $\alpha = 100000000000L$.

Break till 9:29 PM

↳ you didn't come this far only to come this far.

→ Type Casting



Approach

int $\rightarrow \{-2^{31} \text{ to } 2^{31}-1\} \rightarrow \{-2 \times 10^9 \text{ to } 2 \times 10^9\}$
long $\rightarrow \{-2^{63} \text{ to } 2^{63}-1\} \rightarrow \{-2 \times 10^{18} \text{ to } 2 \times 10^{18}\}$

ex1: int n = 100

long y = n; \rightarrow implicit

System.out.println(y);

ex2:

long n = 1000;

explicitly \leftarrow int y = (int)n;

System.out.println(y); \rightarrow 1000

1000
n

1000
y

Quiz 13

int $a = 10000;$

long $b = a;$

System.out.println (b);

6 10000

Quiz 14:

long $a = 100000;$ → int $y = (int)a$

int $y = a;$ → Typecast?

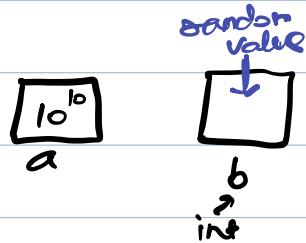
System.out.println (y);

Quiz 15:

long $a = 10^{10};$

int $b = (int)a;$

System.out.println (b);



* reading Inputs

↳ Scanner ^{name} scn = new Scanner (System.in);
int n = scn.nextInt();
System.out.println (n);

Quiz 16:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

System.out.println (n);

b error

(Small S in
System)

Quiz 17:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

small S

System.out.println (n);

b error

Quiz 18:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

System.out.println (n);

b 30

Quiz 19:

Input: 24 30

Scanner Scn = new Scanner (System.in);

int x = Scn.nextInt(); ← 24

int y = Scn.nextInt(); ← 30

System.out.println(x); → 24

Quiz 20:

Input: 24 30

Scanner Scn = new Scanner (System.in);

int x = Scn.nextInt();

int y = Scn.nextInt();

int z = Scn.nextInt();

System.out.println(x+y+z);

error

↳ Done with today's class

Quiz 21 :

float $n = 3.4f;$

double $y = n;$

System.out.println(y);

Quiz 22 :

int $n = 40;$

double $y = n;$

System.out.println(y);

Quiz 23 :

double $n = 3.45;$

→ error

int $y = n;$

System.out.println(y);



Today's agenda

- ↳ float and double
- ↳ Taking input
- ↳ operators
- ↳ if else



AlgoPrep



// Decimal → 2.2, 1.3, 4.7, 2.6 etc.

↳ float & double
↳ int (0.6-7 decimal places) ↳ long (0.14-15 decimal places)

→ In Integer, default values are int

→ In decimal, default values are double.

Writing float

↳ float a = 2.4f;

Writing double

↳ double d = 4.2;



Quiz 1:

double d = 2.8;

2.8

System.out.println(d);

Quiz 2:

float f = 3.3f;

3.3

System.out.println(f);

Quiz 3:

↳ float f = 3.4f;

↳ double d = f;

3.4
d

3.4
d

System.out.println(d); → 3.4

Quiz 4:

double d = 3.4;

↳ float f = d; → error

3.4
d

System.out.println(f);

double d = 3.4;

↳ float f = (float)d;

System.out.println(f);



2 Golden rules of typecasting

1. If there is no loss of data then no error: implicit

2. If there is chance for loss of data then error.

We can still do this change forcefully: explicit (typecasting)

[int to long → works
float to double → works]

long to int → explicit
double to float → explicit

// operation



↳ Rule1: Mathematical operation between decimal and non-decimal, Resultant: Decimal

↳ Rule2: Operation between same Category but different Capacity, Resultant: Higher Size

a	b	res
int	int	int
int	long	long
long long	int long	long long

Quiz5:

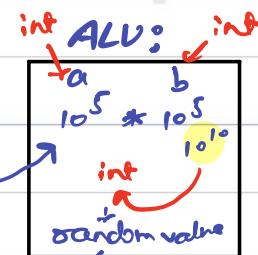
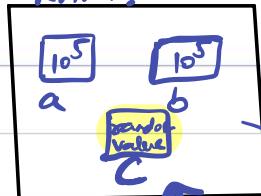
int a = 100000;

int b = 100000;

↳ int c = a * b;

System.out.println(c);

RAM:

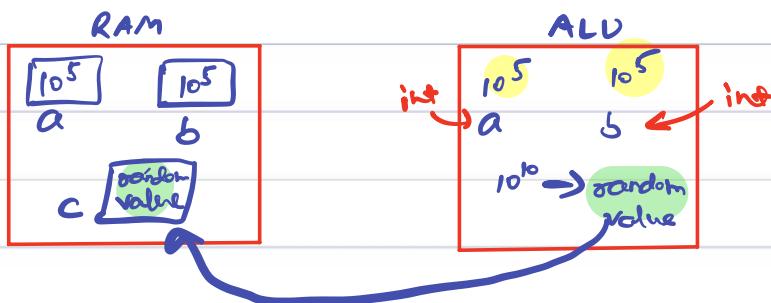




Quiz 6:

```

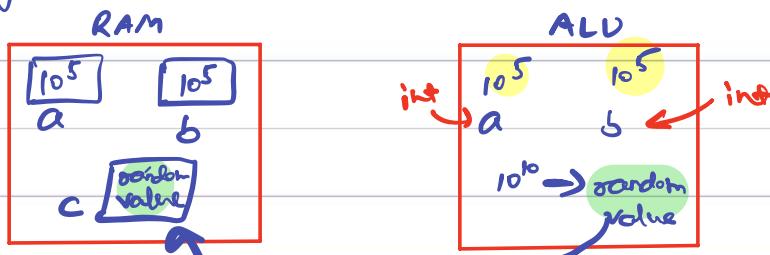
int a = 100000;
int b = 100000;
long c = a*b;
System.out.println(c);
    
```



Quiz 7:

```

int a = 100000;
int b = 100000;
long c = (long)(a*b);
System.out.println(c);
    
```



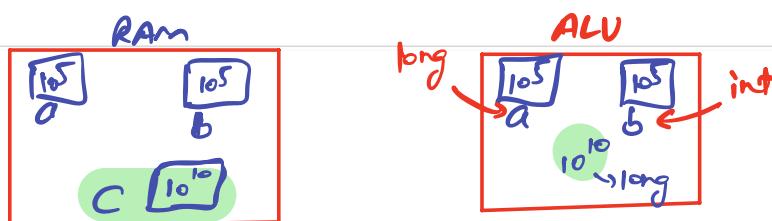
Quiz 8:

```

int a = 100000;
int b = 100000;
long c = (long)(a)*b;
System.out.println(c);
    
```

$\rightarrow b$ is correct

10^{10}





* Arithmetic operators

↳ +, -, *, /, % → remainder

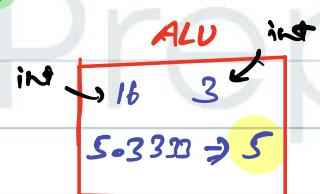
$$\hookrightarrow 20 \% 3 = 2$$

Quiz 9:



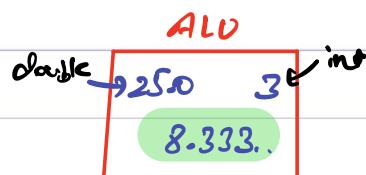
System.out.println(16/3);

→ 5



Quiz 10:

System.out.println(25.0/3); → 8.333...



Quiz 11:

System.out.println(35%9); → 8

↳ $x \% 7 \rightarrow \{0, \dots, 6\}$

$x \% n \rightarrow \{0, \dots, n-1\}$

Break till 9:25



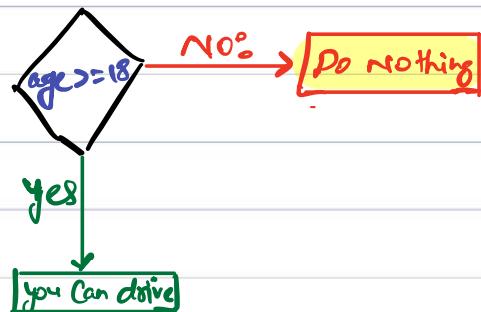
* Relational operators: Used to check relation between 2 data ex: \geq , $<$, $>$ etc.

	$x=8 \ y=10$	$x=15 \ y=7$	$x=13 \ y=13$
x less than y : $x < y$	true	false	false
x greater than y : $x > y$	false	true	false
x greater than equal to y : $x \geq y$	false	true	true
x smaller than equal to y : $x \leq y$	true	false	true
x equal y : $x == y$	false	false	true
x not equal y : $x != y$	true	true	false

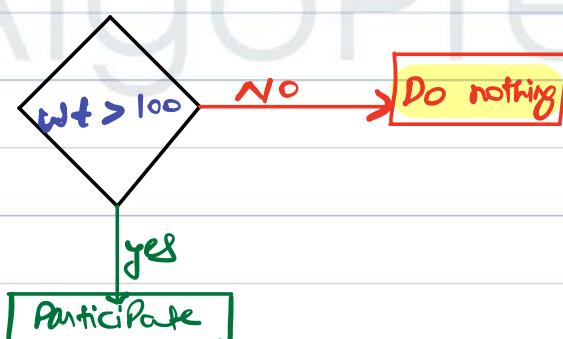


// If

1. Check if person is eligible for car driving license.



2. Check if person is above 100 kg in weight. he can participate in weight lifting.



idea: When we want to do something on the basis of Condition being true.

Syntax:

if (condⁿ) { → only condⁿ which will give you true/false.
// Statement or lines you want to execute if the condⁿ is true. } (Relational op)

}

- a) $4 == 5$ ↗ (Valid but give you false)
- b) $4 < 5$ ↗

c) $4+5 \times 2$



→ age of person

- Q) Read a number and if person is eligible, print "eligible" for driving license.

IPShedo code

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

```
int age= scn.nextInt();
```

```
if (age >= 18) {
```

```
System.out.println ("eligible");
```

```
}
```

Quiz 12 :



```
int n= 20;
```

```
if (n >= 15) {
```

```
System.out.println ("Hello");
```

```
}
```

```
System.out.println ("Hello");
```

20
n

Quiz 13 :

```
int n= 20;
```

```
int y= 25;
```

20
n 25
y

```
if (n >= 25) {
```

```
System.out.println ("AlgoPrep1");
```

```
}
```

```
if (y >= 25) {
```

→ AlgoPrep2

```
System.out.println ("AlgoPrep2");
```

```
}
```



Quiz 14:

```
if (10 > 6) {  
    System.out.println("1st");  
}  
if (15 > 25) {  
    System.out.println("2nd");  
}
```

1st

Quiz 15:

```
int n = 55;  
int y = 65;
```

55
n

67
y

```
if (n > 55) {  
    System.out.print ("first");  
    n = n + 2;
```

Second 122

```
if (y >= 60) {  
    System.out.print ("second");  
    y = y + 2;  
}  
System.out.println (n+y);
```

65
2
↓
67



Today's agenda

↳ if / else

↳ if / else if ... - else

↳ while loop

1st: you guys will get ASSIGN/H.W starting from next class. leetcode / cfc / HackerRank.

HackerRank



Practice Page → add → leetcode/gbg
link:

leaderboard ↗

2nd → quickly solve → levelup

1st / 2nd / 3rd
working projects
foundation

complete

levelup

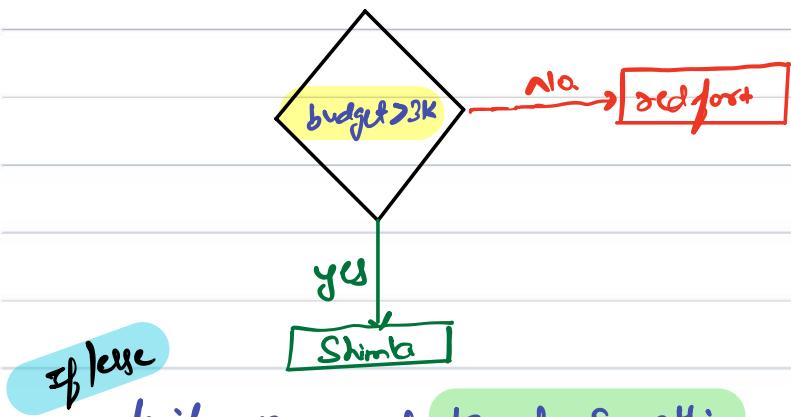
4th year / badges
foundation →
Assays →
Arrays levelup
decodings



// If / Else

budget > 3K → Shimla

budget < 3K → Red fort



If else

If we want to do something when the Condⁿ is true and something else if Condⁿ is false.

Syntax

if (Condⁿ) {
 // Statement 1
}
else {
 // Statement 2
}

if Condⁿ → true : Statement1
if Condⁿ → false : Statement2



11 Pseudo Code

```
if (budget > 3000) {
    System.out.println("Shimla");
}
else {
    System.out.println("Mumbai");
}
```



AlgoPrep



Q) Take input mark scored by a student. Print "Pass" if
mark ≥ 35 , otherwise Print "fail".

```
if (mark >= 35){  
    System.out.println("Pass");  
} else {  
    System.out.println("Fail");  
}
```

Quiz 1 :



```
if (15 > 7) {  
    System.out.println("if");  
} else {  
    System.out.println("else");  
}
```

Quiz 2 :

```
int n = 70;  
if (n > 70){  
    System.out.println("if");  
}  
else {  
    System.out.println("else");  
}
```



Q) Read a number and check if number is even or

odd? Even no: divisible by 2 → remainder 0

Odd no: not divisible by 2

Scanner scn = new Scanner (System.in);

int num = scn.nextInt();

9%2 → 1

12%2 → 0

if (num%2 == 0) {

System.out.println ("even");

else {

System.out.println ("odd");

}

// logical operators → &&, || and & or

&&

Cond¹ && Cond²

answer

false
dominated
relation

T T

T

F T

F

T F

F

F F

F

if (cond¹ cond²)

used to club
multiple Condⁿ

int a = 50;

int y = 30;

if (a > 60 && y > 20) {

// Statement 1

→ statements

3

else {

// Statement 2

}



→ 3 condⁿs

Condⁿ1 & Condⁿ2 & Condⁿ3 answer

T	T	T	T
T	T	F	F

→ if you want to execute "if" only when all the
Conditions are true? → **and**

or || → or

true dominated relation

Condⁿ1 || Condⁿ2

T	T
T	F
F	T
F	F

answer

T
T
T
F



- Q) Read a number. If number is divisible by 2 or 3
Point "divisible", otherwise Point "not divisible".

```
Scanner scn = new Scanner (System.in);
int num = scn.nextInt();
num: 8 → divisible
num: 36 → divisible
if ((num % 2 == 0) || (num % 3 == 0)){
    System.out.println ("divisible");
} else {
    System.out.println ("Not divisible");
},
```

- Q) Read a number. If number is divisible by 2 and 3
Point "divisible", otherwise Point "not divisible".

6 ide

```
Scanner scn = new Scanner (System.in);
int num = scn.nextInt();
if ((num % 2 == 0) && (num % 3 == 0)){
    System.out.println ("divisible");
} else {
    System.out.println ("Not divisible");
},
```



if | Else if () ... | Else

Syntax

↳

```
if (Condn) {  
    // Statement 1  
    } else if (Cond2) {  
    // Statement 2  
    } else if (Cond3) {  
    // Statement 3  
    } else {  
    // Statement 4
```

10 blocks → 9 condⁿ

Condⁿ true → Statement 1 & skip remaining if else

Condⁿ false → Statement 2 & skip remaining if else

Cond¹ false
Cond² true
Cond³ false
Cond⁴ true

→ Statement 3

All the condⁿs are false → else



Comparison

if (condⁿ₁) {
 // Statement 1

}

if (condⁿ₂) {
 // Statement 2

}

if (condⁿ₃) {
 // Statement 3

3

if (condⁿ₄) {
 // Statement 4

3

if (condⁿ₁) {
 // Statement 1

3

else if (condⁿ₂) {
 // Statement 2

3

else if (condⁿ₃) {
 // Statement 3

3

else {
 // Statement 4

3

↳ Condⁿ₁ & Condⁿ₃ are true



Statement 1

Statement 3



Statement 1



Q) Given 3 numbers, Point max out of these.

ex:	a	b	c	ans
	6	3	4	6
	8	3	10	10
	12	4	4	12
	13	10	13	13
	8	6	8	8
	11	11	11	11

a	b	c	
13	10	13	→ 13
11	11	11	→ 11
8	3	10	→ 10
0	2	0	→ 2

II) Pseudo code

```
int a = scn.nextInt();  
int b = scn.nextInt();  
int c = scn.nextInt();  
  
if (a >= b && a >= c){  
    System.out.println(a);  
} else if (b >= a && b >= c){  
    System.out.println(b);  
}  
else {  
    System.out.println(c);  
}
```

Break till 9:43 PM



11 Intro to loops

↳ Print all numbers from 1 to 5.

```
System.out.println(1);  
System.out.println(2);  
System.out.println(3);  
System.out.println(4);  
System.out.println(5);
```

↳ Print numbers from 1 to 1000.

loop! Do same thing multiple times

↳ a) while loop ↗

b) for loop ↗

c) ~~do while~~ xx

Q) Print 1 to 5 using while loop.

```
int i=1;
```

while ($i \leq 5$) { "Cond": loops will run till the Cond is true. }

```
    System.out.println(i);
```

```
    i = i+1;
```

```
}
```



Dry run

i	$i <= 5$	Point
1	T	$1 \rightarrow i = i + 1 \rightarrow i = 2$
2	T	$2 \rightarrow i = i + 1 \rightarrow i = 3$
3	T	$3 \rightarrow i = i + 1 \rightarrow i = 4$
4	T	$4 \rightarrow i = i + 1 \rightarrow i = 5$
5	T	$5 \rightarrow i = i + 1 \rightarrow i = 6$
6	F	

$i = i + 1$ & $i++$ are same

// Structure of while loop

1. Initialize loop variable.

```
int i=1;
```

2. Work while with condn.

```
while ( $i <= 100$ ) {
```

```
}
```

3. The statement you want to run.

```
System.out.println(i);
```

4. updation of loop variable.

```
i = i + 1; or i++;
```



Quiz 3

```

int i=5;
while(i<10){
    System.out.print(i);
    i=i*2;
}
  
```

i	i < 10	Point
5	T	5 → i: i+2 → i: 10
10	F	

Quiz 4:

```

int i=1;
while(i<5){
    System.out.print(i+ " ");
    i=i+2;
}
  
```

i	i < 5	Point
1	T	1 → i: i+2; i: 3
3	T	3 → i: i+2; i: 5
5	F	

1 3...

Quiz 5:

```

int i=1;
while(i>=5){
    System.out.println(i);
    i++;
}
  
```

i	i >= 5	Point
1	F	



Quiz 6:

`int i=1;`

`while (i<=5) {`

`System.out.print(i);`

`}`

i	i < 5	Point
1	T	1
1	T	1
1	T	1
...	...	

Quiz 8:

`int i=0;`

`while (i<=5) {`

`System.out.println ("AlgoPep");`

`i = i + 1;`

`}`

How many times AlgoPep will be printed?

6

i	i < 5	Point
0	T	1st Point i=1
1	T	2nd Point i=2
2	T	3rd Point i=3
3	F	
4	T	
5	T	
6	F	6th Point i=6

↳ 15 classes → language won't matter to you

↓
Writing the code is easiest.



Pain of discipline or Pain of regret

Class starts at 8:15pm today

↳ Signup the Hackerrank Contest Pinned
in the chat.

→ Class will end at 10:30 today.

- ↳ Problems
- ↳ for loops
- ↳ break / continue .



Q) Print reverse

Given an integer N , print all digits from Right to left.

$$\text{ex: } N = 2375$$

↳ 5 7 3 2

//idea!

$$N = 142 \rightarrow \text{int lastdigit} = N \% 10 = 2$$

$$N = 15736 \rightarrow \text{int lastdigit} = N \% 10 = 6$$

// Pseudo Code

P S v main () {

Scanner scn = new Scanner (System.in);

int n = scn.nextInt(); → 1234

if ($n < 0$) { n = $n * -1$; }

$n \neq 0$

while ($n > 0$) {

int lastdigit = $n \% 10$;

System.out.println (lastdigit);

$n = n / 10$;

3

$$n = 10$$

System.out.println(n); → 10



$$n = n + 1;$$

System.out.println(n); → 11

~~$$n + 1;$$~~¹²

System.out.println(n); → 12

$$n = n + 1;$$

"12"
~

$$n = 1234$$

P S v main () {

Scanner scn = new Scanner (System.in);
int n = scn.nextInt(); → 1234

while (true) {

int lastdigit = n % 10;
System.out.println (lastdigit);
n = n / 10;

3

$$0$$

n

lastdigit: 1

4
3
2
1

Variables created inside while() loop gets deleted
before next iteration.

$$n = 12004$$



// For loop basics

```
int i=0;  
while (i<10) {  
    // Statement  
    i++;  
}
```

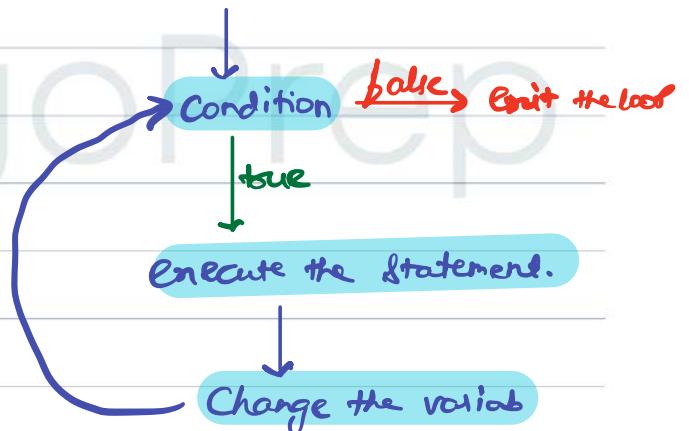


```
for (int i=0; i<10; i++) {  
    // Statement  
}
```

3

flow:
↓

1 time: initialize i



Q) Print numbers from 1 to 5 using for loop.

```
for (int i=1; i<=5; i++) {  
    System.out.println(i);  
}
```



Q) Count factors

Given a Positive number, Point all the factors of that number.

Ex: $12 : 1 \ 2 \ 3 \ 4 \ 6 \ 12$ → numbers completely dividing n
remainders should be 0]

Quiz: $24 : 1 \ 2 \ 3 \ 4 \ 6 \ 8 \ 12 \ 24$

minimum factor of n is 1

maximum factor of n is n

figure out factors between 1 to n .

Ideas

$$N = 10$$

$$1 \rightarrow N \% 1 \Rightarrow 10 \% 1 = 0 \quad \checkmark$$

$$2 \rightarrow N \% 2 \Rightarrow 10 \% 2 = 0 \quad \checkmark$$

$$3 \rightarrow N \% 3 \Rightarrow 10 \% 3 = 1 \quad \times \times$$

$$4 \rightarrow N \% 4 \Rightarrow 10 \% 4 = 2 \quad \times \times$$

$$5 \rightarrow N \% 5 \Rightarrow 10 \% 5 = 0 \quad \checkmark$$

$$6 \rightarrow N \% 6 \Rightarrow 10 \% 6 = 4 \quad \times \times$$

$$7 \rightarrow N \% 7 \Rightarrow 10 \% 7 = 3 \quad \times \times$$

$$8 \rightarrow N \% 8 \Rightarrow 10 \% 8 = 2 \quad \times \times$$

$$9 \rightarrow N \% 9 \Rightarrow 10 \% 9 = 1 \quad \times \times$$

$$10 \rightarrow N \% 10 \Rightarrow 10 \% 10 = 0 \quad \checkmark$$



// Pseudo Code

```
ρ ↳ v main () {  
    Scanner scn = new Scanner (System.in);  
    int n = scn.nextInt();
```

n iterations ↲

```
for (int i=1; i<=n; i++) {  
    if (n% i == 0) {  
        System.out.println(i);  
    }  
}
```



AlgoPrep

Tracing

```
for (int i=1; i<=n; i++) {  
    if (n% i == 0) {  
        System.out.println(i);  
    }  
}
```

1 iteration → 1 completion of loop

n=6

i	i<=n	n% i	i++
1	true	0	1 2
2	true	0	2 3
3	true	0	3 4
4	true	2	5
5	true	1	6
6	true	0	6 7
7	false		exit from loop

Break till 9:50 PM



(Q) IsPrime?

No divisible by
1 and itself

Given a number N , Print "Prime" if the number is a prime number else "Not Prime".

Ex: $N = 3 \rightarrow \{1, 3\} \rightarrow \text{Prime}$
 $N = 13 \rightarrow \{1, 13\} \rightarrow \text{Prime}$
 $N = 25 \rightarrow \{1, 5, 25\} \rightarrow \text{Not Prime}$

Quiz: $N = 1 \rightarrow \text{Neither Prime nor Composite.}$

Ideas

if factor Count == 2 → Prime
else → not Prime

Pseudo code

```

P ↳ r main () {
    Scanner scn = new Scanner (System.in);
    int n = scn.nextInt();
    int count = 0;
    for (int i = 1; i <= n; i++) {
        if (n % i == 0) {
            count = count + 1;
        }
    }
    if (count == 2) {
        System.out.println ("Prime");
    } else {
        System.out.println ("Not Prime");
    }
}

```



$N=5$

i	$i < n \wedge i \neq 0$	Count	$i + 1$
1	+ true	1	2
2	+ false	1	3
3	+ false	1	4
4	+ false	1	5
5	+ true	2	6
6	false	Leave from loop	

```
P &gt; r main () {  
    Scanner scn = new Scanner (System.in);  
    int n = scn.nextInt();  
    int Count = 0;  
    for (int i = 1; i <= n; i++) {  
        if (n % i == 0) {  
            Count = Count + 1;  
        }  
    }  
    if (Count == 2) { System.out.println ("Prime");}  
    else { System.out.println ("Not Prime");}  
}
```



AlgoPrep



1) break Statement

```
for(int i=1; i<4; i++) {  
    System.out.println(i);  
}
```

- 1
- 2
- 3

→ Jarak → the moment you execute break Statement
you exit the Current loop.

Ex :-

```
for(int i=1; i<10; i++) {  
    System.out.println(i);  
    if (i==2) { break; }  
}
```

- 1
- 2

i	i < 10	i == 2
1	true	false
2	true	true



Quiz 3:

```
for(int i=0; i<5; i++) {  
    if (i>2) {break;}  
    System.out.print(i + " ");
```

3

i	i < 5	if
0	+	b
1	+	b
2	+	b
3	+	t

0 1 2

Quiz 4:

```
for (int i=0; i<5; i++) {  
    break;  
    System.out.println(i);
```

3

i	i < 5	
0	+	-> terminate



// Continue Statement → Skip & go to next iteration.

```
for (int i=0; i<=5 ; i++) {  
    if (i==2) {  
        continue;  
    }  
    System.out.println(i);  
}
```

i	i <= 5	i = 2
0	+	↓
1	+	↓
2	+	↑
3	+	↓
4	+	↓
5	+	↓
6	→ exit	↓

0 1 3 4 5



AlgoPrep



Quiz 5:

```
for (int i=0; i<=5 ; i++) {
    if (i==2 || i==3) {
        continue;
    }
    System.out.print(i+"");
}
```

3

0 1 4 5

i	$i <= 5$	if
0	t	b
1	t	b
2	t	t
3	t	t
4	t	b
5	t	b
6		

6 ↳ exit

Quiz 6:

```
for (int i=0; i<=5 ; i++) {
    if (i==2 && i==3) {
        continue;
    }
    System.out.print(i+"");
}
```

3

0 1 2 3 4 5

i	$i <= 5$	if
0	t	b
1	t	b
2	t	b
3	t	b
4	t	b
5	t	b
6	b	b

6 ↳ exit



Q) Paint all "*" in a single row.



AlgoPrep



Q) Given integer n , print square of $n \times n$ using
"*".

$n=3$

```
***  
* * *  
* * *
```

$n=5$

```
*****  
* * * * *  
* * * * *  
* * * * *  
* * * * *
```



AlgoPrep



a) Pattern 1:

↳ Point the following pattern.

$N = 2 :$ *

* *

$N = 4 :$ *

* *

* * *

* * * *



AlgoPrep



Q) Pattern 2:

↳ Point the following Pattern.

$N=3 :$ 1

 2 3

 4 5 6

$N=4 :$ 1

 2 3

 4 5 6

 7 8 9 10



AlgoPrep



Today's agenda

↳ Patterns → {nested loop}



AlgoPrep



Q) Print N "*" in a single row.

Ex: $N = 4 \rightarrow * * * *$

```
for (int i=1; i<=N; i++) {  
    System.out.print("*");  
}
```



AlgoPrep



Q) Given integer n , Print square of $n \times n$ using "*".

$n = 3$

```
* * *
* * *
* * *
```

$n = 5$

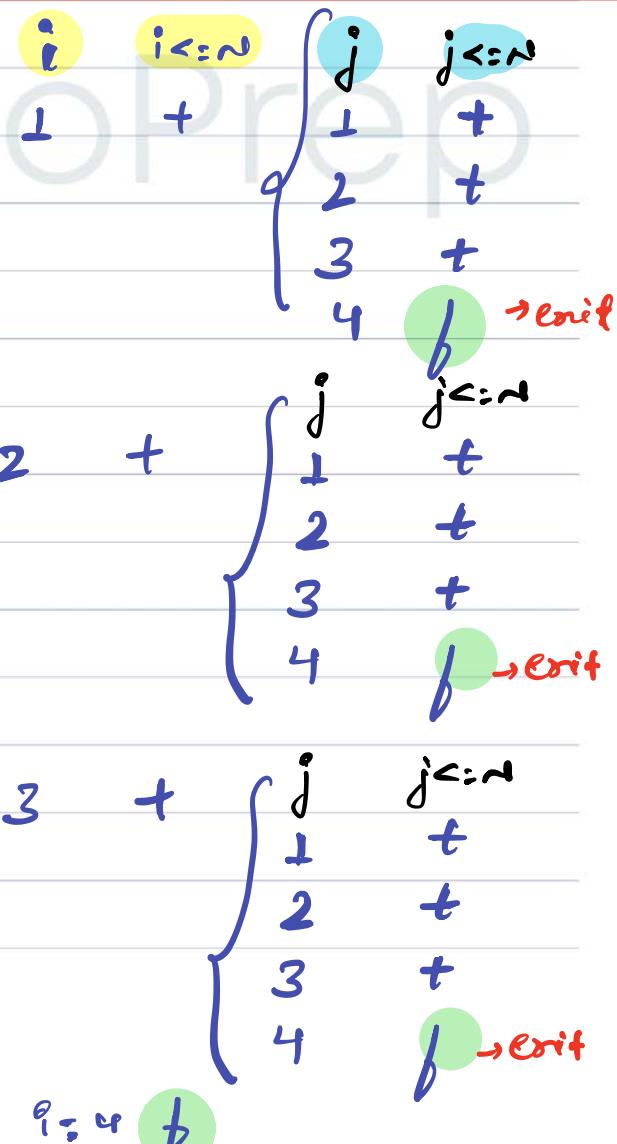
```
* * * *
* * * *
* * * *
* * * *
* * * *
```

$n = 3$

```
for (int i = 1; i <= n; i++) {
    for (int j = 1; j <= n; j++) {
        System.out.print("*");
    }
    System.out.println();
}
```

Output

```
* * *
* * *
* * *
```





Q) Pattern 2:

↳ Point the following Pattern.

$N=3 :$

1		
2	3	
4	5	6

$N=4 :$

1			
2	3		
4	5	6	
7	8	9	10

```

N=4
int Count=1;
int nst=1;
for (int i=1; i<=n; i++){

```

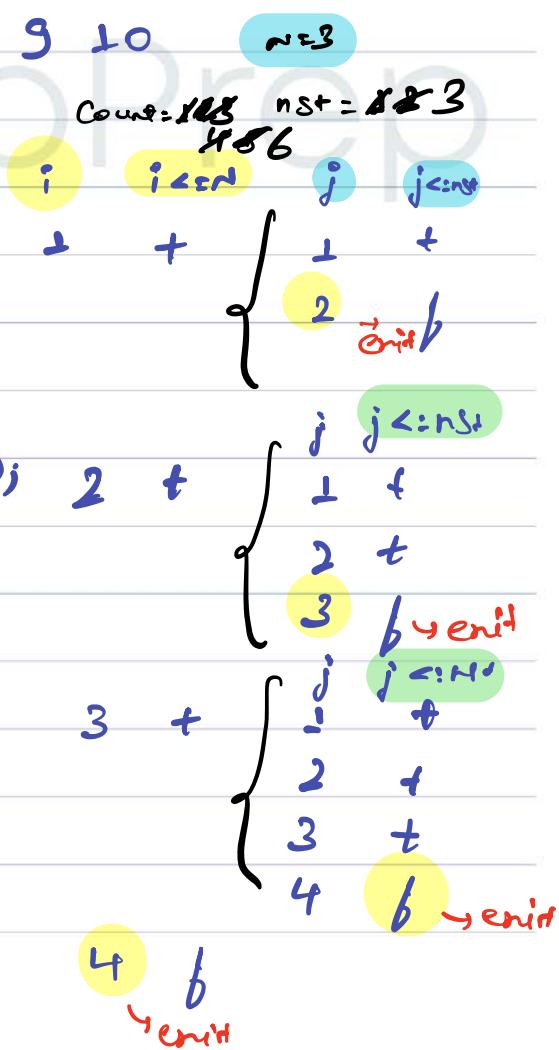
```
    for (int j=1; j<=nst; j++) {
```

```
        System.out.print(Count + " ");
        Count++;
    }
```

```
    nst++;
}
```

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

1
2 3
4 5 6



Break till 9:18 PM



Q) Pattern 3

↳ Point the following Pattern.

$$N=3 \vdash \quad - \quad * \quad nSP = \frac{3}{2} = 1 \vdash$$

* * *

*

$N = 5$: 
 $nSP = \frac{5}{2} = 2$

$$\begin{array}{ccccccc}
 \text{N : T} & & & * & 2+2 & & nSP = \frac{7}{2} = 3 \\
 - & - & - & & & & \\
 \downarrow 1 & & & & & & \\
 - & - & * & + & + & & \\
 \downarrow 1 & & & & & & \\
 - & + & + & + & + & + & + \\
 \downarrow 1 & & & & & & \\
 - & + & + & + & + & + & + \\
 \hline
 "f" & + & + & + & 2-2 & + & 2 \\
 & & & & & & \\
 - & - & * & + & 2-2 & + & \\
 & & & & & & \\
 - & - & - & + & 2-2 & & \\
 & & & & & &
 \end{array}$$

$N=6$: incorrect input



$N = 5$:

— — *
— * * *
* * * * *
— * * *
— — *

int NST = 1;
int NSP = 1/2; } think $\Delta \rightarrow$ is + row

for (int i=1; i<=N; i++) { ← no. of rows

 for (int j=1; j<=NST; j++) {

 System.out.print (" ");

 for (int k=1; k<=NSP; k++) {

 System.out.print ("*");

 if (i <= n/2) {

 NSP--;

 NST = NST + 2;

 }

 else {

 NSP++;

 NST = NST - 2;

point for
every row

→ for
next line

3

System.out.println();



N = 5:

```

      *   *
      *   *   *
      *   *   *   *
      *   *   *   *
      *   *
  
```

```

int nst = 1;
int nsp = 1/2; } think for 1st row

for (int i=1; i<=N; i++) { ← no. of rows
  
```

```

    for (int j=1; j<=nst; j++) {
      System.out.print(" ");
    }
    for (int k=1; k<=nst; k++) {
      System.out.print("#");
    }
    if (i<=n/2) {
      nsp--;
      nst = nst + 2;
    } else {
      nsp++;
      nst = nst - 2;
    }
    System.out.println();
  }
}
  
```

N=5

NST = 1 NSP: $\frac{5}{2} = 2$

i $i \leq \frac{n}{2}$
 ↓ → + → NST = 3
 → → NSP = 1

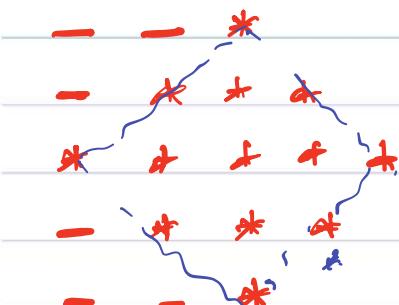
2 $2 \leq \frac{5}{2}$ → NST = 5
 ↓ f → NSP = 0

3 $3 \leq \frac{5}{2}$ → NST = 3
 ↓ b → NSP = 1

4 $4 \leq \frac{5}{2}$ → NST = 2
 ↓ b → NSP = 2

5 $5 \leq \frac{5}{2}$ → NST = 1
 ↓ b → NSP = 3

6 exit





Q) Pattern 4

↳ Point the following Pattern.

$N=5$:

* * * - * + +

* * - - - * +

* - - - - - +

* * - - - * *

* * + - * * *

$$\frac{n+2}{2} \Rightarrow \frac{n}{2} + 1$$

$N=7$:

1 * * * + + +
 2 * + + + - - -
 3 * + + + - - -
 4 * + + + - - -

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

5 * + + + - - -
 6 * + + + - - -
 7 * + + + - - -



II Pseudo Code

int nsp = 1;

int nsr = n/2 + 1

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

 for (int j=1; j<=nsr; j++) {

 System.out.print ("*");

 }

 for (int k=1; k<=nsp; k++) {

 System.out.print (" ");

 }

 for (int l=1; l<=nsr; l++) {

 System.out.print ("*");

 }

 if (i<=n/2) {

 nsr -= 1;

 nsp = nsp + 2;

 }

 else {

 nsr += 1;

 nsp = nsp - 2;



System.out.println();

* * * - * * *

* * - - - * * N=5 :

* * - - - * *

* - - - - *

* * - - - * *

* * * - * * *

//Pseudo code

```
int nsp = 2;
int nsf = n/2 + 1;
```

```
for (int i=1; i<=N; i++) {
```

```
    for (int j=i; j<=nsf; j++) {
        System.out.print("*");
    }
}
```

```
for (int k=1; k<=nsp; k++) {
```

```
    System.out.print(" ");
}
```

```
    for (int l=i; l<=nsf; l++) {
        System.out.print("*");
    }
}
```

```
    if (i<=n/2) {
        nsf--;
        nsp = nsp+2;
    }
}
```

```
    else {
        nsf++;
        nsp = nsp-2;
    }
}
```

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

nsf = 3

nsp = 1

1

nsf = 2

2

nsp = 3

3

nsf = 1

4

nsp = 5

5

nsf = 2

6

nsp = 3

7

nsf = 3

8

nsp = 1

9

nsf = 4

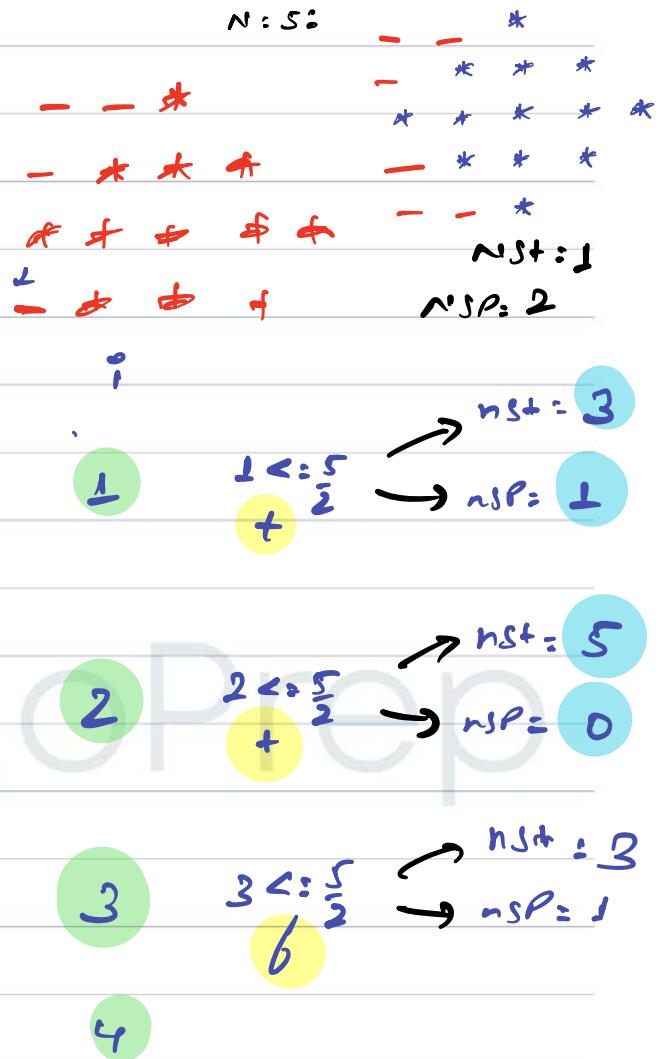
6 → exit



```

int nst = 1;
int nsp = 1/2; } think b is 1st row
for (int i=1; i<=N; i++) { ← no of rows
    for (int j=1; j<=nst; j++) {
        System.out.print(" ");
    }
    for (int k=1; k<=nsp; k++) {
        System.out.print("#");
    }
    if (i<=n/2) {
        nsp--;
        nst = nst + 2;
    } else {
        nsp++;
        nst = nst - 2;
    }
    System.out.println();
}
    
```

point for every row



Pattern 1

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int nst = 1;

        for(int i=1;i<=n;i++){
            for(int j=1;j<=nst;j++){
                System.out.print("* ");
            }
            nst++;
            System.out.println();
        }
    }
}
```

C++ Code:

```
#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
using namespace std;

int main() {
    int n;
    cin >> n;
    int nst = 1;

    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= nst; j++) {
            cout << "* ";
        }
        nst++;
        cout << endl;
    }

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    nst = 1

    for i in range(1, n + 1):
        for j in range(1, nst + 1):
            print("* ", end="")

        nst += 1
        print()

if __name__ == "__main__":
    main()
```

Pattern 2

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int nst = 1;

        int count = 1;
        for(int i=1;i<=n;i++){

            for(int j=1;j<=nst;j++){
                System.out.print(count+" ");
                count++;
            }

            nst++;
            System.out.println();
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;
    int nst = 1;
    int count = 1;

    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= nst; j++) {
            cout << count << " ";
            count++;
        }
        nst++;
        cout << endl;
    }

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    nst = 1
    count = 1

    for i in range(1, n + 1):
        for j in range(1, nst + 1):
            print(count, end=" ")
            count += 1

        nst += 1
        print()

if __name__ == "__main__":
    main()
```

Pattern 3

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        int nspaces = n / 2;
        int nstars = 1;
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= nspaces; j++) {
                System.out.print(" ");
            }

            for (int j = 1; j <= nstars; j++) {
                System.out.print("*");
            }

            if (i <= n / 2) {
                nspaces--;
                nstars += 2;
            } else {
                nspaces++;
                nstars -= 2;
            }

            System.out.println();
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int nspaces = n / 2;
    int nstars = 1;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= nspaces; j++) {
            cout << " ";
        }

        for (int j = 1; j <= nstars; j++) {
            cout << "*";
        }

        if (i <= n / 2) {
            nspaces--;
            nstars += 2;
        } else {
            nspaces++;
            nstars -= 2;
        }

        cout << endl;
    }

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    nspaces = n // 2
    nstars = 1
    for i in range(1, n + 1):
        for j in range(1, nspaces + 1):
            print(" ", end="")
        for j in range(1, nstars + 1):
            print("*", end="")
        if i <= n // 2:
            nspaces -= 1
            nstars += 2
        else:
            nspaces += 1
            nstars -= 2
        print()

if __name__ == "__main__":
    main()
```

Pattern 4

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        int nstars = n / 2 + 1;
        int nspaces = 1;
        for (int i = 1; i <= n; i++) {
            for (int j = 1; j <= nstars; j++) {
                System.out.print("*");
            }

            for (int j = 1; j <= nspaces; j++) {
                System.out.print(" ");
            }

            for (int j = 1; j <= nstars; j++) {
                System.out.print("*");
            }

            if (i <= n / 2) {
                nspaces += 2;
                nstars--;
            } else {
                nspaces -= 2;
                nstars++;
            }
        }

        System.out.println();
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int nstars = n / 2 + 1;
    int nspaces = 1;
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= nstars; j++) {
            cout << "*";
        }

        for (int j = 1; j <= nspaces; j++) {
            cout << " ";
        }

        for (int j = 1; j <= nstars; j++) {
            cout << "*";
        }

        if (i <= n / 2) {
            nspaces += 2;
            nstars--;
        } else {
            nspaces -= 2;
            nstars++;
        }
    }

    cout << endl;
}

return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    nstars = n // 2 + 1
    nspaces = 1
    for i in range(1, n + 1):
        print("*" * nstars, end="")
        print(" " * nspaces, end="")
        print("*" * nstars)

        if i <= n // 2:
            nspaces += 2
            nstars -= 1
        else:
            nspaces -= 2
            nstars += 1

if __name__ == "__main__":
    main()
```

Pattern 5_HW

Solution Video: <https://youtu.be/u1cYgwlc62E>

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        int nst = 1;
        int nsp = n / 2;

        int val = 1;
        for(int i=1;i<=n;i++){
            for (int j = 0; j < nsp; j++) {
                System.out.print(" ");
            }
            if (i <= n / 2) {
                val = i;
            } else {
                val = n + 1 - i;
            }
            for (int j = 0; j < nst; j++) {
                System.out.print(val + " ");

                if (j < nst / 2) {
                    val++;
                } else {
                    val--;
                }
            }
            if (i <= n / 2) {
```

```

        nsp--;
        nst = nst + 2;
    } else {
        nsp++;
        nst = nst - 2;
    }
    System.out.println();
}

}
}

```

C++ Code:

```

#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int nst = 1;
    int nsp = n / 2;

    int val = 1;
    for (int i = 1; i <= n; i++) {
        for (int j = 0; j < nsp; j++) {
            cout << " ";
        }
        if (i <= n / 2) {
            val = i;
        } else {
            val = n + 1 - i;
        }
        for (int j = 0; j < nst; j++) {
            cout << val << " ";
            if (j < nst / 2) {
                val++;
            }
        }
    }
}

```

```

    } else {
        val--;
    }
}

if (i <= n / 2) {
    nsp--;
    nst = nst + 2;
} else {
    nsp++;
    nst = nst - 2;
}
cout << endl;
}

return 0;
}

```

Python Code:

```

def main():
    n = int(input())

    nst = 1
    nsp = n // 2

    val = 1
    for i in range(1, n + 1):
        print(" " * nsp, end="")

        if i <= n // 2:
            val = i
        else:
            val = n + 1 - i

    for j in range(nst):
        print(val, end=" ")

        if j < nst // 2:
            val += 1
        else:
            val -= 1

```

```
if i <= n // 2:  
    nsp -= 1  
    nst += 2  
else:  
    nsp += 1  
    nst -= 2  
  
print()  
  
if __name__ == "__main__":  
    main()
```



Today's agenda

↳ factorial

↳ nC_r & nPr

↳ functions

→ Zoom Class because "Nishant bhaiya" will join at the end

↳ Start at 8:05



→ H.W → next class →
 ↳ codes
 ↳ recording

Assign class problems →

H.W

+ pen paper code

48 hours on your own

↓
Practice for you

↓
Codes / recordings



Q) factorial

Given n , Print factorial of n .

Quiz: $\text{fact}(4) = 1 * 2 * 3 * 4 = 24$

$\text{fact}(n) = 1 * 2 * 3 * 4 \dots \dots * n$

IIIP pseudo code

```
public static void main() {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int ans = 1;
```

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

```
System.out.println(ans);
```

3



$n = 4$

```
public static void main() {  
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();
```

int ans = 1;

```
for (int i = 1; i <= n; i++) {  
    ans = ans * i;
```

System.out.println(ans);

3

4

5

6

ans: ~~1 2 3 4~~ 24
 $i \leq 4$

i $i \leq n$

1 t

2 t

3 t

4 +

5 ~~b~~
6 ~~exit~~





// ${}^nC_\sigma$ and ${}^nP_\sigma$

Quiz 2: ${}^5C_3 \rightarrow \frac{15}{2!} = \frac{120}{2 \cdot 1 \cdot 2} = 10$ | ${}^nC_\sigma = \frac{Ln}{Ln \cdot L\sigma}$

Quiz 3: ${}^5P_3 \rightarrow \frac{15}{2!} = \frac{120}{2} = 60$ | ${}^nP_\sigma = \frac{Ln}{Ln \cdot \sigma}$

Q) Given n and σ , write an algorithm to calculate ${}^nC_\sigma$.



$$\frac{Ln}{Ln \cdot L\sigma}$$



11 Pseudo code

```
public static void main() {  
    Scanner scn = new Scanner(System.in);  
    int n = scn.nextInt();  
    int r = scn.nextInt();
```

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

```
    int rfact = 1;  
    for (int i = 1; i <= r; i++) {  
        rfact = rfact * i;  
    }
```

```
    int nmrfact = 1;  
    for (int i = 1; i <= n - r; i++) {  
        nmrfact = nmrfact * i;  
    }
```

int ans = $\frac{nfact}{(rfact * nmrfact)}$;

}



→ DRY → Do not Repeat yourself.

↓
function/method

- ↳ buy Screwdriver → Open first nut & bolt.
- ↳ use the Screwdriver → open second nut & bolt.
Took 2 hours ago.

idea of function

box

It calculates the
factorial.

input
 $n=3$
 $n=4$
 $n=5$

output

Syntax:

Public static int name (input) {

output
type

function
name

input

1 Statement 1

11 Statement 2

}



II Pseudo code

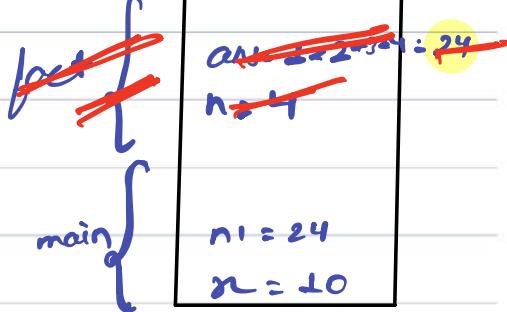
call stack

```
main() {  
    int n = 10; 24  
    int m = fact(4); ←  
    ↳ System.out.println(m); → 24
```

→ Public static int fact (int n){
 int ans = 1;

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

 ↳ return ans;



→ Called fact() but there is no function named fact() in the code. → error

→ Break till 9:35 pm

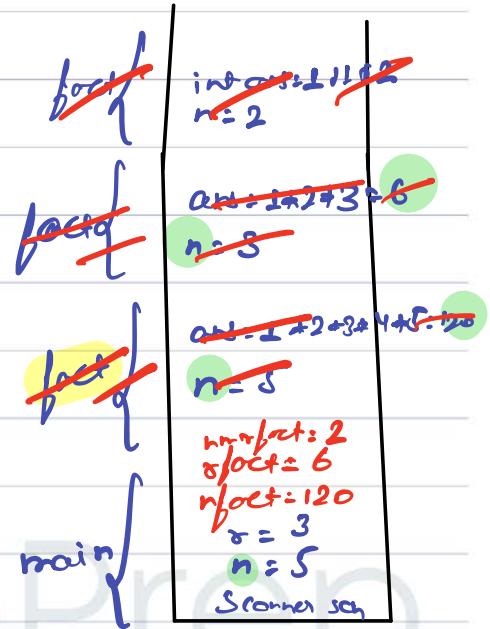


nC_5

1 Public static void main() {

```
Scanner scn = new Scanner(System.in);
int n = scn.nextInt();
int r = scn.nextInt();
int nfact = fact(n);
int rfact = fact(r);
int nmrfact = fact(n-r);
```

2 int ans = nfact / (nfact + nmrfact);
System.out.println(ans);



$$120 / (6+2) = 120 / 12 = 10$$

3 Public static int fact(int n){

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

return ans;

Q)

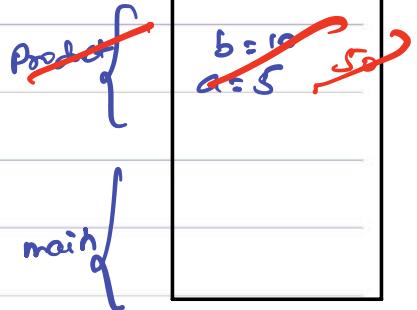
main() {
int ans = sum(10, 20, 30);
}
Public static void sum (int a, int b, int c){
 → no output

3



Quiz 24:

```
Public static void main(String[] args){  
    Product(5, 10);
```



```
Public static int Product(int a, int b){  
    return a * b;
```

3

↓
no output



Quiz 25:

```
Public static void main(String[] args){  
    int ans = Subtract(5, 10);  
    System.out.println(ans);
```

→ error

```
Public static void Subtract(int a, int b){  
    return a - b;
```

3



Quiz 6:

```
Public static void main (String [] args) {
```

```
    int n1 = cube (3);  
    ↳ System.out.println (add (n1, cube (2)));
```

$b = 8$
 $a = 27$, 35

```
    } Public static int add (int a, int b) {
```

```
        → return a+b;
```

~~int n1 = 27~~

~~int a = 35~~

```
    } Public static int cube (int a) {
```

```
        return a*a*a;
```

~~Cube {~~

~~n1 = 27~~

↳ 35

AlgoPrep_Found_Functions_Fact,ncr,npr

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

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

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

    public static int fact(int n){
        int ans = 1;

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

        return ans;
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int fact(int n) {
    int ans = 1;

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

    return ans;
}

int main() {
    int n, r;
    cin >> n >> r;

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

    int npr = nfact / nmrfact;
    int ncr = nfact / (rfact * nmrfact);
    cout << nfact << endl;
    cout << npr << endl;
    cout << ncr << endl;

    return 0;
}
```

Python Code:

```
def fact(n):
    ans = 1
    for i in range(1, n + 1):
        ans *= i
    return ans

def main():
    n, r = map(int, input().split())

    nfact = fact(n)
    rfact = fact(r)
    nmrfact = fact(n - r)

    npr = nfact // nmrfact
    ncr = nfact // (rfact * nmrfact)
    print(nfact)
    print(npr)
    print(ncr)

if __name__ == "__main__":
    main()
```



Today's agenda

- ↳ No. of factors
- ↳ Prime numbers
- ↳ Sum of N natural nos
- ↳ floor & ceil
- ↳ sqrt()



AlgoPrep



Q) Count no. of factors:

Given a number N , Point the Count of factors.

$$N = 24 \rightarrow \{1, 2, 3, 4, 6, 8, 12, 24\} \rightarrow 8$$

$$N = 36 \rightarrow \{1, 2, 3, 4, 6, 9, 12, 18, 36\} \rightarrow 9$$

$$1 \text{ sec} = 10^8 \text{ iterations.}$$

$\theta \leq r \text{ main } () \{$

Scanner scn = new Scanner (System.in);

int n = scn.nextInt();

int count = 0;

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

if ($n \% i == 0$) {

Count++;

return Count;

No. of iterations = N times



$$N = 10^9$$

10^9 iterations

→ How many secs will it take?

$$1 \text{ sec} = 10^8 \text{ iteration}$$

$$\frac{1}{10^8} \text{ secs} = 1 \text{ iteration}$$

$$\frac{1}{10^8} \times 10^9 = 10^9 \text{ iteration}$$

$$10^9 \text{ iterations} = 10 \text{ secs}$$

$$10^8 \text{ iteration} = 1 \text{ sec}$$

$$1 \text{ iteration} = \frac{1}{10^8} \text{ sec}$$

$$10^{18} \text{ iterations} = \frac{1}{10^8} * 10^{18} = 10^{10} \text{ sec}$$

$$10^{10} \text{ secs} = 317 \text{ years}$$

you → child → grandchild → 3rd → 4th → 7th gen.



1) Optimize

$$\hookrightarrow i \cdot j = N \Rightarrow j = N/i$$

$\rightarrow i$ is one factor \rightarrow other factor is N/i or j

$N = 24$		$N = 36$	
i	$j = N/i$	i	N/i
1	< 24	1	< 36
2	< 12 +2	2	< 18
3	< 8	3	< 12
4	< 6	4	< 9
6	> 4	6	=: 6
8	> 3	9	4
12	> 2	12	3
24	> 1	18	2
		36	1



||| Pseudo code

```
int countFactors (int n) {
    int count = 0;
    for (int i = 1; i <= n; i++) {
        if (n % i == 0) {
            if (i != n/i) { count = count + 2; }
            else { count = count + 1; }
        }
    }
    return count;
}
```



→ No. of iterations $\rightarrow \sqrt{n}$

$N = 10^{18} \rightarrow$ Seconds??

↳ 10^8 iterations $\rightarrow 1$ sec $\Rightarrow 1$ iteration $= \frac{1}{10^8}$ sec

↳ $\sqrt{10^{18}} = 10^9$ iterations

10^9 iterations $= \frac{1}{10^8}$ sec $\times 10^9 = 10$ sec.

→ 317 yrs \longrightarrow 10 sec



$$N=24 \rightarrow \sqrt{24} \rightarrow 4.9 \\ \rightarrow \text{Count} = 0 + 2 + 2 + 2 + 2$$

i	$i < \sqrt{n}$	$N \% i == 0$	
1	+	+	24
2	+	+	12
3	+	+	8
4	+	+	6
5	b		
		Count	

$$N=36$$

$$\text{Count} = 0 + 2 + 2 + 2 + 2$$

i	$i < \sqrt{n}$	$N \% i == 0$	N / i
1	+	+	36 + 2
2	+	+	18 + 2
3	+	+	12 + 2
4	+	+	9 + 2
5	+	b	
6	+	+	6 + 1

return count;



$$N = 36 \rightarrow \sqrt{N} = 6$$
$$\text{Count} = 0 + 2 + 2 + 2 + 2 + 1$$

```
int Count = 0;  
for (int i=1; i<=n; i++) {  
    if (N % i == 0) {  
        if (i != N/i) {Count = Count + 2}  
    } else {Count = Count + 1};  
}  
return Count;
```

i $i \leq \sqrt{N}$ $N \% i == 0$ N / i
1 + + 36
2 + + 18
3 + + 12
4 + + 9
5 + + 6
6 + + 6
7 + + 1 count



AlgoPrep



Q) Prime numbers

Given a number N , Check if the number is a prime no.

```
void Prime (int n) {
    int count = 0;
    for (int i = 1; i <= n; i++) {
        if (n % i == 0) {
            if (i != n / i) { count = count + 2 }
        }
        else { count = count + 1 }
    }
    if (count == 2) { S.O.P ("Prime"); }
    else { S.O.P ("Not Prime"); }
}
```



Quiz 1: Sum of all the numbers from 1 to 10.

$$\hookrightarrow \frac{10 \times 11}{2} = 55$$

Quiz 2: Sum of all the numbers from 1 to 1000.

$$\hookrightarrow \frac{1000 \times 1001}{2} = 500500$$

$\rightarrow 1, 2, 3, \dots, n$

Q) Sum of first n natural numbers.

whole no: 0, 1, 2, ...

→ Gauss (4th class)

$$S = 1 + 2 + 3 + \dots + 998 + 999 + 1000$$

$$S = 1000 + 999 + 998 + \dots + 3 + 2 + 1$$

$$2S = 1001 + 1001 + 1001 + \dots + 1000 + 1000 + 1000$$

$$2S = 1001 \times 1000$$

$$S = \frac{1001 \times 1000}{2} = 500500$$

II Sum of first n natural numbers.

$$S = 1 + 2 + 3 + \dots + n - 2 + n - 1 + n$$

$$S = n + (n-1) + (n-2) + \dots + 3 + 2 + 1$$

$$2S = (n+1) + (n+1) + (n+1) + \dots + (n+1) + (n+1) + (n+1)$$

$$2S = (n+1) \times n \Rightarrow S = \frac{n \times (n+1)}{2}$$



Quiz 3: Sum of 1^{st} n whole numbers.

$$0 + 1 + 2 + \dots + n-1$$

$$\begin{array}{c} \downarrow \\ 1 + 2 + \dots + n-1 \Rightarrow \frac{n(n-1)}{2} \end{array}$$

Sum of 1^{st} n whole numbers = Sum of 1^{st}
 $n-1$ natural
numbers

Sum of first 5 whole numbers?

$$0 + 1 + 2 + 3 + 4$$

Break till 9:30 pm



`//floor(num) → just smaller or equal integer`

Ex: $7.4 \rightarrow 7$

$8.9 \rightarrow 8$

$100.01 \rightarrow 100$

$90 \rightarrow 90$

$20.99 \rightarrow 20$

$3 \rightarrow 3$

$\Rightarrow \text{Math.floor}(num) \rightarrow \Leftarrow$



AlgoPrep



ceil(num) → just greater or equal integer

- Ex: 7.4 → 8
8.9 → 9
100.01 → 101
90 → 90
20.99 → 21
3 → 3



AlgoPrep



Q) Given N , return $\lfloor \text{sqrt}(N) \rfloor$.

$$\text{ex: } N=60 \rightarrow 7.0 \rightarrow 7$$

$$N=31 \rightarrow 5.1 \rightarrow 5$$

$$N=29 \rightarrow 5.2 \rightarrow 5$$

$$N=16 \rightarrow 4.0 \rightarrow 4$$

```
int sqrt(int n) {  
    int ans = 1;  
    int i = 1;  
    while (i * i <= n) {  
        ans = i;  
        i++;  
    }  
    return ans;  
}
```

$$N=60$$

i	$i * i \leq N$	ans
1	t	1
2	$2+2 \leq 60$ t	2
3	$3+3 \leq 60$ +	3
4	$4+4 \leq 60$ t	4
5	t	5
6	f	6
7	$7+7 \leq 60$ t	7
8	$8+8 \leq 60$ f	x
..		



Tracing

$n = 24$

int spot(int n) {

int ans = 1;

int i = 1;

while (i * i <= n) {

ans = i;

i++;

}

return ans;

	i	$i * i \leq n$	ans
1	1	t	1
2	2	t	2
3	3	t	3
4	4	t	4
5	5	f	5
6	6	6	6





Q) Given an integer n , return the difference between the product of digits and sum of its digits.

$$n = 234 \rightarrow \text{Product of digits} : 2 \times 3 \times 4 = 24$$

$$\text{Sum of digits} : 2 + 3 + 4 = 9$$

↓

$$24 - 9 = 15$$

```
int Sum=0,  
int multiply=1;  
while (n > 0) {
```

```
    int lastdigit = n % 10;
```

```
    Sum = Sum + lastdigit;
```

```
    multiply = multiply * lastdigit;
```

```
n = n / 10;
```

$n = 289 \ 23 \ 20$

Sum: 0 multiply: 1

$n > 0$	lastdigit	Sum	multiply
+	4	4	4
+	3	7	12
+	2	9	24

b
6. exit

```
return multiply - Sum;
```

Number of Factors Optimal

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner scn = new Scanner(System.in);
        //System.out.println("enter a number");
        int n = scn.nextInt();
        int count = 0;
        for(int i =1;i*i<=n;i++){
            if(n%i==0){
                if(i != n/i){count = count+2;}
                else{count++;}
            }
        }
        System.out.println(count);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int count = 0;
    for (int i = 1; i * i <= n; i++) {
```

```

if (n % i == 0) {
    if (i != n / i) {
        count = count + 2;
    } else {
        count++;
    }
}

cout << count << endl;

return 0;
}

```

Python Code:

```

def main():
    n = int(input())

    count = 0
    for i in range(1, int(n**0.5) + 1):
        if n % i == 0:
            if i != n // i:
                count += 2
            else:
                count += 1

    print(count)

if __name__ == "__main__":
    main()

```

IsPrime Optimal

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner scn = new Scanner(System.in);
        //System.out.println("enter a number");
        int n = scn.nextInt();
        int count = 0;
        for(int i =1;i*i<=n;i++){
            if(n%i==0){
                if(i != n/i){count = count+2;}
                else{count++;}
            }
        }

        if(count == 2){
            System.out.println("Yay");
        }else{
            System.out.println("Nay");
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;
```

```

int main() {
    int n;
    cin >> n;

    int count = 0;
    for (int i = 1; i * i <= n; i++) {
        if (n % i == 0) {
            if (i != n / i) {
                count += 2;
            } else {
                count++;
            }
        }
    }

    if (count == 2) {
        cout << "Yay" << endl;
    } else {
        cout << "Nay" << endl;
    }
}

return 0;
}

```

Python Code:

```

def main():
    n = int(input())

    count = 0
    for i in range(1, int(n ** 0.5) + 1):
        if n % i == 0:
            if i != n // i:
                count += 2
            else:
                count += 1

    if count == 2:
        print("Yay")
    else:
        print("Nay")

```

```
if __name__ == "__main__":
    main()
```

Sum of Natural Numbers

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
        should be named Solution. */
        Scanner scn = new Scanner(System.in);
        int n= scn.nextInt();

        System.out.println(n*(n+1)/2);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    cout << n * (n + 1) / 2 << endl;

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    print(n * (n + 1) // 2)

if __name__ == "__main__":
    main()
```

Floor(SQRTN)

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
        should be named Solution. */
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int i = 1;
        int ans = 1;

        while(i*i<=n){
            ans = i;
            i++;
        }

        System.out.println(ans);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int i = 1;
    int ans = 1;

    while (i * i <= n) {
        ans = i;
        i++;
    }

    cout << ans << endl;

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    i = 1
    ans = 1

    while i * i <= n:
        ans = i
        i += 1

    print(ans)

if __name__ == "__main__":
    main()
```

Product and Sum of Digits

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        System.out.println(subtractProductAndSum(n));
    }

    public static int subtractProductAndSum(int n) {
        int sum = 0;
        int multiply = 1;

        while(n>0){
            int lastdigit = n%10;

            sum = sum + lastdigit;
            multiply = multiply*lastdigit;

            n = n/10;
        }

        return multiply - sum;
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;
```

```

int subtractProductAndSum(int n) {
    int sum = 0;
    int multiply = 1;

    while (n > 0) {
        int lastdigit = n % 10;

        sum = sum + lastdigit;
        multiply = multiply * lastdigit;

        n = n / 10;
    }

    return multiply - sum;
}

int main() {
    int n;
    cin >> n;

    cout << subtractProductAndSum(n) << endl;

    return 0;
}

```

Python Code:

```

def subtractProductAndSum(n):
    sum_ = 0
    multiply = 1

    while n > 0:
        lastdigit = n % 10

        sum_ = sum_ + lastdigit
        multiply = multiply * lastdigit

        n = n // 10

    return multiply - sum_

def main():

```

```
n = int(input())
print(subtractProductAndSum(n))

if __name__ == "__main__":
    main()
```

Fibonacci Number_HW

Solution Video:

<https://youtu.be/xpDqrTKmHdM>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        int a = 0;
        int b = 1;

        for(int i = 1;i<=n;i++){
            int c = a+b;
            a = b;
            b = c;
        }

        System.out.println(a);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int a = 0;
    int b = 1;

    for (int i = 1; i <= n; i++) {
        int c = a + b;
        a = b;
        b = c;
    }

    cout << a << endl;

    return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    a, b = 0, 1

    for i in range(1, n):
        c = a + b
        a = b
        b = c

    print(a)
```

```
if __name__ == "__main__":
    main()
```

Valid Perfect Squares_HW

Solution Video:

<https://youtu.be/Jhqw9lRv1Uc>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int num = scn.nextInt();
        int i = 1;
        int ans = 1;

        while(i*i<=num){
            ans = i;
            i++;
        }

        if(ans*ans == num){
            System.out.println(true);
        }else{
            System.out.println(false);
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int num;
    cin >> num;

    int i = 1;
    int ans = 1;

    while (i * i <= num) {
        ans = i;
        i++;
    }

    if (ans * ans == num) {
        cout << "true" << endl;
    } else {
        cout << "false" << endl;
    }
}

return 0;
}
```

Python Code:

```
def main():
    num = int(input())

    i = 1
    ans = 1

    while i * i <= num:
        ans = i
        i += 1

    if ans * ans == num:
        print(True)
```

```
else:  
    print(False)  
  
if __name__ == "__main__":  
    main()
```

Today's agenda

- ↳ Intro to arrays
 - ↳ Syntax
 - ↳ Storing values
 - ↳ Reading input
- ↳ Return sum of arr[] elements
- ↳ Return max of arr[] elements.
- ↳ Array with functions
- ↳ Swap 2 indices
- ↳ Reverse array
- ↳ 1 more Problem

// Intro to array

```
↳ int a = 1;  
    int b = 2;  
    int c = 3;  
    int d = 4;  
    ...  
    ...  
    ...  
    ...  
    int j = 10;
```

100 Students

||

to store 100 variables



Arrays

Arrays Syntax

```
type [] name = new type [size];
```

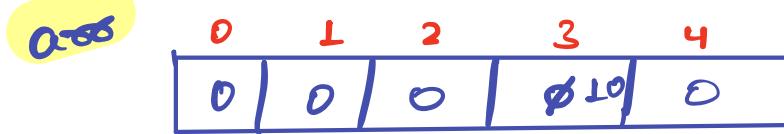
Q) Create an array of size 10 containing integers.

```
int [] arr = new int[10];
```

arr 0 ^{index of array} 1 2 3 4 5 6 7 8 9
 0 0 0 0 0 0 0 0 0 0
arr[2] arr[6]

// Indexing & Properties

↳ `int [] arr = new int[5];`



→ `System.out.println(arr[3]);` → 10

`arr[3] = 10;`

→ `System.out.println(arr[3]);` → 10

→ `System.out.println(arr[5]);` → Error (index out of bound)

→ If array is of length N.

1st index = 0;

last index = N-1;

Q) Create an array of length 5 with values 10 20 30 40 50.

Way 1:

Step 1: Create the array.

↳ `int [] arr = new int[5];`

0	1	2	3	4
0	0	0	0	b

Step 2: assign the values.

`arr[0] = 10;`

`arr[1] = 20;`

`arr[2] = 30;`

`arr[3] = 40;`

`arr[4] = 50;`

Way 2:

`int [] arr = {10, 20, 30, 40, 50};`

`System.out.println(arr[3]);` → 40

→ to get size of array: `arr.length;`

Q) Sum of array

↳ Read an array of n length and print the sum of all elements.

Ex: $\text{arr}[4]: \begin{matrix} 0 & 1 & 2 & 3 \\ 10 & -1 & 3 & -7 \end{matrix} \Rightarrow 5$

Java Pseudo Code

```
void main( ) {  
    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 sum=0;  
    for (int i=0; i<n; i++) {  
        sum = sum + arr[i];  
    }  
    S.O.P (sum);
```

Tracing

$\text{arr}[4]: \begin{matrix} 0 & 1 & 2 & 3 \\ 10 & -1 & 3 & -7 \end{matrix}$

```

int Sum=0;
for (int i=0; i<n; i++) {
    Sum = Sum + arr[i];
}
System.out.println(Sum);

```

i.	i < n	Sum
0	t	10
1	t	9
2	t	12
3	t	5
4	b	5

break

Q) Man of array elements

↳ Read an array of n length and Point the man of all elements.

Ex: $\text{arr}[4]: \begin{matrix} 0 & 1 & 2 & 3 \\ 10 & -1 & 3 & -7 \end{matrix} \rightarrow 10$

$\text{arr}[4]: \begin{matrix} -10 & -20 & -30 & -40 \end{matrix} \rightarrow -10$

IPseudo Code

```
void main( ) {  
    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 man = 0; \rightarrow man: $\text{arr}[0]$;
 \rightarrow man: $-\infty (\text{Integer. MIN_VALUE})$

```
for (int i=0; i<n; i++) {  
    if (arr[i] > man){  
        man = arr[i];  
    }  
    else if  
    }  $\rightarrow$  // nothing  
}  
s.o.p (man);
```

int man = 0;

arr[5]: 0 -1 2 3 -7 20

```
for (int i=0; i<n; i++) {
    if (arr[i] > man) {
        man = arr[i];
    } else
        ; // nothing
}
```

i	man = 0	i < n	arr[i] > man	man
0	t	t		10
1	t	f		10
2	t	f		10
3	t	f		10
4	t	f		20
5	b	b	exit	

man = - ∞

arr[4]: -10 -20 -30 -40

↳ 0

Break till 9:30 PM

Q) Swap the values of 2 variables

$$a = 10 \quad b = 20 \quad \Rightarrow \quad a = 20 \quad b = 10$$

//incorrect way

```
void main() {  
    int a = 10;  
    int b = 20;
```

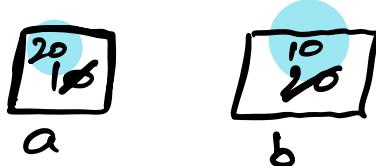


```
a = b;  
b = a;
```

3

//Correct way

```
void main() {  
    int a = 10;  
    int b = 20;
```



```
int temp = a;  
a = b;  
b = temp;
```

temp



3

// Arrays with functions

10 20

```
main( ) {  
    int a = 10;  
    int b = 20;  
    swap(a, b);  
    → S.O.P(a); → 10  
    → S.O.P(b); → 20  
}
```

~~swap~~ }
~~temp = 10~~
~~b = 20~~
~~a = 10~~
10 20

```
public static void swap(int a, int b) {  
    int temp = a;  
    a = b;  
    → b = temp;  
}
```

main }
b = 20
a = 10

→ Variables of 2 functions are not connected.

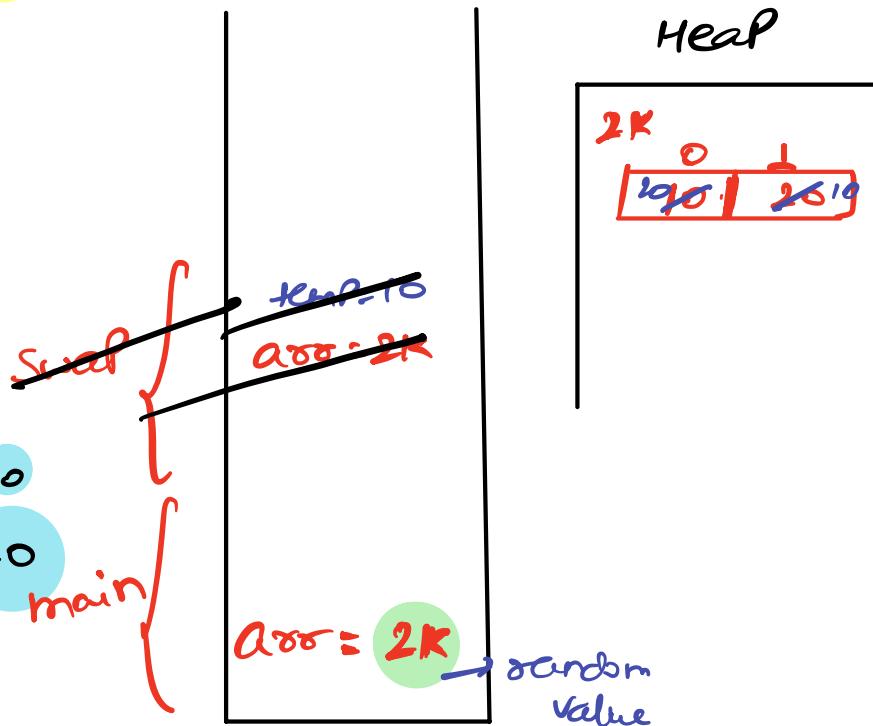
Q)

→ 20 10

```
void main() {  
    int[] arr = {10, 20};
```

Swap(arr);

↳ ↳ S.O.P (arr[0]); → 20
S.O.P (arr[1]); → 10



```
Public static void Swap (int[] arr){
```

```
    int temp = arr[0];  
    arr[0] = arr[1];  
    arr[1] = temp;
```

→ arrays across functions are always connected.

Q) Swap indexes

Given array of length N and two indexes $idn1$ and $idn2$, swap the element of those two indexes.

↳ done just before this Page

arr[] = { 10 20 30 40 50 }
 ^ idn1 ^ idn2
 0 1 2 3 4

$idn1 = 1$ $idn2 = 3$

```
int temp = arr[idn1];
arr[idn1] = arr[idn2];
arr[idn2] = temp;
```

Q) Reverse array

↳ Given array of length N , Reverse the whole array.

Ex: $\text{arr}[5]: \{10 \ 20 \ 30 \ 40 \ 50\}$

0 1 2 3 4
↓
50 40 30 20 10

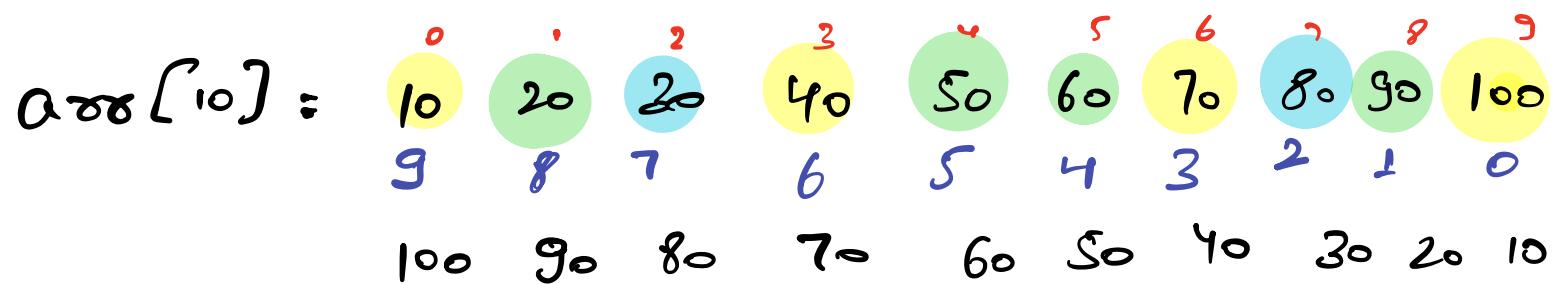
Combination of swaps = reverse

$\text{arr}[5]: \{10 \ 20 \ 30 \ 40 \ 50\}$

0 1 2 3 4
4 3 ↓ 1 0
↓
50 40 30 20 10

Swap (0, 4)

Swap (1, 3)



$\left. \begin{matrix} \text{Swap}(0, 9) \\ \text{Swap}(1, 8) \\ \text{Swap}(2, 7) \\ \text{Swap}(3, 6) \\ \text{Swap}(4, 5) \end{matrix} \right\} = \text{reverse the array}$

// Pseudo Code

```

int main() {
    // arr input
    reverse(arr);
}

```

P S void reverse (int arr[]) {

int SP = 0

int EP = arr.length - 1;

Swap (0, 9)
Swap (1, 8)
Swap (2, 7)
Swap (3, 6)
Swap (4, 5)

while (SP < EP) {

int tempP = arr[SP];

arr[SP] = arr[EP];

arr[EP] = tempP;

SP++;

EP -= 1;

}

}

```

int SP = 0
int EP = arr.length - 1;

```

$\text{arr}[9] = \begin{smallmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 10 & 20 & 20 & 40 & 50 & 60 & 70 & 80 & 90 \\ 90 & 80 & 70 & 60 & 50 & 40 & 30 & 20 & 10 \end{smallmatrix}$

while ($SP < EP$) {

```

int tempP = arr[SP];
arr[SP] = arr[EP];
arr[EP] = tempP;
SP++;
EP--;

```

}

SP	EP	$SP < EP$
0	8	t
1	7	t
2	6	t
3	5	t
4	4	b

exit

```

int SP = 0
int EP = arr.length - 1;

```

$\text{arr}[6] = \begin{smallmatrix} 0 & 1 & 2 & 3 & 4 & 5 \\ 10 & 20 & 30 & 40 & 50 & 60 \\ 60 & 50 & 40 & 30 & 20 & 10 \end{smallmatrix}$

won't work
T

while ($SP < EP$) {

```

int tempP = arr[SP];
arr[SP] = arr[EP];
arr[EP] = tempP;
SP++;
EP--;

```

}

SP	EP	$SP < EP$	$SP != EP$
0	5	t	t
1	4	t	t
2	3	t	t
3	2	b	t

exit

Sum of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 sum = 0;
        for(int i=0;i<n;i++){
            sum = sum + arr[i];
        }

        System.out.println(sum);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int sum = 0;

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

    cout << sum << endl;

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    sum_ = 0
    for num in arr:
        sum_ += num

    print(sum_)

if __name__ == "__main__":
    main()
```

Max of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 max = Integer.MIN_VALUE;

        for(int i=0;i<n;i++){
            if(arr[i] > max){
                max = arr[i];
            }
        }

        System.out.println(max);
    }
}
```

C++ Code:

```
#include <iostream>
#include <limits>
using namespace std;

int main() {
    int n;
    cin >> n;
```

```

int* arr = new int[n];
for (int i = 0; i < n; i++) {
    cin >> arr[i];
}

int max = numeric_limits<int>::min();

for (int i = 0; i < n; i++) {
    if (arr[i] > max) {
        max = arr[i];
    }
}

cout << max << endl;

delete[] arr;
return 0;
}

```

Python Code:

```

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    max_ = float('-inf')

    for num in arr:
        if num > max_:
            max_ = num

    print(max_)

if __name__ == "__main__":
    main()

```

Swap Indexes

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 idx1 = scn.nextInt();
        int idx2 = scn.nextInt();

        int temp = arr[idx1];
        arr[idx1] = arr[idx2];
        arr[idx2] = temp;

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

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
```

```

}

int idx1, idx2;
cin >> idx1 >> idx2;

int temp = arr[idx1];
arr[idx1] = arr[idx2];
arr[idx2] = temp;

for (int i = 0; i < n; i++) {
    cout << arr[i] << " ";
}

delete[] arr;
return 0;
}

```

Python Code:

```

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    idx1, idx2 = map(int, input().split())

    arr[idx1], arr[idx2] = arr[idx2], arr[idx1]

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()

```

Reverse Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 s = 0;
        int e = n-1;
        while(s < e){
            int temp = arr[s];
            arr[s] = arr[e];
            arr[e] = temp;
            s++;
            e--;
        }

        for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }

    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int s = 0;
    int e = n - 1;
    while (s < e) {
        int temp = arr[s];
        arr[s] = arr[e];
        arr[e] = temp;
        s++;
        e--;
    }

    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))

    s = 0
    e = n - 1
    while s < e:
        arr[s], arr[e] = arr[e], arr[s]
        s += 1
        e -= 1

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()
```

Largest Number at least twice HW

Solution Vid: https://youtu.be/_Dj2BNXTzCY

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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();
        }

        System.out.println(dominantIndex(arr));
    }

    public static int dominantIndex(int[] arr) {
```

```

int max = Integer.MIN_VALUE;
int index = -1;
int second = -1;
]
for (int i = 0; i < arr.length; i++) {
    if (arr[i] > max) {
        second = max;
        max = arr[i];
        index = i;
    } else if (arr[i] > second)
        second = arr[i];
}
return second * 2 <= max ? index : -1;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int dominantIndex(vector<int>& arr) {
    int max = INT_MIN;
    int index = -1;
    int second = -1;

    for (int i = 0; i < arr.size(); i++) {
        if (arr[i] > max) {
            second = max;
            max = arr[i];
            index = i;
        } else if (arr[i] > second)
            second = arr[i];
    }

    return second * 2 <= max ? index : -1;
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);

    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
}

```

```
}

cout << dominantIndex(arr) << endl;

return 0;
}
```

Python Code:

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))
    print(dominantIndex(arr))

def dominantIndex(arr):
    max_val = float('-inf')
    index = -1
    second = -1

    for i in range(len(arr)):
        if arr[i] > max_val:
            second = max_val
            max_val = arr[i]
            index = i
        elif arr[i] > second:
            second = arr[i]

    return index if second * 2 <= max_val else -1

if __name__ == "__main__":
    main()
```



Today's agenda

- ↳ Reverse a given Part of array
- ↳ Rotate array by K.
- ↳ greater than itself
- ↳ Two Sum



AlgoPrep



Q) Reverse a Part of array

Given n array element and $[s, e]$, reverse the array from $[s, e]$.

[3,7] Ex: $\text{arr}[] = \{ -3 \ 4 \ 2 \ 8 \ 3 \ 9 \ 6 \ 2 \ 8 \ 10 \}$

$\text{arr}[] = \{ -3 \ 4 \ 2 \ 8 \ 3 \ 9 \ 6 \ 2 \ 8 \ 10 \}$
 $s=3$ $e=7$

int $SP = s;$
int $EP = e;$



while ($SP < EP$) {

int $temp = arr[SP];$
 $arr[SP] = arr[EP];$
 $arr[EP] = temp;$
 $SP++;$
 $EP--;$

}

}



Q) Rotate the array

↳ Given N elements, Rotate array from last to first by K times. {google, meta, amazon}

$K=3$

ex: $\text{arr}[7]: \{3 \ -2 \ 1 \ 4 \ 6 \ 9 \ 8\}$

↓ 1st rot.

$\{8 \ 3 \ -2 \ 1 \ 4 \ 6 \ 9\}$

↓ 2nd rot.

$\{9 \ 8 \ 3 \ -2 \ 1 \ 4 \ 6\}$

↓ 3rd rot.

$\{6 \ 9 \ 8 \ 3 \ -2 \ 1 \ 4\}$



$K=3$

$\text{arr}[7]: \{3 \ -2 \ 1 \ 4 \ 6 \ 9 \ 8\}$



Reverse the whole array.

$\{8 \ 9 \ 6 \ | \ 4 \ 1 \ -2 \ 3\}$

↓ Reverse the first K elements

$\{6 \ 9 \ 8 \ | \ 4 \ 1 \ -2 \ 3\}$

↓ Reverse the elements after K elements

$\{6 \ 9 \ 8 \ | \ 3 \ -2 \ 1 \ 4\}$

$\{6 \ 9 \ 8 | 3 \ -2 \ 1 \ 4\}$



$k=6$
Q) arr[9]: $\{4 \ 1 \ 6 \ 9 \ 2 \ 14 \ 7 \ 8 \ 3\}$

reverse the whole array

$\{3 \ 8 \ 7 \ 14 \ 2 \ 9 | 6 \ 1 \ 4\}$

reverse the first k elements.

$\{9 \ 2 \ 14 \ 7 \ 8 \ 3 | 6 \ 1 \ 4\}$

↓
reverse the rem. elements.

$\{9 \ 2 \ 14 \ 7 \ 8 \ 3 \ 4 \ 1 \ 6\}$

$\{9 \ 2 \ 14 \ 7 \ 8 \ 3 \ 4 \ 1 \ 6\}$

$n = 10^{18}$

$k = 10^7 \rightarrow$

reverse
 $\frac{1}{3}$



// Pseudo Code

```
P S void main () {
    // input
    int n = --n
    int [] arr = new int [n];
    for (int i = 0; i < n; i++)
        arr[i] = --n
    int k = n - i;
    k = k * -1;
    reverse (arr, 0, n - 1);
    // Step 1: reverse whole array.
    // Step 2: reverse the first k elements.
    reverse (arr, 0, k - 1);
    // Step 3: reverse the elements after kth
    reverse (arr, k, n - 1);
}
```

```
P S void reverse (int arr[], int s, int e) {
    int sp = s;
    int ep = e;
```

```
    while (sp < ep) {
```

```
        int temp = arr[sp];
        arr[sp] = arr[ep];
        arr[ep] = temp;
        sp++;
        ep--;
    }
```

```
}
```



$K = 1000$

Q) arr[4]: $\{ \overset{0}{4} \underset{1}{1} \overset{2}{6} \underset{3}{9} \}$

$n=4$ $K=8$

$\{ \overset{0+1}{9} \underset{1}{4} \underset{2}{1} \underset{3}{6} \}$

$\{ \overset{0+2}{1} \underset{2}{6} \underset{3}{9} \underset{4}{1} \}$

$\{ \overset{0+3}{1} \underset{3}{6} \underset{4}{9} \underset{5}{4} \}$

$\{ \overset{0+4}{4} \underset{4}{1} \underset{5}{6} \underset{6}{9} \}$

OBS: :

→ you will get same array if you do rotation in multiples of arr.length.

n
5
5

K
50
45

→ Same array
→ Same array

5

52

Ultimately

$K \% n$

2 rotation $\rightarrow 52 \% 5$

$\% n \rightarrow \{0 \pm 2 \ n\}$

$\% 5 \rightarrow \{0 \ 1 \ 2 \ 3 \ 4\}$

$\Rightarrow K = K \% n \rightarrow$ this much rotation you have to do.



n (arr.length)

7

K=13

$$13 - 7 = 6 \Rightarrow 6 \text{ mod } 1$$

7

$$31 - 7 = 24 - 7 = 17 - 7 = 10 - 7 = 3$$

$$31 \% 7 = 3$$

8

34

$$\rightarrow 34 \% 8 = 2$$

$$K = K \% \underset{\uparrow}{\text{arr.length}}$$



8

AlgoPrep

6

4

7

$$\rightarrow 4 \% 8 = 4$$

$$\rightarrow 7 \% 6 = 1$$

Break till 9:55 PM



Q) Given n array elements, Count total no. of elements having atleast 1 element greater than itself.

ex: $\text{arr}[7]: \{ -4, -3, 7, 9, 3, 9, 4 \}$
↳ ans = 5

$\text{arr}[8]: \{ 3, 4, 11, 8, 2, 10, 9, 13 \}$
↳ ans = 6

$\text{arr}[5]: \{ 7, 7, 7, 7, 7 \}$
↳ ans = 0

OBS1: man elements of the array are invalid.

OBS2: except for man element, all the elements are valid.

II find the occ. of man element \rightarrow Count.

$$\text{ans} = n - \text{Count}$$



// Pseudo Code

```
int countgreater ( int arr[n]) {
```

```
    int man = Integer.min-value;
```

```
    for (int i=0; i<n; i++) {  
        if (arr[i] > man) {  
            man = arr[i];  
        }  
    }
```

```
    int Count = 0;
```

```
    for (int i=0; i<arr.length; i++) {  
        if (arr[i] == man) {  
            Count++;  
        }  
    }
```

```
    return arr.length - Count;
```



int man = Integer.min.VALUE;

```
for (int i=0; i<n; i++) {  
    if (arr[i]>man) {  
        man = arr[i];  
    }  
}
```

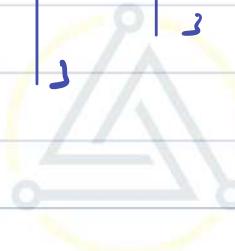
arr[7] = { 3 4 11 8 2 10 9 }

man = 10

Count = 1

ans = 7 - 1 = 6

```
int Count = 0;  
for (int i=0; i<arr.length; i++) {  
    if (arr[i] == man) {  
        Count++;  
    }  
}
```



AlgoPrep



Q) Two Sum

Given N array elements, Check if there exists a pair (i, j) such that $\text{arr}[i] + \text{arr}[j] = k$ and $i \neq j$.

Note: i & j are index value, k is given sum.

ex: $\text{arr}[7]: \{ 2^0 -1^1 0^2 3^3 4^4 5^5 7^6 \}$
 $K=8$ ↳ true

$\text{arr}[4]: \{ 1^0 3^1 -2^2 6^3 \}$
 $K=5$ ↳ false

$\text{arr}[5]: \{ 2^0 4^1 -3^2 7^3 10^4 \}$
 $K=8$ $\text{arr}[1] + \text{arr}[2] = 8$
 $4 + 4$ ↳ false

$\text{arr}[6]: \{ 3^0 5^1 1^2 8^3 3^4 7^5 \}$
 $K=6$ ↳ true



$\text{arr}[5] = \{3^0, 5^1, 2^2, 7^3, 5^4\}$

$k=12$

$i \downarrow$	$j \downarrow$
$0,0$	$1,0$
$0,1$	$1,1$
$0,2$	$1,2$
$0,3$	$1,3$
$0,4$	$1,4$
$2,0$	$3,0$
$2,1$	$3,1$
$2,2$	$3,2$
$2,3$	$3,3$
$2,4$	$3,4$
$4,0$	$4,1$
$4,1$	$4,2$
$4,2$	$4,3$
$4,3$	$4,4$

$n=5$

$i \rightarrow 0 \quad 1 \quad 2 \quad 3$
 $j \left(\begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix} \right) \quad j \left(\begin{matrix} 2 \\ 3 \\ 4 \end{matrix} \right) \quad j \left(\begin{matrix} 3 \\ 4 \end{matrix} \right) \quad j \left(\begin{matrix} 4 \end{matrix} \right)$

Public static boolean twoSum(int arr[], int k){

int n = arr.length; $i < n-1$

for (int i=0; $i < n-1$; i++) {

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

 if ($arr[i] + arr[j] == k$)

 return true;

}

3

2

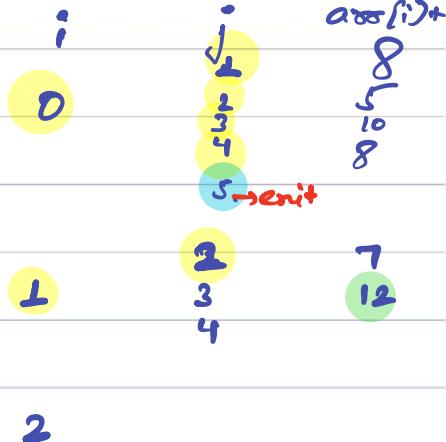
return false;



$n=5 \quad k=12$

```

public static boolean twoSum(int[] arr, int k) {
    int n = arr.length;
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] + arr[j] == k)
                return true;
        }
    }
}
  
```



3

3



$arr[5] = \{3, 5, 2, 7, 5\}$

i \ j				
0,0	1,0	2,0	3,0	4,0
0,1	1,1	2,1	3,1	4,1
0,2	1,2	2,2	3,2	4,2
0,3	1,3	2,3	3,3	4,3
0,4	1,4	2,4	3,4	4,4

$n=5$

Active learning

↓
H.W

Passive learning

Reverse Part of Array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 s = scn.nextInt();
        int e = scn.nextInt();

        reversePart(arr,s,e);

        for(int i=0;i<n;i++){
            System.out.print(arr[i]+" ");
        }
    }

    public static void reversePart(int[]arr, int s, int e){
        int sp = s;
        int ep = e;
        while(sp < ep){
            int temp = arr[sp];
            arr[sp] = arr[ep];
            arr[ep] = temp;
            sp++;
            ep--;
        }

    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

void reversePart(vector<int>& arr, int s, int e) {
    int sp = s;
    int ep = e;
    while (sp < ep) {
        int temp = arr[sp];
        arr[sp] = arr[ep];
        arr[ep] = temp;
        sp++;
        ep--;
    }
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    int s, e;
    cin >> s >> e;
    reversePart(arr, s, e);
    for (int i = 0; i < n; i++) {
        cout << arr[i] << " ";
    }
    return 0;
}
```

Python Code:

```
def reverse_part(arr, s, e):
    sp = s
    ep = e
    while sp < ep:
        arr[sp], arr[ep] = arr[ep], arr[sp]
        sp += 1
        ep -= 1
```

```

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    s, e = map(int, input().split())

    reverse_part(arr, s, e)

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()

```

Rotate Array

Java Code:

```

import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 k = scn.nextInt();

        k = k%n;
        reversePart(arr,0,n-1);
        reversePart(arr,0,k-1);
    }
}

```

```

reversePart(arr,k,n-1);

for(int i=0;i<n;i++){
    System.out.print(arr[i]+" ");
}
}

public static void reversePart(int[]arr, int s, int e){
    int sp = s;
    int ep = e;
    while(sp < ep){
        int temp = arr[sp];
        arr[sp] = arr[ep];
        arr[ep] = temp;
        sp++;
        ep--;
    }
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

void reversePart(vector<int>& arr, int s, int e) {
    int sp = s;
    int ep = e;
    while (sp < ep) {
        int temp = arr[sp];
        arr[sp] = arr[ep];
        arr[ep] = temp;
        sp++;
        ep--;
    }
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);

```

```

for (int i = 0; i < n; i++) {
    cin >> arr[i];
}
int k;
cin >> k;
k = k % n;
reversePart(arr, 0, n - 1);
reversePart(arr, 0, k - 1);
reversePart(arr, k, n - 1);
for (int i = 0; i < n; i++) {
    cout << arr[i] << " ";
}
return 0;
}

```

Python Code:

```

def reverse_part(arr, s, e):
    sp = s
    ep = e
    while sp < ep:
        arr[sp], arr[ep] = arr[ep], arr[sp]
        sp += 1
        ep -= 1

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    k = int(input())
    k = k % n

    reverse_part(arr, 0, n - 1)
    reverse_part(arr, 0, k - 1)
    reverse_part(arr, k, n - 1)

    for num in arr:
        print(num, end=" ")

if __name__ == "__main__":
    main()

```

Count Greater

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 max = Integer.MIN_VALUE;
        for(int i=0;i<n;i++){
            if(arr[i] > max){
                max = arr[i];
            }
        }

        int count = 0;
        for(int i=0;i<n;i++){
            if(arr[i] == max){
                count++;
            }
        }
    }
}
```

```
        System.out.println(n - count);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int max = INT_MIN;
    for (int i = 0; i < n; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }

    int count = 0;
    for (int i = 0; i < n; i++) {
        if (arr[i] == max) {
            count++;
        }
    }

    cout << n - count << endl;

    delete[] arr;
    return 0;
}
```

Python Code:

```
def main():
    n = int(input())

    arr = list(map(int, input().split()))

    max_val = float('-inf')
    for num in arr:
        if num > max_val:
            max_val = num

    count = 0
    for num in arr:
        if num == max_val:
            count += 1

    print(n - count)

if __name__ == "__main__":
    main()
```

Two Sum Brute

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        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 k = scn.nextInt();

        System.out.println(twosum(arr,k));
    }

    public static boolean twosum(int[] arr, int k){
        int n = arr.length;
        for(int i=0;i<n-1;i++){
            for(int j = i+1;j<n;j++){
                if(arr[i] + arr[j] == k){
                    return true;
                }
            }
        }

        return false;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

bool two_sum(const vector<int>& arr, int k) {
    int n = arr.size();
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (arr[i] + arr[j] == k) {
                return true;
            }
        }
    }
    return false;
}

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int k;
    cin >> k;

    cout << (two_sum(arr, k) ? "true" : "false") << endl;

    return 0;
}
```

Python Code:

```
def two_sum(arr, k):
    n = len(arr)
    for i in range(n - 1):
        for j in range(i + 1, n):
            if arr[i] + arr[j] == k:
                return True
    return False

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    k = int(input())

    print("true" if two_sum(arr, k) else "false")

if __name__ == "__main__":
    main()
```

Max Difference 1

Solution Vid: <https://youtu.be/vlbcMdMgY5o>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        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();
        }

        System.out.println(maxdifference_1(arr));

    }

public static int maxdifference_1(int[]arr){
    int max = Integer.MIN_VALUE;
    int min = Integer.MAX_VALUE;

    for(int i=0;i<arr.length;i++){
        if(arr[i] > max){
            max = arr[i];
        }

        if(arr[i] < min){
            min = arr[i];
        }
    }

    return max - min;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(const vector<int>& arr) {
    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }
}

```

```

        return max_val - min_val;
    }

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    cout << max_difference_1(arr) << endl;

    return 0;
}

```

Python Code:

```

def max_difference_1(arr):
    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()

```

Max Difference 2

Solution Vid: <https://youtu.be/njkBFHgz05Q>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        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();
        }

        System.out.println(maxdifference_1(arr));

    }

    public static int maxdifference_1(int[] arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] + i;
        }

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

        for(int i=0;i<arr.length;i++){
            if(arr[i] > max){
                max = arr[i];
            }

            if(arr[i] < min){
                min = arr[i];
            }
        }

        return max - min;
    }
}
```

```

    }

    return max - min;
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(vector<int>& arr) {
    int n = arr.size();

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

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    return max_val - min_val;
}

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
}

```

```

cout << max_difference_1(arr) << endl;
return 0;
}

```

Python Code:

```

def max_difference_1(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] + i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()

```

Max Difference 3

Solution Vid: <https://youtu.be/MUFdnVghGkY>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        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();
        }

        System.out.println(maxdifference_1(arr));

    }

    public static int maxdifference_1(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] - i;
        }

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

        for(int i=0;i<arr.length;i++){
            if(arr[i] > max){
                max = arr[i];
            }
        }
    }
}
```

```

        if(arr[i] < min){
            min = arr[i];
        }

    }

    return max - min;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int max_difference_1(vector<int>& arr) {
    int n = arr.size();

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

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

    return max_val - min_val;
}

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
}

```

```
}

cout << max_difference_1(arr) << endl;

return 0;
}
```

Python Code:

```
def max_difference_1(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] - i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    print(max_difference_1(arr))

if __name__ == "__main__":
    main()
```

Max Difference 4

Solution Vid: <https://youtu.be/SnzF5MAaO3w>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

// "static void main" must be defined in a public class.
public class Main {
    public static void main(String[] args) {

        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 ans1 = maxdifference_2(arr);
        int ans2 = maxdifference_3(arr);

        System.out.println(Math.max(ans1,ans2));
    }

    public static int maxdifference_3(int[]arr){
        for(int i=0;i<arr.length;i++){
            arr[i] = arr[i] - i;
        }

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

        for(int i=0;i<arr.length;i++){
            if(arr[i] > max){
                max = arr[i];
            }
        }

        if(arr[i] < min){
            min = arr[i];
        }

        return max - min;
    }
}
```

```

        min = arr[i];
    }

}

for(int i=0;i<arr.length;i++){
    arr[i] = arr[i] + i;
}
return max - min;
}

public static int maxdifference_2(int[]arr){
    for(int i=0;i<arr.length;i++){
        arr[i] = arr[i] + i;
    }

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

    for(int i=0;i<arr.length;i++){
        if(arr[i] > max){
            max = arr[i];
        }

        if(arr[i] < min){
            min = arr[i];
        }
    }

    for(int i=0;i<arr.length;i++){
        arr[i] = arr[i] - i;
    }

    return max - min;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int max_difference_2(vector<int>& arr) {
    for (int i = 0; i < arr.size(); i++) {

```

```

        arr[i] = arr[i] + i;
    }

int max_val = INT_MIN;
int min_val = INT_MAX;

for (int num : arr) {
    if (num > max_val) {
        max_val = num;
    }

    if (num < min_val) {
        min_val = num;
    }
}

for (int i = 0; i < arr.size(); i++) {
    arr[i] = arr[i] - i;
}

return max_val - min_val;
}

int max_difference_3(vector<int>& arr) {
    for (int i = 0; i < arr.size(); i++) {
        arr[i] = arr[i] - i;
    }

    int max_val = INT_MIN;
    int min_val = INT_MAX;

    for (int num : arr) {
        if (num > max_val) {
            max_val = num;
        }

        if (num < min_val) {
            min_val = num;
        }
    }

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

    return max_val - min_val;
}

```

```

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int max_val = INT_MIN;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            if (abs(arr[i] - arr[j]) + i - j > max_val) {
                max_val = abs(arr[i] - arr[j]) + i - j;
            }
        }
    }

    int ans1 = max_difference_2(arr);
    int ans2 = max_difference_3(arr);

    cout << max(max_val, max(ans1, ans2)) << endl;
}

return 0;
}

```

Python Code:

```

def max_difference_2(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] + i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    for i in range(len(arr)):
        arr[i] = arr[i] - i

```

```

    return max_val - min_val

def max_difference_3(arr):
    for i in range(len(arr)):
        arr[i] = arr[i] - i

    max_val = float('-inf')
    min_val = float('inf')

    for num in arr:
        if num > max_val:
            max_val = num

        if num < min_val:
            min_val = num

    for i in range(len(arr)):
        arr[i] = arr[i] + i

    return max_val - min_val

def main():
    n = int(input())
    arr = list(map(int, input().split()))

    max_val = float('-inf')
    for i in range(n):
        for j in range(n):
            if abs(arr[i] - arr[j]) + i - j > max_val:
                max_val = abs(arr[i] - arr[j]) + i - j

    ans1 = max_difference_2(arr.copy())
    ans2 = max_difference_3(arr.copy())

    print(max(max_val, max(ans1, ans2)))

if __name__ == "__main__":
    main()

```



Today's agenda

- ↳ Intro to 2D Arrays
- ↳ Point motion row wise
- ↳ Point motion Colwise
- ↳ Point motion in wave form



AlgoPrep



11 Intro to 2d array

↳

JEE Mains →

Physics

Chemistry

Maths

	0	1	2
0th Student	80	85	65
1st Student	80	85	65
2nd Student.	80	85	65

⋮
⋮
⋮

↳ Excel Sheet → 2d array use

11 Syntax

int arr = new int [5][4];

row
col

arr

Column				
0	1	2	3	
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

⇒ arr[3][2]

↳ $5 \times 4 = 20$ elements



Q) Print matrix downward

↳ Print the given mat[n][m] downward.

Ex: arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

10 20 30 40 50
 → 60 70 80 90 100
 110 120 130 140 150
 160 170 180 190 200

j\i	0,0	1,0	2,0	3,0
0,0	00	10	20	30
0,1	01	11	21	31
0,2	02	12	22	32
0,3	03	13	23	33
0,4	04	14	24	34

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

 for (int j=0; j<m; j++) {
 System.out.println(arr[i][j]);

}

}



	0				$m-1$
0					
.					
.					
.					
1					
.					
$n-1$					
					$n \times m$

```

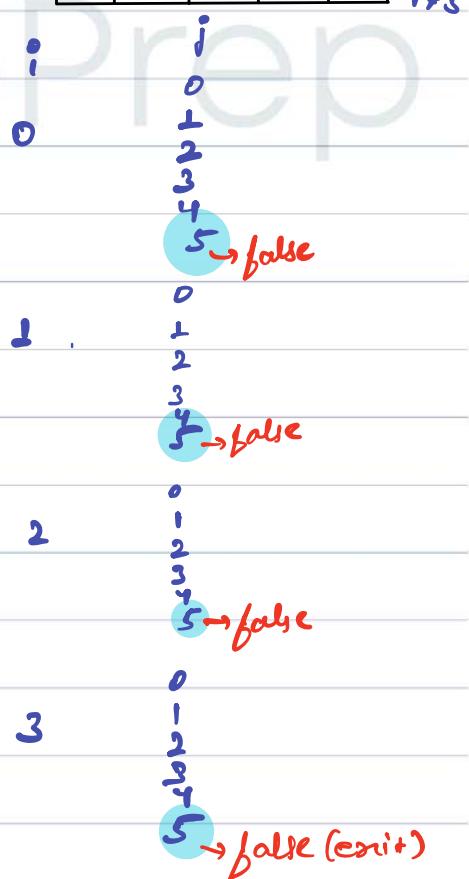
for (int i=0; i<n; i++) {
    for (int j=0; j<m; j++) {
        System.out.print(arr[i][j] + " ");
    }
    System.out.println();
}

```

10 20 30 40 50
60 70 80 90 100
110 120 130 140 150
160 170 180 190 200

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4x5



4 → false (exit)



Q) Print matrix Colwise

↳ Point the given matrix [n][m] col wise.

Ex: arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

10 60 110 160
20 70 120 170
30 80 130 180
40 90 140 190
50 100 150 200



0 1
1 1
2 1
3 1

0 2
1 2
2 2
3 2

0 3
1 3
2 3
3 3

0 4
1 4
2 4
3 4

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

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

 System.out.print(arr[i][j] + " ");

 }

}



```
for (int j=0; j<m; j++) {
```

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

```
        System.out.print(arr[i][j] + " ");
```

```
    }
```

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

```
}
```

$\text{arr}[4][5] = 0$

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4x5

10 - 60 - 110 - 160

20 - 70 - 120 - 170

j
0

i
0
1
2
3
4 → end

1
2

0
1
2
3
4 → end

Break till 9:20 Pm



Q) Print matrix in wave form

↳ Point the given mat[n][m] in wave form.

Ex: $\text{arr}[4][5] =$

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

n*m

L-R \rightarrow 10 20 30 40 50

R-L \rightarrow 100 90 80 70 60

L-R \rightarrow 110 120 130 140 150

R-L \rightarrow 200 190 180 170 160

L-R \rightarrow 0, 2, ... (even no.)

R-L \rightarrow 1, 3, ... (odd no.)

even no. that are divisible by 2

down-wise

```

for(int i=0; i<n; i++){
    if(i%2 == 0){
        for(int j=0; j<m; j++){
            System.out.print(arr[i][j]+");
        }
    } else {
        for(int j=m-1; j>=0; j--){
            System.out.print(arr[i][j]+");
        }
    }
}

```

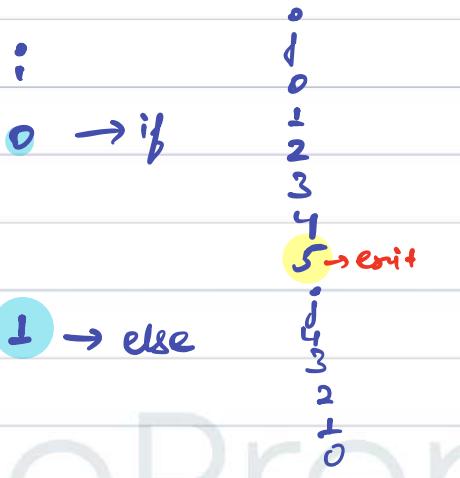


```
for(int i=0; i<n; i++) {  
    if(i%2 == 0) {  
        for (int j=0; j<m; j++) {  
            System.out.print(arr[i][j] + " ");  
        }  
    } else {  
        for (int j=m-1; j>=0; j--) {  
            System.out.print(arr[i][j] + " ");  
        }  
    }  
    System.out.println();  
}
```

arr[4][5] =

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

arr



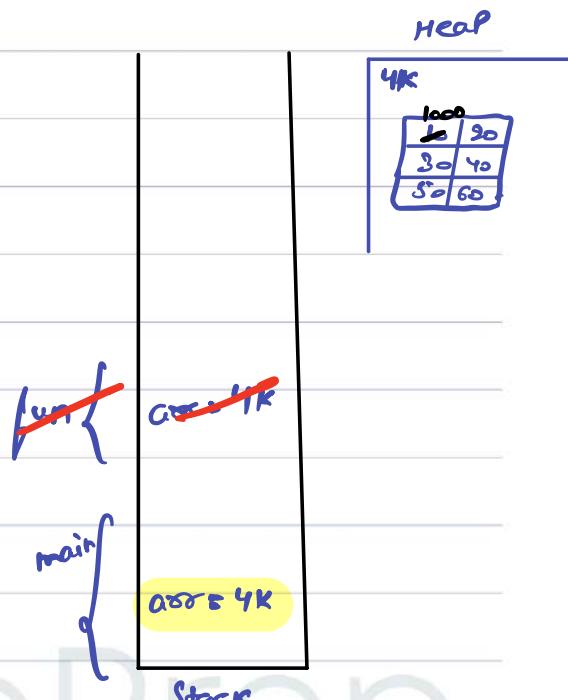
2

3



II Pseudo code

```
void main() {  
    // Input 2d array -> arr[2][3]  
    : {  
        10 20  
        30 40  
        50 60  
    }  
    S.O.P (arr[0][0]); → 10  
    fun(arr);  
}  
→ S.O.P (arr[0][0]);
```



```
public static void fun (int arr[]){  
    arr[0][0] = 1000;  
}
```

3

→ java is Pass by value only. you Pass the Stack value to the function.



Today's agenda

- ↳ Char and String
- ↳ ASCII
- ↳ Problems

↳ Sunday → optional class (Problem Solving) → 8PM
↳ Alpoor Kumar → 7* on Codechef
masters on Codeforces



AlgoPrep



String: Sequence of Characters
↳ Upper case, lower case, special etc.

String st = "AlgoBsp";

Characters: a) Alphabet

- ↳ a-z (lowercase)
- ↳ A-Z (uppercase)

b) Special characters

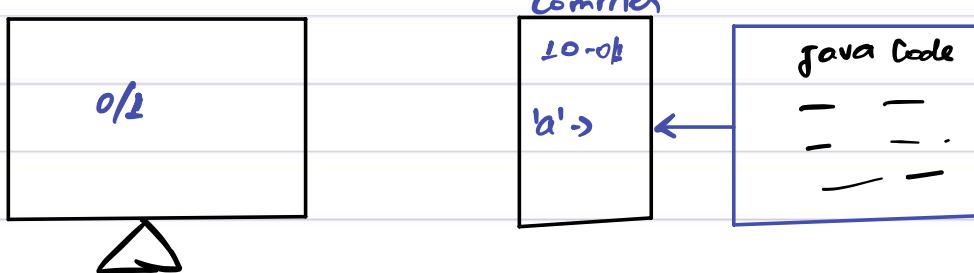
- ↳ @ # ? etc.

c) Numbers

- ↳ '0' - '9'

Syntax:

char ch = 'A';
↳ type, name, character



1. numbers → binary number
2. Characters → number → binary number
↳ Predefined



ASCII → 256 Characters

'A' : 65

'B' : 66

'C' : 67

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:

:</



* char rules

1. char to int: implicit. ex: int a = 'A'; ↗

2. int to char: Complicated. → (Typecast always)
↳ (implicity).

Char ch = (char)66;



AlgoPrep



Quiz 1 :

Char ch1 = 'B';
S.O.P(ch1);

Quiz 2 :

Char ch2 = 66; → typecast automatic → 'B'
S.O.P(ch2);

Quiz 3 :

Char ch3 = 'xyz';
S.O.P(ch3); → error

Quiz 4 :

int n = 'A'; → 65
n = n + 2; → n = 67
System.out.println(n); → 67



Quiz 5:

→ mathematical operation in character allowed
(ASCII)

Char ch4 = 'A';

ch4 = $\text{char}(65) + 3;$ $65 + 3 = 68$

s.o.p (ch4);

Implicit conversion.

Quiz 6:

Char ch5 = 'A';

if ($ch5 >= 90$) {

s.o.p ("Greater");

}

→ Smaller

else {

s.o.p ("Smaller");

}

Break till 9:45 PM

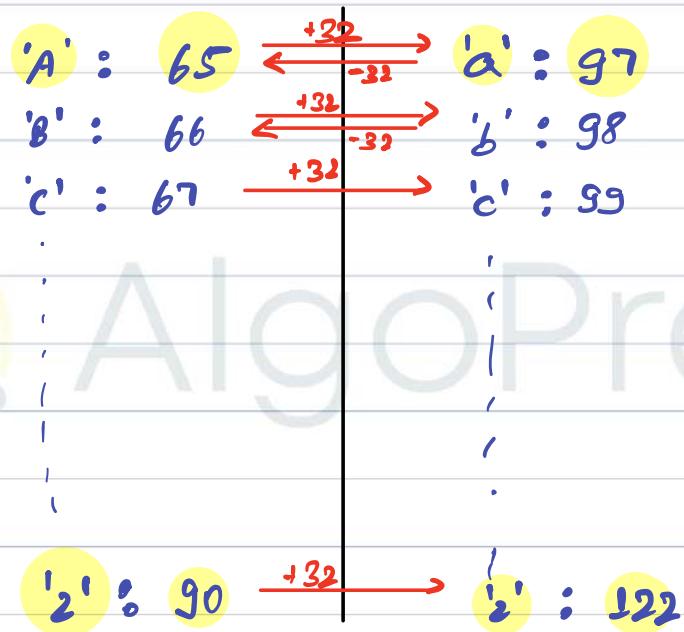


Q) Toggle Character

Given a `char[]` which contains only small and capital letters, toggle them.

↳ lowercase → uppercase
uppercase → lowercase

Ex: AlgoPrep : aLGHOpREP



uppercase to lowercase → +32

lowercase to uppercase → -32



II PSuedo Code

```
P S void main( ) {
    Scanner scn = new Scanner (System.in);
    int n = scn.nextInt();
    char[] ch = new char[n];
    String st = scn.nextLine();

    for (int i=0; i<n; i++) {
        ch[i] = st.charAt(i);
        Toggle (ch);
    }
}
```



AlgoPrep

```
P S void Toggle (char[] ch) {
    for (int i=0; i<ch.length; i++) {
        if (ch[i] >= 65 && ch[i] <= 90) { // uppercase
            ch[i] = (char)(ch[i] + 32);
        } else { // lowercase
            ch[i] = (char)(ch[i] - 32);
        }
    }
}
```



Q) Reverse the given string

Given a string str, reverse and print it.

Ex: algoPrep : prepoga

// Pseudo Code

```
P S String reverseString (String str){  
    int n = str.length();
```

```
    Char [ ] ch = str.toCharArray ();
```

```
    int SP = 0;
```

```
    int EP = n - 1;
```

```
    while (SP < EP) {
```

```
        char temp = ch[SP];
```

```
        ch[SP] = ch[EP];
```

```
        ch[EP] = temp;
```

```
        SP++; EP--;
```

```
String ans = str.valueOf (ch);
```

```
return ans;
```

}



String st = "Hello";

st = st + "e";

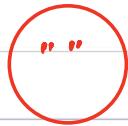
S-O-P (st); → Helloe

Char [] ch = { 'a', 'e', 'o', 'm' };

String ans = "";

i	ans
0	"a"
1	"ac"
2	"aeo"
3	"aeom"

```
for (int i=0; i<ch.length; i++) {  
    ans = ans + ch[i];  
}
```



Toggle Character

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        char[] ch = st.toCharArray();
        toggle(ch);
        for(int i=0;i<ch.length;i++){
            System.out.print(ch[i]);
        }
    }

    private static void toggle(char[]ch) {

        for(int i=0;i<ch.length;i++){
            if(ch[i] >= 65 && ch[i]<= 90){
                ch[i] = (char)(ch[i] + 32);
            }else{
                ch[i] = (char)(ch[i] - 32);
            }
        }
    }
}
```

C++ Code:

```
#include <iostream>
#include <cctype>
using namespace std;

void toggle(char *ch, int length) {
    for (int i = 0; i < length; i++) {
        if (isupper(ch[i])) {
```

```

        ch[i] = tolower(ch[i]);
    } else {
        ch[i] = toupper(ch[i]);
    }
}

int main() {
    string st;
    getline(cin, st);

    char ch[st.length() + 1];
    strcpy(ch, st.c_str());

    toggle(ch, st.length());

    for (int i = 0; i < st.length(); i++) {
        cout << ch[i];
    }

    return 0;
}

```

Python Code:

```

def toggle(ch):
    toggled = []
    for char in ch:
        if char.isupper():
            toggled.append(char.lower())
        else:
            toggled.append(char.upper())
    return ''.join(toggled)

def main():
    st = input()
    toggled_string = toggle(st)
    print(toggled_string)

if __name__ == "__main__":
    main()

```

Reverse String

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        char[] ch = st.toCharArray();
        reverse(ch);
        for(int i=0;i<ch.length;i++){
            System.out.print(ch[i]);
        }
    }

    private static void reverse(char[]ch) {

        int sp =0;
        int ep = ch.length - 1;

        while(sp < ep){
            char temp = ch[sp];
            ch[sp] = ch[ep];
            ch[ep] = temp;
            sp++;
            ep--;
        }
    }
}
```

C++ Code:

```
#include <iostream>
```

```

#include <cstring>
using namespace std;

void reverse(char *ch, int length) {
    int sp = 0;
    int ep = length - 1;

    while (sp < ep) {
        char temp = ch[sp];
        ch[sp] = ch[ep];
        ch[ep] = temp;
        sp++;
        ep--;
    }
}

int main() {
    string st;
    getline(cin, st);

    char ch[st.length() + 1];
    strcpy(ch, st.c_str());

    reverse(ch, st.length());

    for (int i = 0; i < st.length(); i++) {
        cout << ch[i];
    }

    return 0;
}

```

Python Code:

```

def reverse(ch):
    sp = 0
    ep = len(ch) - 1

    while sp < ep:
        ch[sp], ch[ep] = ch[ep], ch[sp]
        sp += 1
        ep -= 1

def main():
    st = input()

```

```

ch = list(st)
reverse(ch)
reversed_string = ".join(ch)
print(reversed_string)

if __name__ == "__main__":
    main()

```

Insert Difference_HW

Solution Vid:

https://youtu.be/6u99_gdqtOU

Java Code:

```

import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn=new Scanner(System.in);
        String st=scn.nextLine();
        System.out.println(insert(st));
    }

    private static String insert(String st) {
        String ans = "";

        for(int i=0;i<st.length()-1;i++){
            char ch1 = st.charAt(i);
            ans = ans + ch1;
            int temp = st.charAt(i+1)-st.charAt(i);
            ans = ans + temp;
        }

        ans = ans + st.charAt(st.length()-1);
        return ans;
    }
}

```

C++ Code:

```
#include <iostream>
#include <string>
using namespace std;

string insert(string st) {
    string ans = "";

    for (int i = 0; i < st.length() - 1; i++) {
        char ch1 = st[i];
        ans += ch1;
        int temp = st[i + 1] - st[i];
        ans += to_string(temp);
    }

    ans += st[st.length() - 1];
    return ans;
}

int main() {
    string st;
    getline(cin, st);

    string result = insert(st);
    cout << result << endl;

    return 0;
}
```

Python Code:

```
def insert(st):
    ans = ""

    for i in range(len(st) - 1):
        ch1 = st[i]
        ans += ch1
        temp = ord(st[i + 1]) - ord(st[i])
        ans += str(temp)

    ans += st[-1]
    return ans

def main():
```

```

st = input()
result = insert(st)
print(result)

if __name__ == "__main__":
    main()

```

Is Palindrome_HW

Solution Vid:

<https://youtu.be/QBsaSTN48kA>

Java Code:

```

import java.io.*;
import java.util.*;

public class Solution {

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

        char[] arr = new char[n];
        for(int i=0;i<n;i++){
            if(st.charAt(i)>='A' && st.charAt(i)<='Z'){
                arr[i] = (char)(st.charAt(i) + 32);
            }else{
                arr[i] = st.charAt(i);
            }
        }

        int sp = 0;
        int ep = n-1;

        while(sp<ep){
            if(arr[sp]!= arr[ep]){
                System.out.println("false");
                return;
            }
        }
    }
}

```

```

        sp++;
        ep--;
    }

    System.out.println("true");
}
}

```

C++ Code:

```

#include <iostream>
#include <cctype>
using namespace std;

int main() {
    string st;
    getline(cin, st);
    int n = st.length();

    char arr[n];
    for (int i = 0; i < n; i++) {
        if (isupper(st[i])) {
            arr[i] = tolower(st[i]);
        } else {
            arr[i] = st[i];
        }
    }

    int sp = 0;
    int ep = n - 1;

    while (sp < ep) {
        if (arr[sp] != arr[ep]) {
            cout << "false" << endl;
            return 0;
        }

        sp++;
        ep--;
    }

    cout << "true" << endl;
    return 0;
}

```

Python Code:

```
def main():
    st = input()
    n = len(st)

    arr = [0] * n
    for i in range(n):
        if st[i].isupper():
            arr[i] = st[i].lower()
        else:
            arr[i] = st[i]

    sp = 0
    ep = n - 1

    while sp < ep:
        if arr[sp] != arr[ep]:
            print("false")
            return

        sp += 1
        ep -= 1

    print("true")

if __name__ == "__main__":
    main()
```



Today's Content

↳ How to calculate No. of iteration

↳ Big O notation

Time complexity ! = Time taken to execute

↳ No. of iterations [functions of length of the input]
Rate of change of time of a function of input.

ASYMPTOTIC NOTATIONS :

- Big O O → worst case } mostly used.
- Omega Ω → Best case
- Theta Q → Average Case



Quiz 1: Sum of n natural numbers:

$$\rightarrow \frac{n(n+1)}{2}$$

Quiz 2: $[3, 10] \rightarrow 8$

$\lceil \rightarrow$ inclusive	$[a, b]$	$[a, b)$	(a, b)
$(\rightarrow$ exclusive	$b - a + 1$	$b - a$	$b - a - 1$

Quiz 3: $N \rightarrow \frac{N}{2} \rightarrow \frac{N}{4} \rightarrow \frac{N}{8} \dots 1$

$$\left[\left[\left[\left[\left[\left(N \right) \frac{1}{2} \right] \times \frac{1}{2} \right] \times \frac{1}{2} \right] \times \frac{1}{2} \right] \times \frac{1}{2} = 1$$

$\log_2 n + 1$ $4^{\log_2 n}$

$$\frac{N}{2^n} = 1$$

8
↓
4
↓
2
↓
1

$$N = 2^n \Rightarrow \log N = \log 2^n$$

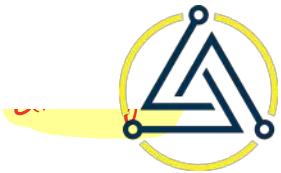
$$\log_2 N = \log_2 2^n$$

↳ $\log_a a^x = x$

$\log_2 n =$ How many times you are dividing

$$\log_2 N = n$$

$\log_2 n + 1 =$ How many nos. will be there while dividing



A.P.: Arithmetic Progression

Series:

$$4 \quad 7 \quad 10 \quad 13 \quad 16 \quad 19$$

$\underbrace{\qquad\qquad}_{+3} \quad \underbrace{\qquad\qquad}_{+3} \quad \underbrace{\qquad\qquad}_{+3} \quad \underbrace{\qquad\qquad}_{+3} \quad \underbrace{\qquad\qquad}_{+3}$

$$\begin{array}{ccccccc} 1^{\text{st}} & 2^{\text{nd}} & 3^{\text{rd}} & 4^{\text{th}} & & & n^{\text{th term}} \\ a & (a+d) & (a+2d) & (a+3d) & \dots & & a+(n-1)d \end{array}$$

first term = a

Common difference = d

// Sum of N terms in A.P. = $\frac{n}{2} [2a + (n-1)d]$

Geometric Progression:

$$3 \quad 6 \quad 12 \quad 24 \quad 48 \quad \dots$$

$$\begin{array}{ccccccc} 1^{\text{st}} & 2^{\text{nd}} & 3^{\text{rd}} & 4^{\text{th}} & & & n^{\text{th term}} \\ a\sigma^0 & a\sigma^1 & a\sigma^2 & a\sigma^3 & \dots & & a\sigma^{n-1} \end{array}$$

a : first term

σ : Common ratio

// Sum of N terms in G.P. =

$$a + \left(\frac{\sigma^n - 1}{\sigma - 1} \right)$$



Quiz 2)

void func (int n) {

s=0

[1, 2, 3, ..., N]

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

$N \times i$ = N iterations

s=s+i;

↑

}

O(N)

return s;

}



Q)

```
void func (int n, int m) {
```

```
    for (int i=1; i<=n; i++) { [1, ..., n]  
        if (i%2==0) {  
            Point(i);  
        }  
    }  
}
```

```
    for (int i=1; i<=m; i++) { [1, ..., m]  
        if (i%2==0) {  
            Point(i);  
        }  
    }  
}
```

Total = $(N+m)$ iterations

$O(N+m)$ $\xrightarrow{N>m} O(N)$
 $\xrightarrow{m>n} O(m)$

Quiz 4:

```
int func (int n) {
```

S=0;

```
    for (int i=0; i<=100; i++) { [0, ..., 100]  
        S = S + i + i2;
```

$$100 - 0 + 1 = 101$$

}

↓

return S;

101 iterations

}



Quiz 5)

void fun (N){
 $i^2 \leq N$
 $i \leq \sqrt{N} [1, \dots, \sqrt{N}]$

for (int i=1; $i * i \leq N$; i++) {

$$S = S + i^2;$$

}

return S;

}

$$\sqrt{N} - 1 + 1 \Rightarrow \sqrt{N}$$

iterations

$\frac{1}{2} O(\sqrt{N})$

Quiz 6)

void fun (n){

$$i = N$$



$$\text{int } i = n;$$

$$i = \frac{n}{2}$$



}

$$i = i/2;$$

$\Rightarrow \log_2 n$
iterations

$$i = \frac{n}{4}$$



}

$$i = \frac{n}{8}$$



$O(\log_2 n)$

$$n \rightarrow \frac{n}{2} \rightarrow \frac{n}{4} \rightarrow \dots \rightarrow 1$$

$$\frac{1}{2} \log_2 n$$

$$i = 1$$



Quiz 7)

```
void fun (int n){  
    s=0;  
    for(int i=0; i<=n; i=i+2){  
        s = s + i;  
    }  
}
```

$[0, 0, 0, \dots, 0]$
 \downarrow
infinite times

3

Quiz 8)

```
void fun (int n){  
    int s=0;  
    for(int i=1; i<n; i=i*2){  
        s = s + i;  
    }  
}
```

1
↓
2
↓
4
↓
8
↓
16
↓
32
↓
 $\Rightarrow \log_2 n$

}

$$n \rightarrow 1 \Rightarrow \log_2 n$$

$$1 \rightarrow n \Rightarrow \log_2 n$$

\downarrow
 n



Nested loops

Qn129) void fun1 (int n){

Num of iteration = How many

times you are running
the statement

for (int i=1; i<=10; i++) {

 for (int j=1; j<=N; j++) {

 Point (i,j);

}

10+N iterations

i	j	
1	[1, N]	N
2	[1, N]	N
3	[1, N]	N
:	:	+
10	[1, N]	N

Qn1210) void fun2 (int n) {

 for (int i=1; i<=N; i++) {

 for (int j=1; j<=N; j++) {

 Point (i,j);

}

i	j	
1	[1, N]	N
2	[1, N]	N
:	:	+
N	[1, N]	N

N+N iteration



Quiz 11:

void fun3 (int n) {

```
for (int i=0; i<N; i++) {
    for (int j=0; j<=i; j++) {
        Point(i*j);
    }
}
```

Point($i*j$);

i	j	
0	[0, 0]	1
1	[0, 1]	2
2	[0, 2]	3
.	.	.
.	.	.
N-1	[0, N-1]	N

$$\frac{N + (N+1)}{2} = \frac{N^2 + N}{2} \text{ iterations}$$

$$\frac{N^2}{2} + \frac{N}{2} = O(N^2)$$

break: 10:30 PM

Quiz 12:

void fun4 (int n) {

```
for (int i=1; i<=n; i++) {
    for (int j=1; j<=n; j=j+2) {
        Point(i*j);
    }
}
```

Point($i*j$);

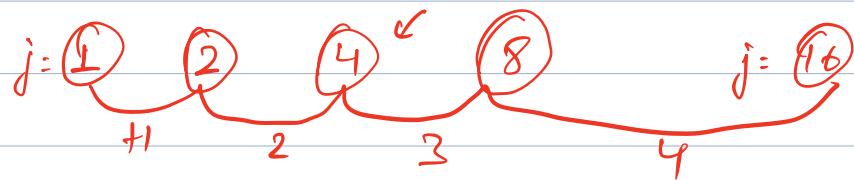
i	j	
1	[1, $n+2$]	$\log_2 n + 1$
2	[1, n]	$\log_2 n + 1$
.	.	.
N	[1, n]	$\log_2 n + 1$

$N * (\log_2 n + 1)$ iteration

$$N \log_2 n + N = O(N \log_2 n)$$



$$N = 16$$



$$\log_2 16 = 4$$

4+1 = 5 times



Quiz 13)

void fun5(int n){

 for (int i=1; i<=2ⁿ; i++) {

 Point(i);

[1, 2, 3 ... 2ⁿ]

}

}

$2^n - 1 + 1 = 2^n$ iterations

Quiz 14:

void fun6(int n){

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

 for (int j=1; j<=2ⁱ; j++) {

 Point(i*j);

}

}

$$g \& sum = \alpha + \left(\frac{2^n - 1}{2 - 1} \right)$$

i	j	
1	[1, 2 ¹]	2^1
2	[1, 2 ²]	2^2
3	[1, 2 ³]	2^3
⋮	⋮	⋮
n	[1, 2 ⁿ]	2^n

$$2^1 + 2^2 + 2^3 + \dots + 2^n = 2 \alpha \left(\frac{2^n - 1}{2 - 1} \right)$$

gp

$$= 2 \alpha (2^n - 1)$$

Ques 15)

$$6 \cdot 2 + 2^n - 2 \Rightarrow O(2^n)$$

```
for(int i=N; i>0; i=i/2) {
    for(int j=1; j<=i; j++) {
        cout << i+j;
    }
}
```

i	j	
N	[1, N]	N
N/2	[1, N/2]	N/2
N/4	[1, N/4]	N/4
...
1	[1, 1]	1

$$N + \frac{N}{2} + \frac{N}{4} + \dots + 1$$

$$\text{no. of terms} = \log_2 N + 1$$

$$\text{(I) } \text{if sum} = a \times \left(\frac{s^{\text{No. of terms}} - 1}{s - 1} \right) = N + \left(\left(\frac{1}{2} \right)^{\log_2 N + 1} - 1 \right)$$

$$\text{(II)} \quad = \frac{N}{-\frac{1}{2}} \times \left[\left(\frac{1}{2} \right)^{\log_2 N + 1} - 1 \right]$$

$$2^{\log_2 N} = N \quad \text{(III)} \quad = -2N \times \left[\frac{1}{\cancel{2^{\log_2 N + 2}}} = 1 \right]$$

$$2^{\log_2 N} = N \quad \text{(IV)} \quad = -2N \left[\frac{1}{2N} - 1 \right]$$

$$\log_2 N = \log_2 N$$

$$N = N \quad \text{(V)} \quad = -\frac{2^{\frac{1}{2N}}}{2^{\frac{1}{2N}}} - (-2N) \Rightarrow 2N - 1$$



II Comparison

↳ for large value of N

$$1 < \log^{\log N} < \sqrt{N} < N < N \log N < \underbrace{N\sqrt{N}}_{N^{\frac{3}{2}}} < \underbrace{N^2}_{N^{\frac{4}{2}}} < 2^N$$
$$1 < \log^{10} < 10 < 10 \log 10 < 10^2 < 10^2 < 2^{10}$$

II How to write Big O

→ what?
Why?

I Calculate iterations based on loop

II Neglect lower order terms.

or

Keep only the highest order term

III Neglect Constant Co-efficient

e.g.: ~~$\alpha N^2 + 100N + 10^4$~~ total no. of iteration
 $O(N^2)$



Quiz 1) $4N^2 + 3N + 10^6$

$\mathcal{O}(N^2)$

Quiz 2) $4N + 3N \log N + 10^6$

$\mathcal{O}(N \log N)$

Quiz 3) $4N \log N + 3N \sqrt{N} + 10^6$

$\mathcal{O}(N \sqrt{N})$



$$\rightarrow 2^n = 1 \Rightarrow n = 0$$

$$n \rightarrow \frac{n}{2} \rightarrow \frac{n}{4} \rightarrow \frac{n}{8} \dots \perp$$

↓
TODO

$n^m + 1^m$
 $a \propto \frac{1}{n-1}$

D

$$\perp = n + \left(\frac{1}{2}\right)^{n-1}$$

$$a^{b+c} = a^b * a^c$$

$$2^{10} = 2^6 * 2^4$$

b) you didn't come this far, only to come this

far



Today's agenda

↳ Binary number system

↳ Operators

↳ Problems

↳ Constraints → Time complexity



AlgoPrep



↳ decimal no. system

↳ {0 - 9} digits

000	010	020	30	91	100
001	011	021	1	1	1
:	012	:	1	1	1
:	:	1	1	1	1
:	1	1	1	1	1
009	019	029	39	99	

↳ binary number system

↳ {0 - 1}

000	010	100	:	:
001	011	101	:	:



11. Conversion

↳ Convert decimal nos to binary:

30 →

2	30	- 0	↑
2	15	- 1	
2	7	- 1	
2	3	- 1	
2	1	- 1	
	0		

→ 11110

Quiz

2	45	- 1	↑
2	22	- 0	
2	11	- 1	
2	5	- 1	
2	2	- 0	
2	1	- 1	
	0		

→ 101101



Us binary no. to decimal no. \rightarrow bit index

$$\begin{array}{r} 0 \cdot 2^3 \quad 0 \cdot 2^2 \\ \uparrow \quad \uparrow \\ 4 \ 3 \ 2 \ 1 \ 0 \\ (1 \ 0 \ 1 \ 0 \ 1)_2 : \\ \downarrow \quad \downarrow \quad \downarrow \\ 1 \cdot 2^4 \quad 1 \cdot 2^3 \quad 0 \cdot 2^2 \\ \rightarrow \text{no.} * 2^{\text{bit index}} \end{array}$$

add all

$$6 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 2^0 + 2^2 + 2^4 = 21$$

0: unset bit / off bit

1: set bit / on bit

quiz:

6 5 4 3 2 1 0
1 0 1 1 0 1 0

$$\hookrightarrow 2^1 + 2^3 + 2^4 + 2^6 = 2 + 8 + 16 + 64 = 90$$

Us $(102010)_2 :$ \rightarrow invalid input



↳ Add binary number

$$\begin{array}{r} 1 \ 3 \ 6 \ 8 \\ + 4 \ 5 \ 4 \\ \hline 8 \ 2 \ 2 \end{array}$$

$$0+0 \rightarrow 0$$

$$1+0 \rightarrow 1$$

$$1+1 \rightarrow 10$$

$$0+1 \rightarrow 1$$

$$1+1+1 \rightarrow 11$$

→ {0, 1}

①

$$\begin{array}{r} 1 & 1 & +1 & +1 & +1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ \hline 0 & 1 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 1 & 0 & 1 \end{array}$$

②

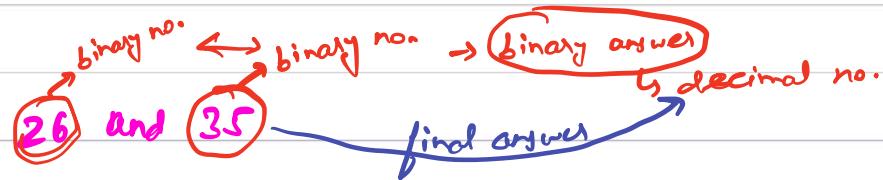
$$\begin{array}{r} 1 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 0 & 1 \end{array}$$

Quiz

$$\begin{array}{r} 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ \hline 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{array}$$



Bitwise operators: → {and, or, xor, leftshift, rightshift?}



A	B	$A \& B$	$A B$	$A ^ B$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Annotations: 0 is dominating, 1 is dominating, Same Same Purify Shame

Symbol → $\&\&$
↓ relational operator
(To combine multiple conditions)

$\&$
↓ Bitwise operator

→ $||$
↓ relational operator
true dominating

↓ Bitwise operator



① $23 \& 10$

$$\begin{array}{l} 23 : 10111 \\ 10 : \frac{01010}{00010} \rightarrow 2^1 = 2 \end{array}$$

Quiz:

$$\begin{array}{l} 20 : 10100 \\ 10 : \frac{01010}{00000} \rightarrow 0 \end{array}$$

② $23 | 10$

$$\begin{array}{l} 23 : 10111 \\ 10 : \frac{01010}{11111} \Rightarrow 2^0 + 2^1 + 2^2 + 2^3 + 2^4 = 31 \end{array}$$

Quiz:

$$\begin{array}{l} 20 : 10100 \\ 10 : \frac{01010}{11110} \Rightarrow 2^1 + 2^2 + 2^3 + 2^4 = 30 \end{array}$$

③ $23^\wedge 10$

$$\begin{array}{l} 23 : 10111 \\ 10 : \frac{01010}{11101} \Rightarrow 2^0 + 2^1 + 2^2 + 2^3 + 2^4 = 29 \end{array}$$

Quiz:

$20^\wedge 15$

$$\begin{array}{l} 20 : 10100 \\ 15 : \frac{01111}{11011} \Rightarrow 2^0 + 2^1 + 2^2 + 2^3 + 2^4 = 27 \end{array}$$



Q) you have been given a Positive no; identify whether the number is even or odd.

Ex: $n = 8 \rightarrow$ even

$n = 7 \rightarrow$ odd

Note: Use of $+$, $-$, $*$, $/$ or $\%$ is not allowed.

$n = 50:$ $\begin{array}{r} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ \hline + 1 \\ \hline 0 \end{array}$

$n = 51:$ $\begin{array}{r} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ \hline + 1 \\ \hline 0 \end{array}$

$n = 12:$ $\begin{array}{r} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ \hline + 1 \\ \hline 0 \end{array}$

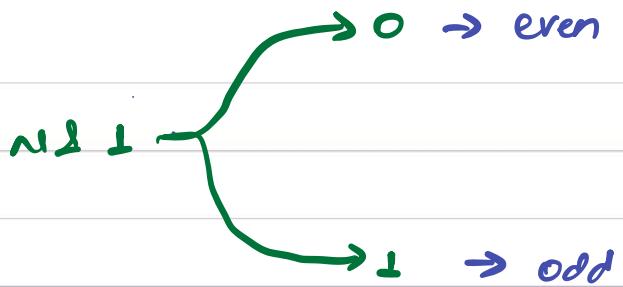
$n = 13:$ $\begin{array}{r} 1 \\ 1 \\ 0 \\ 1 \\ \hline + 1 \\ \hline \end{array}$

- Observation:
- ① rightmost bit for even no. is 0
 - ② rightmost bit for odd no. is 1

To check rightmost bit is 0 or 1, use bitwise AND operator.

$n = 50:$ $\begin{array}{r} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ \hline + 1 \\ \hline 0 \end{array}$

$n = 51:$ $\begin{array}{r} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ \hline + 1 \\ \hline 0 \end{array}$



// Pseudo code

```
if ( $\lfloor \frac{n}{2} \rfloor = 0$ ) {
    S.O.P ("even");
}
else {
    S.O.P ("odd");
}
```



Break till 9:33pm



11 Properties

6 ① Commutative Property

$$a \& b = b \& a$$

$$a \mid b = b \mid a$$

$$a^n b = b^n a$$

7 ② Associative Property

$$a \& b \& c \rightarrow (a \& b) \& c = a \& (b \& c)$$

$$(a \mid b) \mid c = a \mid (b \mid c)$$

$$(a^n b)^n c = a^n (b^n c)$$



7.1

$$\textcircled{a} \quad N \& N = N$$

$$N \rightarrow 110$$

$$\overline{\Delta} \quad \overline{N \rightarrow 110}$$

$$N \rightarrow 110$$

$$\overline{\Delta} \quad \overline{0 \rightarrow 000}$$

$$\textcircled{b} \quad N \& 0 = 0$$

$$N \& N \rightarrow 110$$

$$\overline{0 \rightarrow 000}$$

$$N \& 0 \rightarrow 000$$

$$\textcircled{c} \quad N \mid 0 = N$$

$$N \rightarrow 110$$

$$\overline{0 \rightarrow 000}$$

$$N \rightarrow 110$$

$$\overline{N \rightarrow 110}$$

$$\textcircled{d} \quad N \mid N = N$$

$$N \mid 0 \rightarrow 110$$

$$N \mid N \rightarrow 110$$

$$\textcircled{e} \quad N^n = N$$

$$N \rightarrow 110$$

$$N \rightarrow 110$$

$$\overline{0 \rightarrow 000}$$

$$\overline{N \rightarrow 110}$$

$$\textcircled{f} \quad N^n N = 0$$

$$N^n 0 \rightarrow 110$$

$$N^n N \rightarrow 000$$

$$\overline{N^n 0 \rightarrow 110}$$

$$\overline{N^n N \rightarrow 000}$$



Q) Given $\text{arr}[n]$, every element appears twice except for one element which appears once, find that unique element.

$$\text{Ex: } \text{arr}[7]: \{ 6 \ 8 \ 8 \ 7 \ 7 \ 10 \ 6 \} \rightarrow 10$$

$$\text{arr}[5]: \{ 2 \ 2 \ 1 \ 9 \ 9 \} \rightarrow 1$$

1/Idea

b) Take xor of all array elements.

$$\text{arr}[7]: \{ 6 \ 8 \ 8 \ 7 \ 7 \ 10 \ 6 \}$$

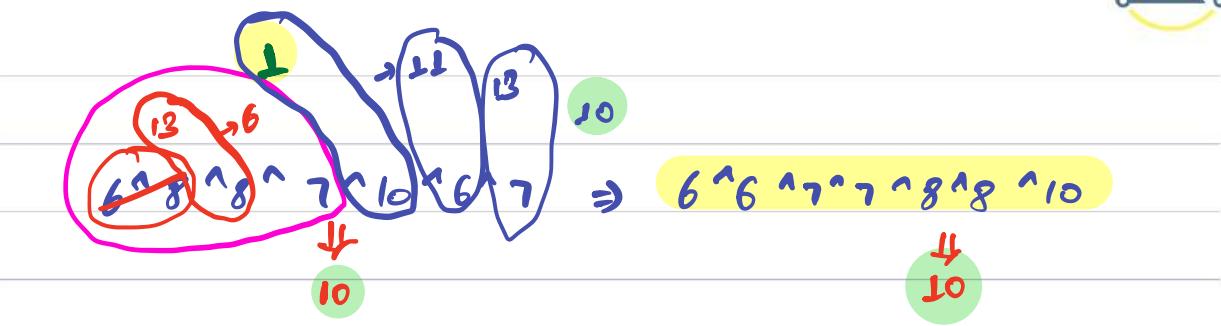
↓

$$6^{\circ} 8^{\circ} 8^{\circ} 7^{\circ} 7^{\circ} 10^{\circ} 6 \Rightarrow 6^{\circ} 6^{\circ} 1^{\circ} 7^{\circ} 7^{\circ} 8^{\circ} 8^{\circ} 10^{\circ} = 10$$

1/ Pseudo code

int ans = 0;

```
| for (int i=0; i<n; i++) {  
|     ans = ans ^ arr[i];  
| }  
| s.o.p(ans);
```



6: 0110

7: 1110

6: 0110

$$8: \frac{1000}{1110 = 14}$$

$$8: \frac{1000}{0110 = 6}$$

$$7: \frac{0111}{0001 = 1}$$

1: 0001

11: 1011

13: 1101

$$10: \frac{1010}{1011 = 11}$$

$$6: \frac{0110}{1101 = 13}$$

$$7: \frac{0111}{1010 = 15}$$

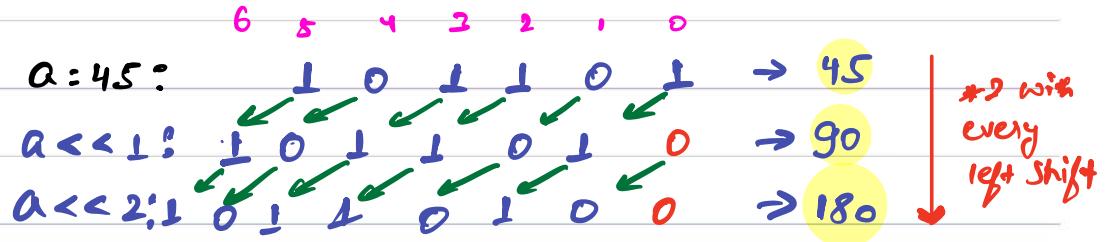
$$10 + 12 + 13 + 15 = 12 + 15 + 10 + 13$$

$$\begin{matrix} 14 \\ \text{Ans1} \end{matrix}$$

$$= \begin{matrix} 14 \\ \text{Ans2} \end{matrix}$$



↳ left shift ($<<$)



$$a << n = a * 2^n$$

Quiz:

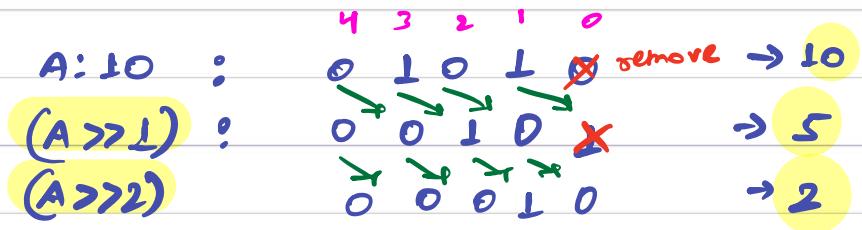
$$a = -1$$

$$-1 << n = -1 * 2^n = 2^n$$

↳ for -2^n , you need
to use left shift
operator.



//right shift ($>>$)



$$A >> n : \frac{A}{2^n}$$



AlgoPrep



* Constraints

1 sec = 10^8 iterations

en: $\hat{N}^{\text{proxy length}} = 10^5$

↳ max allowed

↳ $O(n^2) \rightarrow$ Solution $\rightarrow \times \alpha$

$$(10^5)^2 = 10^{10}$$

↳ $O(n\sqrt{n}) \rightarrow$

$$10^5 * \sqrt{10^5} \Rightarrow 10^5 * 10^{2.5} = 10^{7.5} \quad \checkmark$$



AlgoPrep

b

vmc → Satyam → iit delhi

dtu



↓
Maths → Problems memorize

understand



understand

+
memorize

45 min

+
3 questions



AlgoPrep

IsEven

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        if((n & 1) == 0){
            System.out.println("even");
        }else{
            System.out.println("odd");
        }
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;

    if ((n & 1) == 0) {
        cout << "even" << endl;
    } else {
        cout << "odd" << endl;
    }

    return 0;
}
```

Python Code:

```
n = int(input())
```

```
if n % 2 == 0:  
    print("even")  
else:  
    print("odd")
```

Single Number

Java Code:

```
import java.io.*;  
import java.util.*;  
import java.text.*;  
import java.math.*;  
import java.util.regex.*;  
  
public class Solution {  
  
    public static void main(String[] args) {  
        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 xor = 0;  
        for(int i=0;i<n;i++){  
            xor = xor^arr[i];  
        }  
  
        System.out.println(xor);  
    }  
}
```

C++ Code:

```
#include <iostream>  
using namespace std;
```

```

int main() {
    int n;
    cin >> n;

    int* arr = new int[n];
    for(int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int xorValue = 0;
    for(int i = 0; i < n; i++) {
        xorValue = xorValue ^ arr[i];
    }

    cout << xorValue << endl;

    delete[] arr;
    return 0;
}

```

Python Code:

```

n = int(input())
arr = list(map(int, input().split()))

xor_value = 0
for num in arr:
    xor_value ^= num

print(xor_value)

```

Bit Set ON_HW

Solution Vid:

<https://youtu.be/KbBxyfAfWLA>

Java Code:

```

import java.io.*;
import java.util.*;

```

```
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        int ans = n | (1<<k);

        System.out.println(ans);
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

int main() {
    int n, k;
    cin >> n >> k;

    int ans = n | (1 << k);

    cout << ans << endl;

    return 0;
}
```

Python Code:

```
n, k = map(int, input().split())
ans = n | (1 << k)
print(ans)
```


Today's agenda

↳ No. of iteration

↳ Big O Notations

Quiz

↳ How many numbers are in range $[3, 10]$ {corners included}
 $\{3, 4, 5, 6, 7, 8, 9, 10\} \rightarrow 8$

$$\left[\begin{matrix} a, b \\ \text{inc} & \text{inc} \end{matrix} \right] \rightarrow b - a + 1 \quad \left| \begin{matrix} a, b \\ \text{inc} & \text{enc.} \end{matrix} \right. = b - a \quad \left| \begin{matrix} a, b \\ \text{enc.} & \text{enc.} \end{matrix} \right. = b - a - 1$$

Quiz

↳ No. of steps for $N \rightarrow \frac{N}{2} \rightarrow \frac{N}{4} \rightarrow \dots \perp$

$$n/2 = 10 \Rightarrow n = 10/2$$

$$n/2 = 10 \Rightarrow n = 10 \cdot 2$$

$$n^2 = 10 \Rightarrow n = \sqrt{10}$$

$$n^2 = 10 \Rightarrow 2 = \log_n(10)$$

$\log_b a$
 referent
 base
 number

$$\Rightarrow \log_2 10 = \text{ans} \rightarrow 3.33 \dots$$

$$10 = 2^{\text{ans}}$$

$$\text{ans} = 2^{3.33 \dots}$$

$$\log_b a = \text{ans} \Rightarrow a = b^{\text{ans}}$$

$$\rightarrow \log_2 64 = \text{ans} \rightarrow 64 = 2^{\text{ans}} \rightarrow 6$$

$$\rightarrow \log_2 32 = \text{ans} \rightarrow 2^{\text{ans}} = 32 \rightarrow 5$$

$$\rightarrow \log_2 33 = \text{ans} \rightarrow 2^{\text{ans}} = 33 \rightarrow 5. \dots$$

Properties

$$\text{i) } \log_b a^n = n$$

$$\text{ex: } \log_5 5^{10} = 10$$

$$\text{ii) } \log_c (a \times b) = \log_c a + \log_c b$$

$$\text{ex: } \log_2 10 = \log_2 2 + \log_2 5$$

Quiz

↳ No. of steps for $N \rightarrow \frac{N}{2} \rightarrow \frac{N}{4} \rightarrow \dots \rightarrow 1$

$$\left(\left(N * \frac{1}{2} \right) * \frac{1}{2} \right) * \frac{1}{2} \dots = 1$$

$$\frac{N}{2^{\text{Count}}} = 1 \Rightarrow N = 2^{\text{Count}}$$

$$N = \underbrace{2}_{2^{\text{Count}}} \underbrace{\downarrow}_{\vdots} \underbrace{N}_{\downarrow}$$

$$\text{Count} = \log_2 N$$

A.P \rightarrow Arithmetic Progression

$$3 \quad 7 \quad 11 \quad 15 \quad 19 \quad 23 \quad 27 \quad \dots$$

+4 +4 +4 +4 +4 +4

first term = a = Starting number

Common diff = d = difference between consecutive nos

$$(a+0+d) \quad (a+d) \quad (a+2+d) \quad (a+3+d) \quad \dots \quad a+(n-1)*d$$

\hookrightarrow Sum of first n terms : $\frac{n}{2} [2*a + (n-1)*d]$

G.P \rightarrow geometric Progression

$$3 \quad 6 \quad 12 \quad 24 \quad 48 \quad 96 \quad \dots$$

*2 *2 *2 *2 *2

$$50 \quad 25 \quad 12.5 \quad 6.25$$

*1/2 *1/2 *1/2

first term = a = Starting number.

Common ratio: σ = multiple to get next no.

$$a \quad a*\sigma \quad a*\sigma^2 \quad a*\sigma^3 \quad \dots \quad a*\sigma^{n-1}$$

\hookrightarrow Sum of n terms in G.P = $a * \frac{\sigma^n - 1}{\sigma - 1}$

Quiz

```
int Sum=0;
```

```
| for (int i=1; i<=n; i++) {  
|   Sum = Sum + i;  
| }
```

$[1, n] \rightarrow n - 1 + 1$

$n^{\text{iterations}}$

\downarrow
 $O(n)$

Quiz

```
void func(int n, int m){
```

```
| for (int i=1; i<=n; i++) {  
|   Point(i);  
| }
```

$[1, n] = n - 1 + 1 = n$ iterations

```
| for (int i=1; i<=m; i++) {  
|   Point(i);  
| }
```

Total = $n + m$ iterations



Quiz

```
int fun (int n){  
    int s=0;  
    for (int i=0; i<=100; i++) {  
        s = s + i*i;  
    }  
    return s;  
}
```

$\rightarrow [0, 100] \rightarrow 100 - 0 + 1 = 101$ iterations
 \downarrow
 $O(n)$

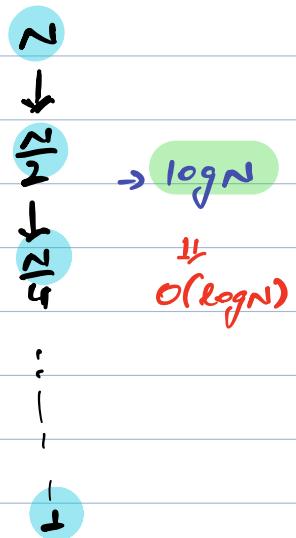
Quiz

```
void fun (int n){  
    int s=0;  
    for (int i=1; i<=n; i++) {  
        s = s + i*i;  
    }  
    return s;  
}
```

$\rightarrow [1, \sqrt{n}] \Rightarrow \sqrt{n} - 1 + 1 = \sqrt{n}$ iterations
 \downarrow
 $O(\sqrt{n})$

Quiz

```
void fun(int n){  
    int i=n;  
    while(i>=1){  
        i = i/2;  
    }  
}
```



Quiz

```
void fun(int n){  
    int s=0;  
    for(int i=0; i<=n; i=i*2){  
        s = s+i;  
    }  
}
```

[0, 0, 0, 0, ...]

infinite

Quiz

```
void fun (int N) {  
    int S=0;  
    for (int i=1; i<=N; i=i+2) {  
        S = S + i;  
    }  
}
```

3
3

1 → 2 → 4 → 8 . . .

... $\frac{N}{4} \rightarrow \frac{N}{2} \rightarrow N$

($\log N$)
 $O(\log N)$

$N \rightarrow \frac{N}{2} \rightarrow \frac{N}{4} \rightarrow \frac{N}{8} \dots \frac{N}{4^k}$

1
+ 2
+ 4
. . .
1
—
 $\frac{N}{4}$
 $\frac{N}{8}$
 $\frac{N}{16}$

$\log N \leftarrow$

1 2 3 4 5 ⇒ 5
5 4 3 2 1 ⇒ 5

Bölekt till 9:30pm

Nested Loops

Quiz

```
void fun (int n) {
    int S=0;
    for (int i=1; i<=10; i++) {
        for (int j=1; j<=n; j++) {
            S = S + 10;
        }
    }
}
```

$$S = S + 10;$$

i	j	Count
1	[1, n]	n ⁺
2	[1, n]	n ⁺
3	[1, n]	n ⁺
.	.	.
.	.	.
.	.	.
10	[1, n]	n ⁺

↓
10*n iterations

Quiz:

```
void fun (int n) {
    int S=0;
    for (int i=1; i<=N; i++) {
        for (int j=1; j<=n; j++) {
            S = S + 10;
        }
    }
}
```

$$S = S + 10;$$

i	j	Count
1	[1, n]	n ⁺
2	[1, n]	n ⁺
3	[1, n]	n ⁺
.	.	.
.	.	.
.	.	.
N	[1, n]	n ⁺

↓
(N*n) iterations

Quiz:

```
void fun (int n) {
    int s=0;
    for (int i=1; i<=n; i++) {
        for (int j=1; j<=i; j++) {
            s = s + j;
        }
    }
}
```

$$S = S + j;$$

3

$$\frac{N(N+1)}{2} = \frac{N^2+N}{2} = \frac{N^2}{2} + \frac{N}{2}$$

i	j	Count
1	[1,1]	1
2	[1,2]	2
3	[1,3]	3
.	.	.
N	[1,N]	N

$\frac{N(N+1)}{2}$ iterations

$O(N^2)$

Quiz

```
void fun (int n) {
```

$[1, 2^N] \rightarrow 2^N$ iterations

```
for (int i=1; i<=2^N; i++) {
```

Point(i);

3

\downarrow
 $O(2^n)$

3

Quiz:

```

void fun (int n) {
    int s=0;
    for (int i=1; i<=n; i++) {
        for (int j=1; j<=2i; j++) {
            s = s + 10;
        }
    }
}

```

i	j	Count
1	[1, 2 ¹]	2^1
2	[1, 2 ²]	2^2
3	[1, 2 ³]	2^3
.	.	.
1	1	
1	1	
1	1	
N	[1, 2 ^N]	2^N

$$2 + \underbrace{2^2}_{*2} + \underbrace{2^3}_{*2} + \underbrace{2^4}_{*2} \dots - - - 2^N = 2 * \frac{2^N - 1}{2 - 1} = 2 * (2^N - 1)$$

iterations

$$2 * \frac{2^N - 1}{2 - 1}$$

$$\cancel{2 * 2^N} - \cancel{2}$$

$$O(2^N)$$

Comparison of iteration

$$1 < \log_2 n < \sqrt{n} < n < n \cdot \log n < n\sqrt{n} < n^2 < 2^n \dots$$

Time Complexity

↳ Big O Notation → Approach iteration

- (I) Calculate iteration
- (II) Neglect lower order terms
- (III) Neglect Constants

e.g.: $(2n^2 + 15n + 10)$ iteration

$$\downarrow \\ O(n^2)$$

e.g.:

$$8n^2 + 15n + 10 \\ (\text{is } O(n^2))$$

e.g. $4n + 8n \log n + 10$
 $(\text{is } O(n \log n))$

en:

$$\cancel{N \log N} + \cancel{15N} + \cancel{100}$$

$\in O(N \log N)$



Today's agenda

↳ HashMap Intro

↳ frequency of each query

↳ first non-repeating elements

↳ HashSet

↳ number of distinct elements

↳ Pair sum == k → O(n)

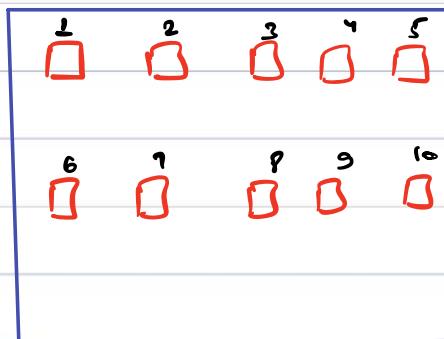


AlgoPrep



II Hashmap Intro

① mohit:



true = occupied
false = empty

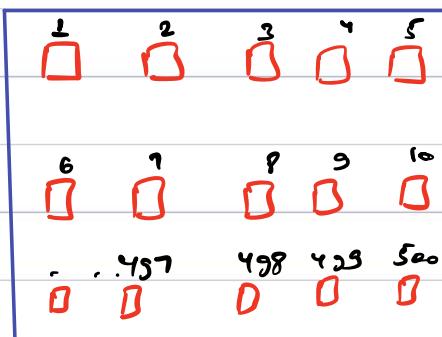
boolean n1 =

n2 =

n3 = *false* *true*

i
n = 10 :

② Adarsh



boolean arr[501]

↓

arr[500] = true

if (arr[350] == false){

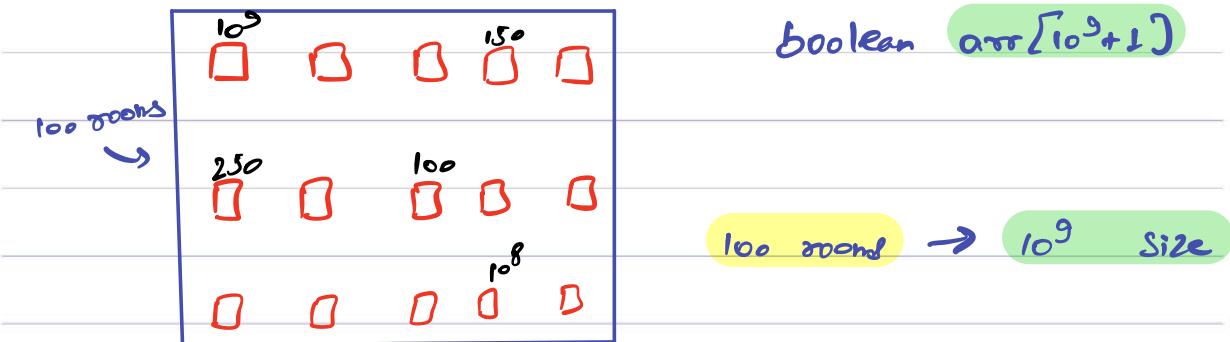
Point ("Room given");
arr[350] = true;

}



III Caneham

$\rightarrow \{1, 10^9\}$



6 Mathmal solves the issue

100 size

+
0.99

key (int)	value (boolean)
10 ⁹	false
150	false
250	false
100	false
10 ⁸	false

(key, value) Pair



Q) Store Population of every Country:

Key: Country name: String

value: Population : int/long

Key (String) value (int)

Key (String)	value (int)
India	140...

Q) Store All the School with their Principal name.

Key: String

value: String



facts:

① Keys can be only of following datatype:

Wrapper Class

boolean → Boolean

int → Integer

long → Long

char → Character

double → Double

String → String

Not applicable: arrays, Objects, null

②

values can be of any type.



Syntax:

HashMap< Integer, Integer > hm =
new HashMap<>();

key type value type name

//add

hm.put(10, 50);
hm.put(20, 60);
hm.put(30, 60);
hm.put(20, 70); → replace

hm	Integer	Integer
10	50	50
20	60	70
30	60	

T.C: $O(1)$

//get

hm.get(20); → 70
hm.get(40); → null

//size

hm.size()

T.C: $O(1)$

// ContainsKey → check if key exists

hm.containsKey(20); → true
hm.containsKey(40); → false

T.C: $O(1)$



//remove

hm.remove (10); → it will remove (10, 50)

T.C: O(1)

int arr[n];
→ for (int i=0; i<n; i++) {
 System.out.println (arr[i]);

2

int arr[n];
for (int val: arr) {
 System.out.println (val);

3

iterating on keys

↳ for (int key: hm.keySet()) {

3



Q) find frequency

↳ Given N array elements & Q queries. for every query find frequency of element in array.

Ex: $\text{arr}[i] = \{2, 6, 3, 8, 2, 8, 2, 3, 8, 10, 6\}$
 $\text{queries}[i] = \{2, 8, 3, 5\}$

Idea:

↳ iterate and count for every query.

IPSuedo code

void PointFrequency (int arr[], int queries[]){

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

 int val = queries[i];

 int count = 0;

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

 if (arr[j] == val) {

 Count++;

}

 System.out.println(count);

T.C: $O(m \times n)$

3

3



void PointFrequency (int arr[n], int queries[m])

Ex: $\text{arr}[i] = \{2, 6, 3, 8, 2, 8, 2, 3, 8, 10, 6\}$
 $\text{queries}[i] = \{2, 8, 3, 5\}$

```
for (int i=0; i<m; i++) {  
    int val = queries[i];  
    int count = 0;  
    for (int j=0; j<n; j++) {  
        if (arr[j] == val) {  
            count++;  
        }  
    }  
    System.out.println(count);  
}
```

val = 2 count = 0



AlgoPrep

Idea 2

$\text{arr}[i] = \{2, 6, 3, 8, 2, 8, 2, 3, 8, 10, 6\}$

$\text{queries}[i] = \{2, 8, 3, 5\}$

hm

Integer(key) → Integer(val) → frequency

2	XZ3
6	XZ2
3	XZ2
8	XZ3
10	1

3 3 2 0

T.C: $O(N) + O(M)$



IIIPseudo Code

```
void PointFrequency (int arr[n], int queries[m]) {
```

```
    HashMap<Integer, Integer> hm = new HashMap<>();
```

```
    for (int i=0; i<n; i++) {
        if (hm.containsKey (arr[i])) := true) {
            int temp = hm.get (arr[i]);
            hm.put (arr[i], temp + 1);
        } else {
            hm.put (arr[i], 1);
        }
    }
```

T.C = O(N+m)

```
    for (int i=0; i<m; i++) {
        int val = queries[i];
        if (hm.containsKey (val)) := true) {
            Point (hm.get (val));
        } else {
            Point (0);
        }
    }
```



$\text{arr}[i] = \{ 2 \ 6 \ 3 \ 8 \ 2 \ 8 \ 2 \ 3 \ 8 \ 10 \ 6 \}$

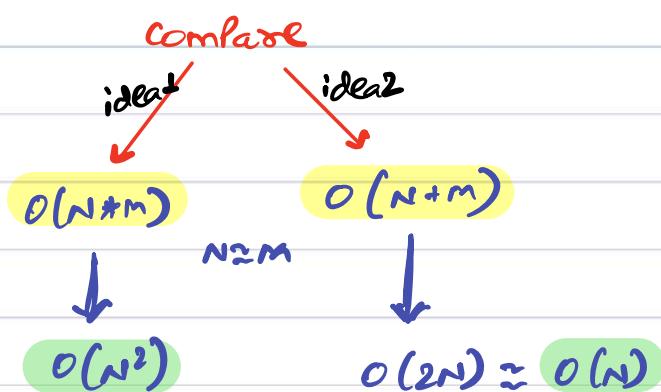
```

for (int i=0; i<n; i++) {
    if (hm.containsKey(arr[i])) := true) {
        int temp = hm.get(arr[i]);
        hm.put(arr[i], temp + 1);
    } else {
        hm.put(arr[i], 1);
    }
}

```

hm	integer(key) → integer(val)
2	- ↗ 23
6	- ↗ 22
3	- ↗ 22
8	- ↗ 23
10	- ↗ 1

i	arr[i]	hm.containsKey(arr[i])	temp
0	2	b	
1	6	b	
2	3	b	
3	8	b	
4	2	T	1
5	8	T	1
6	2	T	2



Break till 9:50 PM



Q) Find the first non-repeating elements

(return -1 if all the elements are repeating.)

Ex1: arr[7]: {1 2 3 1 2 5} → 3

arr[8]: {5 4 4 3 6 7 5 6} → 3

//idea

arr[8]: {5 4 4 3 6 7 5 6}

hm

5	-	x 2
4	-	x 2
3	-	1
6	-	x 2
7	-	1

ll iterate on array and get the first element with frequency 1.



11 Psuedo Code

P S int firstNonRepeating (int arr[N]) {

 HashMap<Integer, Integer> hm = new HashMap<>();

```
    for (int i=0; i<N; i++) {  
        if (hm.containsKey(arr[i])) := true) {  
            int temp = hm.get(arr[i]);  
            hm.put (arr[i], temp + 1);  
        }  
        else {  
            hm.put (arr[i], 1);  
        }  
    }
```

T.C: $O(2N)$
 $\approx O(N)$

```
    for (int i=0; i<N; i++) {  
        if (hm.get(arr[i]) == 1) {  
            return arr[i];  
        }  
    }
```

return -1;

}

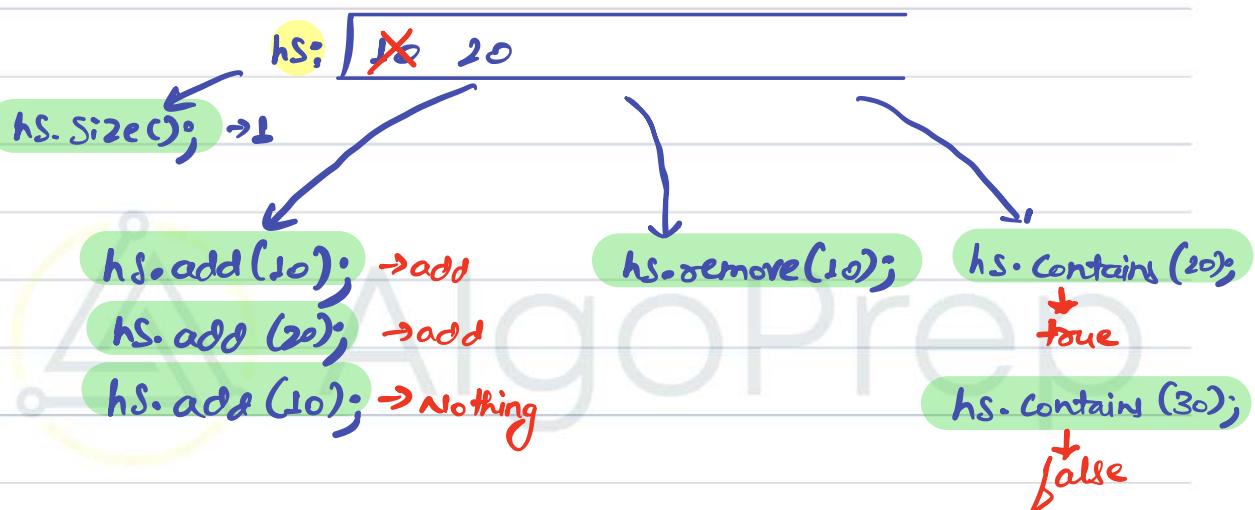


// HashSet

↳ only the key part of hashmap

HashSet < Integer > hs = new HashSet<>();

↳ random order





Q) Given $\text{arr}[n]$, find no. of distinct elements.

Ex: $\text{arr}[5] = \{4, 6, 7, 6, 5\} \rightarrow \text{ans} = 4$

$\text{arr}[5] = \{10, 10, 10, 20, 20, 3\} \rightarrow \text{ans} = 2$

Idea

$\text{arr}[5] = \{10, 10, 10, 20, 20, 3\}$

hs: 10 20

($\text{ans} = \text{hs.size()}$)

IPsuedo code

P S int distinctelements (int arr[n]) {

 HashSet<Integer> hs = new HashSet<>();

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

 return hs.size();

}



Q) Pair Sum = k

↳ Given arr[n], check if there exists a pair (i, j) such that $arr[i] + arr[j] = k$ and $(i \neq j)$.

$arr[10]:$ 0 1 2 3 4 5 6 7 8 9
 8 9 1 -2 4 5 11 -6 7 5

$$K = 11 \rightarrow arr[4] + arr[8] \rightarrow 4 + 7 = 11 \rightarrow \text{true}$$

$$K = 6 \rightarrow arr[0] + arr[3] \rightarrow 8 - 2 = 6 \rightarrow \text{true}$$

$$K = 22 \rightarrow \cancel{arr[6]} + \cancel{arr[6]} \rightarrow \text{false}$$

1/ideal

↳ Nested loop, Check all pairs for $\text{Sum} = k$.

T.C: $O(n^2)$



Idea 2

$\text{arr}[10]: 8 \ 9 \ 1 \ -2 \ 4 \ 5 \ 11 \ -6 \ 7 \ 5$

Step 1: Insert all the elements in HashSet.

hs: { 8 9 1 -2 4 5 11 -6 7 }

$$\textcircled{1} \quad a + b = 11$$

a	b	b is Present?
8	3	NO
9	2	NO
1	10	NO
-2	13	NO
4	7	YES \rightarrow true

\textcircled{2} $\text{arr}[10]: 8 \ 9 \ 1 \ -2 \ 4 \ 5 \ 11 \ -6 \ 7 \ 5$

hs: { 8 9 1 -2 4 5 11 -6 7 }

$$a + b = -4$$

a	b	is b Present
8	-12	NO
9	-13	NO
1	-5	NO
-2	-2	YES \rightarrow true



Ideas

↳ insert all elements of array in your hashmap with frequency.

arr[10]: 8 9 1 -2 4 5 11 -6 7 5

hm	0	1	2	3	4	5	6	7	8	9
	8	-1	11-1							
	9	-1		-6-1						
	1	-1			7-1					
	-2	-1								
	4	-1								
	5	-12								

$$a+b = -4$$

a

b

is b Present

8

-12

NO

9

-13

NO

1

-5

NO

-2

-2

yes, but at same index



//Pseudo Code

boolean PairSum (int arr[], int k){

 HashMap<Integer, Integer> hm = new HashMap<>();

```
    for (int i=0; i<n; i++) {  
        if (hm.containsKey(arr[i])) := true) {  
            int temp = hm.get(arr[i]);  
            hm.put(arr[i], temp + 1);  
        }  
        else {  
            hm.put(arr[i], 1);  
        }  
    }
```

T.C: $\Theta(n)$

```
    for (int i=0; i<n; i++) {  
        int a = arr[i];  
        int b = k - a;
```

```
        if (a != b && hm.containsKey(b) := true) {  
            return true;  
        }  
        else if (a == b && hm.get(b) > 1) {  
            return true;  
        }  
    }  
    return false;
```

3

D -



`arr[10]: 8 9 1 -2 4 5 11 7 8 9`

hm

8 - 1	11 - 1
9 - 1	-8 - 1
1 - 1	7 - 1
-2 - 1	
4 - 1	
5 - 12	

$$a+b = 4$$

a b is b Present

8 -12 NO

9 -13 NO

1 -5 NO

-2 -2 NO, Same index

4 -8 YES → true

```
for (int i=0; i<n; i++) {
    int a = arr[i];
    int b = k-a;
```

```
if (a != b && hm.containsKey(b) == true) {
    return true;
}
else if (a == b && hm.get(b) > 1) {
    return true;
}
```



AlgoPrep

Frequency queries

Java Code:

```
import java.util.*;
import java.util.Map;

class Solution {
    public static void FrequencyQueries(int[] arr,int[] queries) {
        HashMap<Integer, Integer> hm = new HashMap<>();
        for (int num : arr) {
            if(hm.containsKey(num)== true){
                int temp = hm.get(num);
                hm.put(num,temp+1);
            }else{
                hm.put(num,1);
            }
        }

        for (int num : queries) {
            if(hm.containsKey(num)== true){
                System.out.println(hm.get(num));
            }else{
                System.out.println(0);
            }
        }
    }

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

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

        int[] queries = new int[m];
        for(int i=0;i<m;i++){
            queries[i] = scn.nextInt();
        }

        FrequencyQueries(arr, queries); // Output: 3
    }
}
```

```
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <unordered_map>  
#include <vector>  
  
using namespace std;  
  
void FrequencyQueries(vector<int>& arr, vector<int>& queries) {  
    unordered_map<int, int> hm;  
  
    for (int num : arr) {  
        if (hm.find(num) != hm.end()) {  
            int temp = hm[num];  
            hm[num] = temp + 1;  
        } else {  
            hm[num] = 1;  
        }  
    }  
  
    for (int num : queries) {  
        if (hm.find(num) != hm.end()) {  
            cout << hm[num] << "\n";  
        } else {  
            cout << "0\n";  
        }  
    }  
}  
  
int main() {  
    int n, m;  
    cin >> n >> m;  
  
    vector<int> arr(n);  
    for (int i = 0; i < n; i++) {  
        cin >> arr[i];  
    }  
  
    vector<int> queries(m);  
    for (int i = 0; i < m; i++) {  
        cin >> queries[i];  
    }
```

```
FrequencyQueries(arr, queries);

return 0;
}
```

Python Code:

```
def FrequencyQueries(arr, queries):
    hm = {}

    for num in arr:
        if num in hm:
            hm[num] += 1
        else:
            hm[num] = 1

    result = []
    for num in queries:
        if num in hm:
            result.append(hm[num])
        else:
            result.append(0)

    return result

if __name__ == "__main__":
    n, m = map(int, input().split())
    arr = list(map(int, input().split()))
    queries = []

    for _ in range(m):
        queries.append(int(input()))

    results = FrequencyQueries(arr, queries)
    for res in results:
        print(res)
```

Non repeating elements

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

class Solution {
    public static int firstNonRepeating(int[] arr) {
        HashMap<Integer, Integer> hm = new HashMap<>();
        for (int num : arr) {
            if(hm.containsKey(num)== true){
                int temp = hm.get(num);
                hm.put(num,temp+1);
            }else{
                hm.put(num,1);
            }
        }
        for (int num : arr) {
            if (hm.get(num) == 1) {
                return num;
            }
        }
        return -1; // If no non-repeating element is found
    }

    public static void main(String[] args) {
        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();
        }
        System.out.println(firstNonRepeating(arr)); // Output: 3
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
```

```

#include <unordered_map>
using namespace std;

int firstNonRepeating(const vector<int>& arr) {
    unordered_map<int, int> hm;
    for (int num : arr) {
        if (hm.count(num) == 1) {
            int temp = hm[num];
            hm[num] = temp + 1;
        } else {
            hm[num] = 1;
        }
    }
    for (int num : arr) {
        if (hm[num] == 1) {
            return num;
        }
    }
    return -1; // If no non-repeating element is found
}

int main() {
    int n;
    cin >> n;
    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }
    cout << firstNonRepeating(arr) << endl; // Output: 3
    return 0;
}

```

Python Code:

```

def first_non_repeating(arr):
    hm = {}
    for num in arr:
        if num in hm:
            hm[num] += 1
        else:
            hm[num] = 1
    for num in arr:
        if hm[num] == 1:
            return num
    return -1 # If no non-repeating element is found

```

```
def main():
    n = int(input())
    arr = list(map(int, input().split()))
    print(first_non_repeating(arr)) # Output: 3

if __name__ == "__main__":
    main()
```

Distinct element in an array

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
        should be named Solution. */
        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();
        }

        System.out.println(countDistinct(arr));
    }

    public static int countDistinct(int[] arr){
        int n = arr.length;
        HashSet<Integer> hs = new HashSet<Integer>();

        for (int i = 0; i < n; i++) {
            // add all the elements to the HashSet
            hs.add(arr[i]);
        }

        // return the size of hashset as
        // it consists of all Unique elements
        return hs.size();
    }
}
```

C++ Code:

```
#include <iostream>
#include <unordered_set>
using namespace std;
```

```

int countDistinct(int arr[], int n) {
    unordered_set<int> us;

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

    return us.size();
}

int main() {
    int n;
    cin >> n;

    int arr[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    cout << countDistinct(arr, n) << endl;

    return 0;
}

```

Python Code:

```

def countDistinct(arr):
    return len(set(arr))

n = int(input())
arr = list(map(int, input().split()))

print(countDistinct(arr))

```

Check Pair Sum

Java Code:

```

import java.io.*;
import java.util.*;
import java.text.*;

```

```

import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int k = scn.nextInt();
        int[]arr = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = scn.nextInt();
        }

        if(PairSum(arr,k) == true){
            System.out.println("Y");
        }else{
            System.out.println("N");
        }
    }

    public static boolean PairSum(int[] arr, int k) {
        int n = arr.length;
        HashMap<Integer,Integer> map = new HashMap<>();
        for (int num : arr) {
            if(map.containsKey(num) == true){
                int temp = map.get(num);
                map.put(num,temp+1);
            }else{
                map.put(num,1);
            }
        }

        for(int i=0;i<n;i++){
            int a = arr[i];
            int b = k - a;

            if((a != b) && (map.containsKey(b)==true)){
                return true;
            }else if((a == b) && (map.get(b) > 1)){
                return true;
            }
        }

        return false;
    }
}

```

```
}
```

C++ Code:

```
#include <iostream>
#include <unordered_map>
#include <vector>
using namespace std;

bool PairSum(vector<int>& arr, int k) {
    unordered_map<int, int> map;
    for (int num : arr) {
        if (map.count(num) == 1) {
            map[num]++;
        } else {
            map[num] = 1;
        }
    }

    int n = arr.size();
    for (int i = 0; i < n; i++) {
        int a = arr[i];
        int b = k - a;

        if ((a != b) && (map.count(b) == 1)) {
            return true;
        } else if ((a == b) && (map[b] > 1)) {
            return true;
        }
    }

    return false;
}

int main() {
    int n, k;
    cin >> n >> k;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    if (PairSum(arr, k)) {
        cout << "Y" << endl;
    }
}
```

```

} else {
    cout << "N" << endl;
}

return 0;
}

```

Python Code:

```

def pair_sum(arr, k):
    freq_map = {}
    for num in arr:
        if num in freq_map:
            freq_map[num] += 1
        else:
            freq_map[num] = 1

    n = len(arr)
    for i in range(n):
        a = arr[i]
        b = k - a

        if (a != b) and (b in freq_map):
            return True
        elif (a == b) and (freq_map[b] > 1):
            return True

    return False

def main():
    n, k = map(int, input().split())
    arr = list(map(int, input().split()))

    if pair_sum(arr, k):
        print("Y")
    else:
        print("N")

if __name__ == "__main__":
    main()

```

Count of right triangles_HW

Solution Vid:

<https://youtu.be/219ae5GyKJA>

Java Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int[][] a = new int[n][2];

        for(int i=0;i<n;i++){
            for(int j=0;j<2;j++){
                a[i][j] = scn.nextInt();
            }
        }

        System.out.println(RightAngled(a,n));
    }

    static int RightAngled(int a[][], int n){

        HashMap<Integer, Integer> xpoints = new HashMap<>();
        HashMap<Integer, Integer> ypoints = new HashMap<>();

        for (int i = 0; i < n; i++) {
            if(xpoints.containsKey(a[i][0])){
                xpoints.put(a[i][0], xpoints.get(a[i][0]) + 1);
            }else{
                xpoints.put(a[i][0], 1);
            }
            if(ypoints.containsKey(a[i][1])){
                ypoints.put(a[i][1], ypoints.get(a[i][1]) + 1);
            }else{
                ypoints.put(a[i][1], 1);
            }
        }

        int count = 0;
        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                if (xpoints.get(a[i][0]) > 0 && xpoints.get(a[j][0]) > 0 && ypoints.get(a[i][1]) > 0 && ypoints.get(a[j][1]) > 0) {
                    count++;
                }
            }
        }

        return count;
    }
}
```

```

        }

    }

    int count = 0;

    for (int i = 0; i < n; i++){
        if (xpoints.get(a[i][0]) >= 1 &&
            ypoints.get(a[i][1]) >= 1){

            count += (xpoints.get(a[i][0]) - 1) *
                (ypoints.get(a[i][1]) - 1);
        }
    }

    return count;
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <map>

using namespace std;

int RightAngled(vector<vector<int>>& a, int n) {
    map<int, int> xpoints;
    map<int, int> ypoints;

    for (int i = 0; i < n; i++) {
        if (xpoints.count(a[i][0])) {
            xpoints[a[i][0]] = xpoints[a[i][0]] + 1;
        } else {
            xpoints[a[i][0]] = 1;
        }
        if (ypoints.count(a[i][1])) {
            ypoints[a[i][1]] = ypoints[a[i][1]] + 1;
        } else {
            ypoints[a[i][1]] = 1;
        }
    }

    int count = 0;

```

```

for (int i = 0; i < n; i++) {
    if (xpoints[a[i][0]] >= 1 && ypoints[a[i][1]] >= 1) {
        count += (xpoints[a[i][0]] - 1) * (ypoints[a[i][1]] - 1);
    }
}

return count;
}

int main() {
    int n;
    cin >> n;
    vector<vector<int>> a(n, vector<int>(2));

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < 2; j++) {
            cin >> a[i][j];
        }
    }

    cout << RightAngled(a, n) << endl;

    return 0;
}

```

Python Code:

```

def RightAngled(a, n):
    xpoints = {}
    ypoints = {}

    for i in range(n):
        if a[i][0] in xpoints:
            xpoints[a[i][0]] += 1
        else:
            xpoints[a[i][0]] = 1

        if a[i][1] in ypoints:
            ypoints[a[i][1]] += 1
        else:
            ypoints[a[i][1]] = 1

    count = 0

```

```

for i in range(n):
    if xpoints[a[i][0]] >= 1 and ypoints[a[i][1]] >= 1:
        count += (xpoints[a[i][0]] - 1) * (ypoints[a[i][1]] - 1)

return count

if __name__ == "__main__":
    n = int(input())
    a = []

    for i in range(n):
        row = list(map(int, input().split()))
        a.append(row)

print(RightAngled(a, n))

```

Distinct Points

Java Code:

```

import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

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

        int[] x = new int[n];
        int[] y = new int[n];

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

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

```

```

        System.out.println(Max_Points(x,y));
    }

static long Max_Points(int[] X, int[] Y){
    HashSet<String> hs = new HashSet<>();

    for(int i=0;i<X.length;i++){
        hs.add(X[i]+" "+Y[i]);
    }

    return hs.size();
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <unordered_set>

using namespace std;

long Max_Points(vector<int>& X, vector<int>& Y) {
    unordered_set<string> hs;

    for (int i = 0; i < X.size(); i++) {
        hs.insert(to_string(X[i]) + " " + to_string(Y[i]));
    }

    return hs.size();
}

int main() {
    int n;
    cin >> n;

    vector<int> x(n);
    vector<int> y(n);

    for (int i = 0; i < n; i++) {
        cin >> x[i];
    }

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

```

```
    cin >> y[i];
}

cout << Max_Points(x, y) << endl;

return 0;
}
```

Python Code:

```
def Max_Points(X, Y):
    hs = set()

    for i in range(len(X)):
        hs.add(str(X[i]) + " " + str(Y[i]))

    return len(hs)

if __name__ == "__main__":
    n = int(input())
    x = list(map(int, input().split()))
    y = list(map(int, input().split()))

    print(Max_Points(x, y))
```



Today's agenda

↳ Recursion

↳ How to write Recursive Code.

Why recursion?

↳ Tree

↳ Backtracking

↳ DP → Amazon

↳ Graph → Google





Recursion:

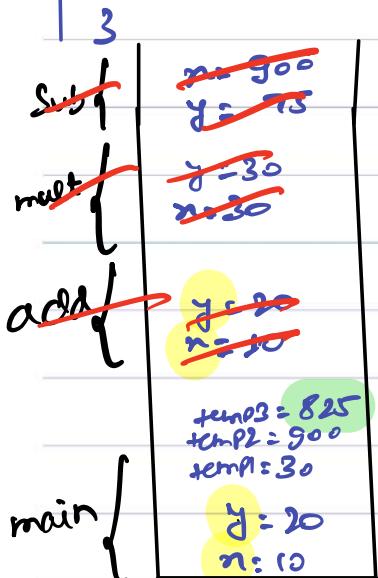
↳ function calling itself.

* function call

```
main () {  
    int x = 10;  
    int y = 20;  
    int temp1 = add(x, y);  
    int temp2 = mult(temp1, 30);  
    → int temp3 = Sub(temp2, 75);
```

s.o.p(temp3);

6,825



```
int add(int x, int y){  
    → return x+y;  
}
```

```
int mult(int x, int y){  
    → return x*y;  
}
```

```
int Sub(int x, int y){  
    → return x-y;  
}
```



// Thought Process

$$\text{Sum}(n) = 1 + 2 + 3 + 4 + \dots + (n-1) + n$$

$$\text{Sum}(n) = \text{Sum}(n-1) + n$$

$$\text{Sum}(n-1) = 1 + 2 + 3 + \dots + (n-2) + (n-1)$$

$$\text{Sum}(n-1) = \text{Sum}(n-2) + (n-1)$$

$$\text{Sum}(n-2) = \text{Sum}(n-3) + (n-2)$$

$$\text{Sum}(1) = \text{Sum}(0) + 1$$

$$\text{Sum}(5) = \text{Sum}(4) + 5$$



Q) Given n , find sum of no.s from $(1 \dots n)$.

Three magical steps of recursion.

Faith: what your function should do and have faith that the function works.

Main logic: Solving your Problem with SubProblem.

Smaller instance of
Same Problem

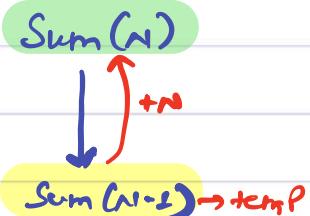
Base Conditions: Solution to Smallest SubProblem.

int sum (int n){
if ($n == 1$) return 1; }

Faith: Given n , calculate & return Sum of n numbers.

int temp = sum ($n - 1$);
return temp + n ;

Main logic:



3

base case:

$$\text{sum}(1) = 1$$



```
int sum (int n) {  
    1 if (n==1) return 1; 3  
  
    2 int temp = sum (n-1);  
  
    3 return temp + n;  
}
```

Sum(n) ✓

↓
Sum(n-1) ✓

↓
Sum(n-2) ✓

↓
Sum(n-3) ✓

↓
Sum(n-4) ✓

↓
Sum(n-5) ✓

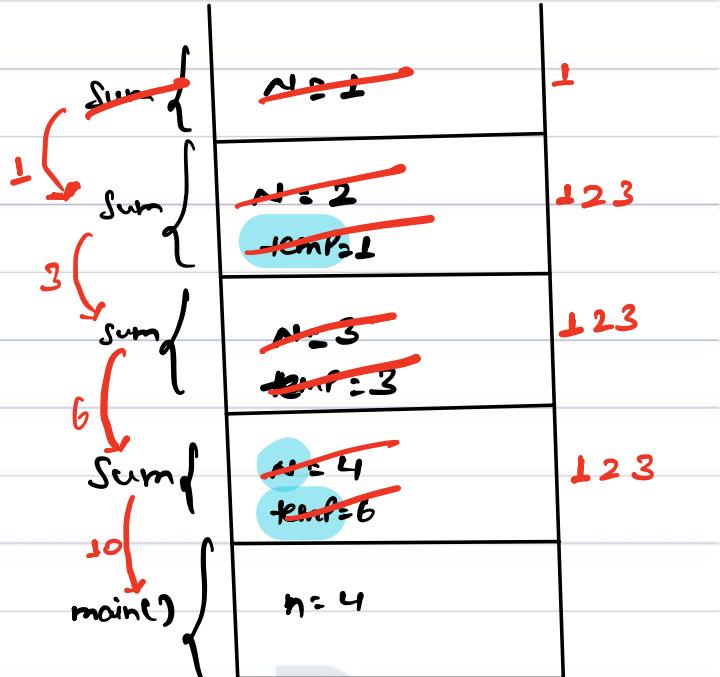
↓
Sum(n-6) ✓

↓
Sum(n-7) ✓

↓
Sum(n-8) ✓

↓
Sum(n-9) ✓

↓
Sum(n-10) ✓





Q) find factorial of n .

$$\text{Ex: } n=3 \rightarrow 3 \times 2 \times 1 = 6$$

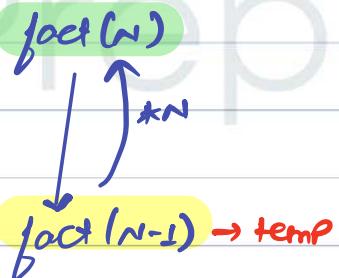
$$n=4 \rightarrow 4 \times 3 \times 2 \times 1 = 24$$

```
int fact(int n){  
    if(n==1){ return 1; }  
}
```

Goal: Given n , calculate &
return factorial of n .

```
int temp = fact(n-1);  
return n * temp;
```

Main logic:



3

```
fact(4) 6 * 4 = 24  
| 6  
fact(3) 2 * 3 = 6  
| 2  
fact(2) 1 * 2 = 2  
| 1  
fact(1)
```

Base case:

$$\text{fact}(1) = 1$$

Break till 9:37 PM



int fact (int n) {
 1 if (n == 1) { return 1; }
}

2 int temp = fact (n - 1);

3 return N * temp;

3

fact

1

fact

2

fact

3

fact

4

N = 1

N = 2

N = 3

N = 4

1

1 2 3

1 2 3

1 2 3



AlgoPrep



Q) Print N^{th} fibonacci number

Ex: 0 1 2 3 4 5 6 7 8 9 10
 0 1 1 2 3 5 8 13 21 34 55

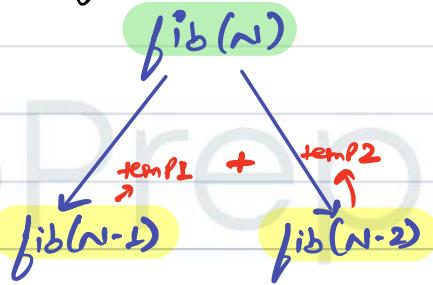
$$fib(n) = fib(n-1) + fib(n-2);$$

```
int fib (int n){  
    if (n==0 || n==1) return n;  
}
```

Faith: Given n , calculate & return n^{th} fibonacci number.

```
int temp1 = fib(n-1);  
int temp2 = fib(n-2);  
return temp1 + temp2;
```

main logic:



3

Base Case:

$$\begin{aligned} \hookrightarrow fib(0) &= 0 \\ \hookrightarrow fib(1) &= 1 \end{aligned}$$



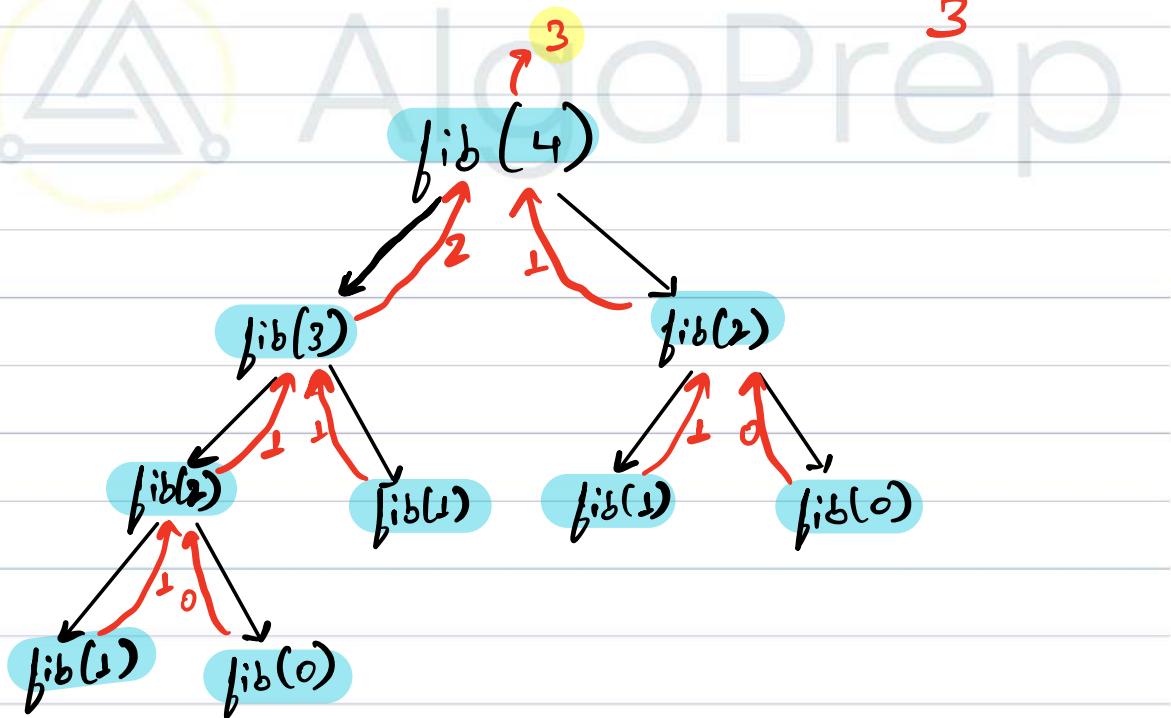
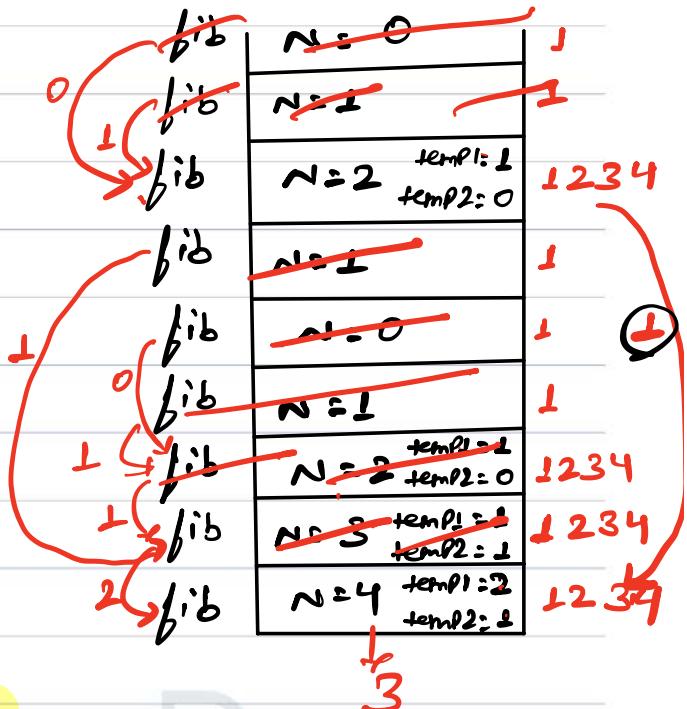
```
int fib (int N) {
    1 if(N==0 || N==1) return N; }
```

2 int temp1 = fib(N-1);

3 int temp2 = fib(N-2);

4 return temp1 + temp2;

}





Q) Point increasing

↳ Given n , Point all the numbers from $1 \rightarrow n$.

$$n=5 \rightarrow 1 \ 2 \ 3 \ 4 \ 5$$

```
void inc (int n){  
    if (n==1) { Point(1);  
        return; }  
    inc (n-1);  
    s.o.p(n);  
    return; }
```

Faith: Given n , Point no. from $1 \rightarrow n$.

Main logic:

{ $1, 2, 3, 4, \dots, n-1, n$ } $\{n\}$

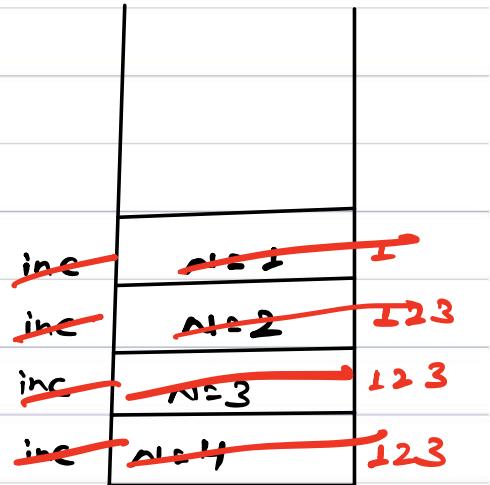
3

Base case:

↳ Print (1) for $n=1$.



```
void inc(int n){  
    if (n==1) { print(1);  
        return; }  
    2 inc(n-1);  
    3 s.o.p(n);  
    return;  
3
```



1 2 3 4

↳ you didn't come this far only to come this far.

Sum

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int ans = sum(n);
        System.out.println(ans);
    }

    public static int sum(int n){
        if(n == 1){
            return 1;
        }

        int temp = sum(n-1);
        return temp + n;
    }
}
```

C++ Code:

```
#include <iostream>

using namespace std;

int sum(int n) {
    if (n == 1) {
        return 1;
    }

    int temp = sum(n - 1);
    return temp + n;
}
```

```
int main() {
    int n;
    cin >> n;

    int ans = sum(n);
    cout << ans << endl;

    return 0;
}
```

Python Code:

```
def sum_recursive(n):
    if n == 1:
        return 1

    temp = sum_recursive(n - 1)
    return temp + n

if __name__ == "__main__":
    n = int(input())
    ans = sum_recursive(n)
    print(ans)
```

Factorial

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int ans = factorial(n);
```

```
        System.out.println(ans);
    }

public static int factorial(int n){
    if(n == 1){
        return 1;
    }

    int temp = factorial(n-1);
    return temp*n;
}
}
```

C++ Code:

```
#include <iostream>

using namespace std;

int factorial(int n) {
    if (n == 1) {
        return 1;
    }

    int temp = factorial(n - 1);
    return temp * n;
}

int main() {
    int n;
    cin >> n;

    int ans = factorial(n);
    cout << ans << endl;

    return 0;
}
```

Python Code:

```
def factorial(n):
    if n == 1:
        return 1

    temp = factorial(n - 1)
    return temp * n

if __name__ == "__main__":
    n = int(input())
    ans = factorial(n)
    print(ans)
```

Fibonacci

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        int ans = fibonacci(n);
        System.out.println(ans);
    }

    public static int fibonacci(int n){
        if(n == 0 || n == 1){
            return n;
        }

        int temp1 = fibonacci(n-1);
        int temp2 = fibonacci(n-2);
        return temp1 + temp2;
    }
}
```

C++ Code:

```
#include <iostream>

using namespace std;

int fibonacci(int n) {
    if (n == 0 || n == 1) {
        return n;
    }

    int temp1 = fibonacci(n - 1);
    int temp2 = fibonacci(n - 2);
    return temp1 + temp2;
}

int main() {
    int n;
    cin >> n;

    int ans = fibonacci(n);
    cout << ans << endl;

    return 0;
}
```

Python Code:

```
def fibonacci(n):
    if n == 0 or n == 1:
        return n

    temp1 = fibonacci(n - 1)
    temp2 = fibonacci(n - 2)
    return temp1 + temp2

if __name__ == "__main__":
    n = int(input())
    ans = fibonacci(n)
    print(ans)
```

Print Increasing

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        inc(n);

    }

    public static void inc(int n){
        if(n == 1){
            System.out.println(1);
            return;
        }

        inc(n-1);
        System.out.println(n);

    }
}
```

C++ Code:

```
#include <iostream>

using namespace std;

void inc(int n) {
    if (n == 1) {
        cout << 1 << endl;
        return;
    }
```

```
}

inc(n - 1);
cout << n << endl;
}

int main() {
    int n;
    cin >> n;

    inc(n);

    return 0;
}
```

Python Code:

```
#include <iostream>

void inc(int n) {
    if (n == 1) {
        std::cout << 1 << std::endl;
        return;
    }

    inc(n - 1);
    std::cout << n << std::endl;
}

int main() {
    int n;
    std::cin >> n;

    inc(n);

    return 0;
}
```

Print Decreasing HW

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();

        dec(n);

    }

    public static void dec(int n){
        if(n == 1){
            System.out.println(1);
            return;
        }

        System.out.println(n);
        dec(n-1);
    }
}
```

C++ Code:

```
#include <iostream>

void dec(int n) {
    if (n == 1) {
        std::cout << 1 << std::endl;
        return;
    }

    std::cout << n << std::endl;
    dec(n - 1);
}
```

```
int main() {
    int n;
    std::cin >> n;

    dec(n);

    return 0;
}
```

Python Code:

```
def dec(n):
    if n == 1:
        print(1)
        return

    print(n)
    dec(n - 1)

if __name__ == "__main__":
    n = int(input())

    dec(n)
```



Today's agenda

↳ Pow(a, n)

↳ TC & SC of recursion



AlgoPrep



Q) Given a and n , calculate a^n .

Ex: $a \ n$

$$2 \ 3 \longrightarrow 8$$

$$3 \ 5 \longrightarrow 243$$

```
int Pow (int a, int n){  
    if (n == 1) { return a; }  
}
```

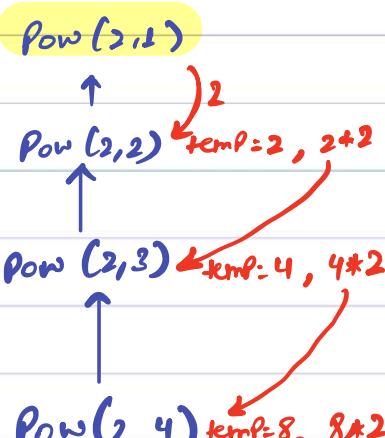
Faith: Given a and n , calculate and return a^n .

int temp = Pow(a, n-1);
return temp*a;

main logic:



base case:



$$n = 0 \rightarrow 1$$

$$n = 1 \rightarrow a$$

$$\text{No. of functions} = 4 \rightarrow n$$

$$\text{T.C of one function} = O(1)$$

$$\text{Overall T.C: } O(1) \times n = O(n)$$

16



// Give me something better than $O(n)$.

$$\textcircled{1} \quad a^n = a^{n-1} * a$$

$$\begin{array}{c} 2^5 \\ \downarrow \\ 2^4 \\ \downarrow \\ 2^3 \\ \downarrow \\ 2^2 \\ \downarrow \\ 2^1 \end{array}$$

$$a^n \rightarrow a^{\frac{n}{2}} \xrightarrow{\text{temp}}$$

↳ if $n = \text{even} \rightarrow a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}}$ $\xrightarrow{\text{temp}}$ $\xrightarrow{\text{temp}}$

↳ if $n = \text{odd} \rightarrow a^n = a^{\frac{n-1}{2}} * a^{\frac{n-1}{2}} * a$

$$a^{n-1} \text{ & } a^{\frac{n-1}{2}} = a^n$$

$$2^{16} \xrightarrow{\text{temp: 256}} 256 + 256 = 512$$

$$2^8 \xrightarrow{\text{temp: 16}} 16 + 16 = 32$$

$$2^4 \xrightarrow{\text{temp: 4}} 4 + 4 = 8$$

$$2^2 \xrightarrow{\text{temp: 2}} 2 + 2 = 4$$

$$2^{17} \rightarrow 256 + 256 + 2$$

$$2^8$$



$$a^n \rightarrow a^{\frac{n}{2}}$$

if $n = \text{even} \rightarrow a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}}$

if $n = \text{odd} \rightarrow a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}} * a$

```
int Pow (int a, int n){  
    if (n == 1) { return a; }  
}
```

Faith: Given a and n , calculate and return a^n .

T.C: $O(\log n)$

int temp = Pow(a, $\frac{n}{2}$);

if ($n \% 2 == 0$) { return temp * temp; }

else { return temp * temp * a; }

main logic? $a^n \rightarrow a^{\frac{n}{2}}$

if $n = \text{even} \rightarrow a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}}$

if $n = \text{odd} \rightarrow a^n = a^{\frac{n}{2}} * a^{\frac{n}{2}} * a$

base case:

$a = 2$ $n = -3$

$$\frac{1}{2^3}$$



```
int Pow (int a, int n){
```

```
    if (n == 1) { return a; }
```

$a=2 \quad n=37$



```
    int temp = Pow(a, n/2);
```

```
    if (n%2 == 0) { return temp*temp; }
```

$n=18$

```
    else { return temp*temp*a; }
```

$n=9$

}

$n=4$

$n=2$

$n=1$

Overall T.C: $(T.C \text{ of } \uparrow \text{function})^{\log N} \times O(1)$

$\approx O(\log N)$

$$\rightarrow a^n = a^{n/2} * a^{n/2}$$

$$a^n \approx a^{n/3} * a^{n/3} + a^{n/3}$$



TC of recursive code

Overall T.C: $(T.C \text{ of } 1 \text{ function})^*$
 $\xrightarrow{o(1)}$
(Total no. of function)

```
int sum (int n) {  
    if (n==1) return 1; 3  
}
```

T.C of 1 function: $O(1)$
No. of function: N

int temp = sum (N-1);

return temp + N;

3

S.C: $O(N)$

Total T.C: $O(N)$

Sum {	$n=1$	$\rightarrow O(1)$
Sum {	$n=2$	$\rightarrow O(1)$
Sum {	$n=3$	$\rightarrow O(1)$
Sum {	$n=4$	$\rightarrow O(1)$
Sum {	$n=5$	$\rightarrow O(1)$

↳ If you delete a space that is already considered in your space complexity and create new space in place of previous space, you are not increasing your space complexity.

→ T.C of fibonacci



```
int fib (int N){  
    if(N==0 || N==1) return N;  
    int temp1 = fib(N-1);  
    int temp2 = fib(N-2);  
    return temp1 + temp2;
```

T.C of 1 function: $O(1)$

No. of function: $2^N \Rightarrow O(2^N)$

Total T.C:

S.C:
1
3



return temp1 + temp2;

$N=6 \quad 1 \rightarrow 2^0 \rightarrow 1$

$+ \quad 2 \rightarrow 2^1 \rightarrow 2$

$+ \quad 4 \rightarrow 2^2 \rightarrow 4$

$+ \quad 8 \rightarrow 2^3 \rightarrow 8$

$+ \quad 16 \rightarrow 2^4 \rightarrow 16$

$+ \quad 32 \rightarrow 2^5 \rightarrow 32$

$\vdots \quad \vdots \quad \vdots \quad \vdots$

$$2^0 + 2^1 + 2^2 + \dots + 2^{N-1} = \text{Sum of G.P: } a * \frac{(x^n - 1)}{x - 1}$$

$$= \frac{1 * 2^N - 1}{2 - 1}$$

$$= 2^N - 1$$

Total S.C: (Space complexity of one function) \times max no. of levels.



→ S.C: $O(N)$



$$\begin{aligned} \text{No. of Calls} &= \alpha \\ \text{Count of levels} &= n \end{aligned} \Rightarrow \text{No. of functions} = \alpha^n$$

Break till 9:42 Pm

Space Complexity:

↳ How much Space are you using.



iterative: variables, Array, String
+ O(j) + nSize + O(n)

recursive: variables, Array, String, Call Stack

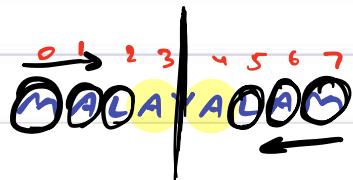
Note: Input or output won't be considered in Space Complexity (Big O notation).



Q) Given an array, check if it is Palindrome or not?
 ↗ character
 ↗ recursion

Ex: MALAYALAM → true

Ex: aabba → false



```
boolean isPalindrome (char ch[], int s, int e){  

    if (s == e) { return true; }
```

→ Task: Check and return whether

if ($ch[s] == ch[e]$) {

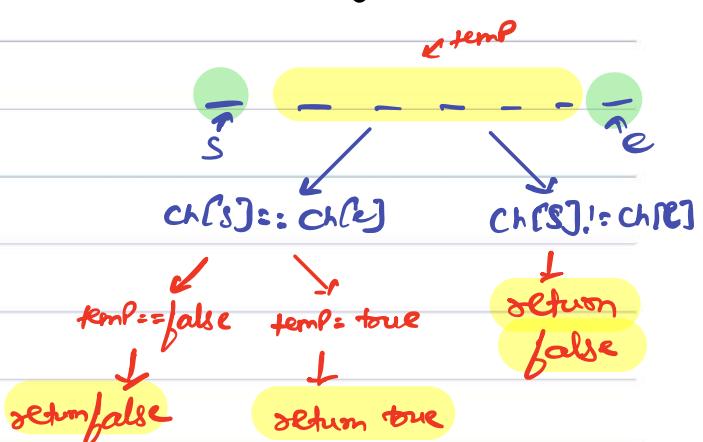
Ch array is Palindrome
between s to e.

boolean temp = isPalindrome (ch, s+1, e-1);

Main logic:

return temp;

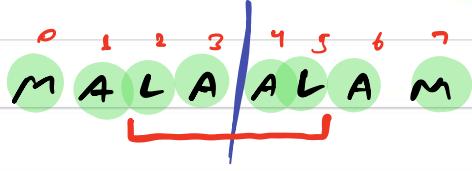
} else { return false; }



base case:

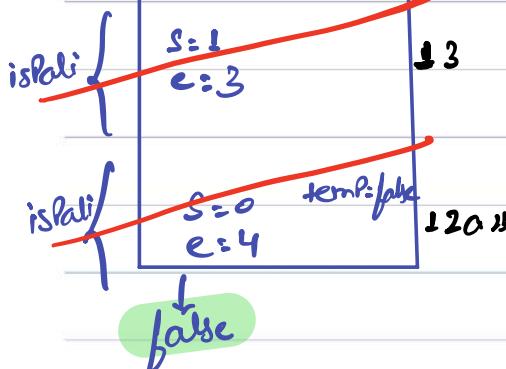
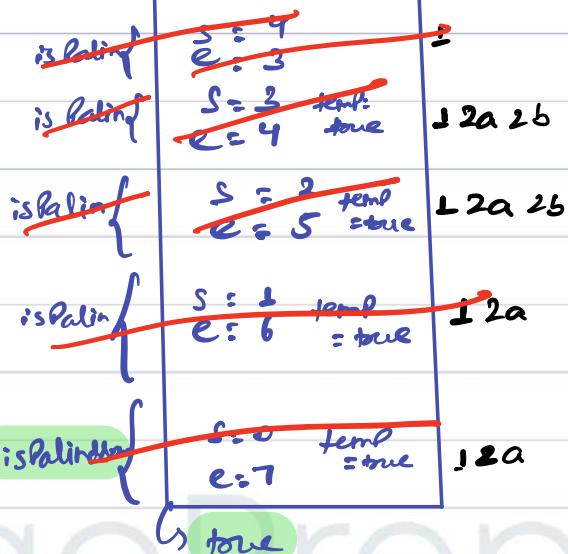


```
boolean isPalindrome (char ch[], int s, int e){  
    if (s == e || s > e) { return true; }
```



```
2 if (ch[s] == ch[e]) {  
    a boolean temp = isPalindrome (ch, s+1, e-1);  
    b return temp; }  
3 else { return false; }
```

a b c d Q



```
boolean isPalindrome (char ch[], int s, int e){  
    if (s == e || s > e) { return true; }
```

```
2 if (ch[s] == ch[e]) {  
    a boolean temp = isPalindrome (ch, s+1, e-1);  
    b return temp; }  
3 else { return false; }
```

T.C: $O(1) * \frac{n}{2} \approx O(n/2) \approx O(n)$

S.C: $O(1) * \frac{n}{2} \approx O(n/2) = O(n)$

Today's agenda

↳ quizzes & learning

Quiz 1:

$\rightarrow \text{int } n = 10;$

Syntax to declare the variable?

- a) type variable-name = value ; X
- b) variable-name type = value ;
- c) variable-name = value ;
- d) None of the above.

Quiz 2:

Declare an int variable with name y and value 20?

a) Int y = 20; xx

$\hookrightarrow \text{int } y = 20;$ (c)

Quiz 3

$\text{int } a = 10;$

$\boxed{10}$
a

$\text{int } b = 20;$

$\boxed{20}$
b

$\text{System.out.println}(a+b); \rightarrow 30$

Quiz 4:

int a = 20;

int b = 30;

→ removed from
radar board

System.out.println(a + " " + b);

↳ 20 30

Quiz 5:

int temp: 10;

↑
①

System.out.println(Temp);

↳ error

Quiz 6:

System.out.println(a);

int a = 10;

↳ error

Quiz 7:

```
int a = 20;
```

```
int a = 30;
```

```
System.out.println(a);
```

↳ error

you can't initialize

same variable name

twice?

Quiz 8:

```
int n = 20;
```

```
System.out.println(n);
```

```
n = 40;
```

```
System.out.println(n);
```



2040



Quiz 9:

```
int a = 10;
```



10 = 10

```
int b = 20;
```

↳ a = ~~b~~ + 30;

LHS = RHS

```
System.out.println(a);
```

→ 50

LHS $\stackrel{+}{=}$ RHS

↳ Computer \rightarrow = \rightarrow assignment operator

↳ assign right side

value to left side.

* Variable naming rules

1. Name can contain lowercase, uppercase alphabets or digit (0-9) or '\$' (dollar) or '-' (underscore)

2. First letter / character of the name cannot be number.

3. Can't use reserved keywords as variable name:

Reserved word: words which are already predefined in java. ex: Public, Static, void, int etc.

Quiz 10:

int m = 10; ✗

int 1y = 20; ✗

⇒ ans = 1

int y@b = 30; ✗

Q: How many of above are correct variable names?

Quiz 11:

int -y = 10; ✗

int any = 20; ✗

⇒ ans = 2

int or a = 30; ✗

int y\$2 = 45; ✗

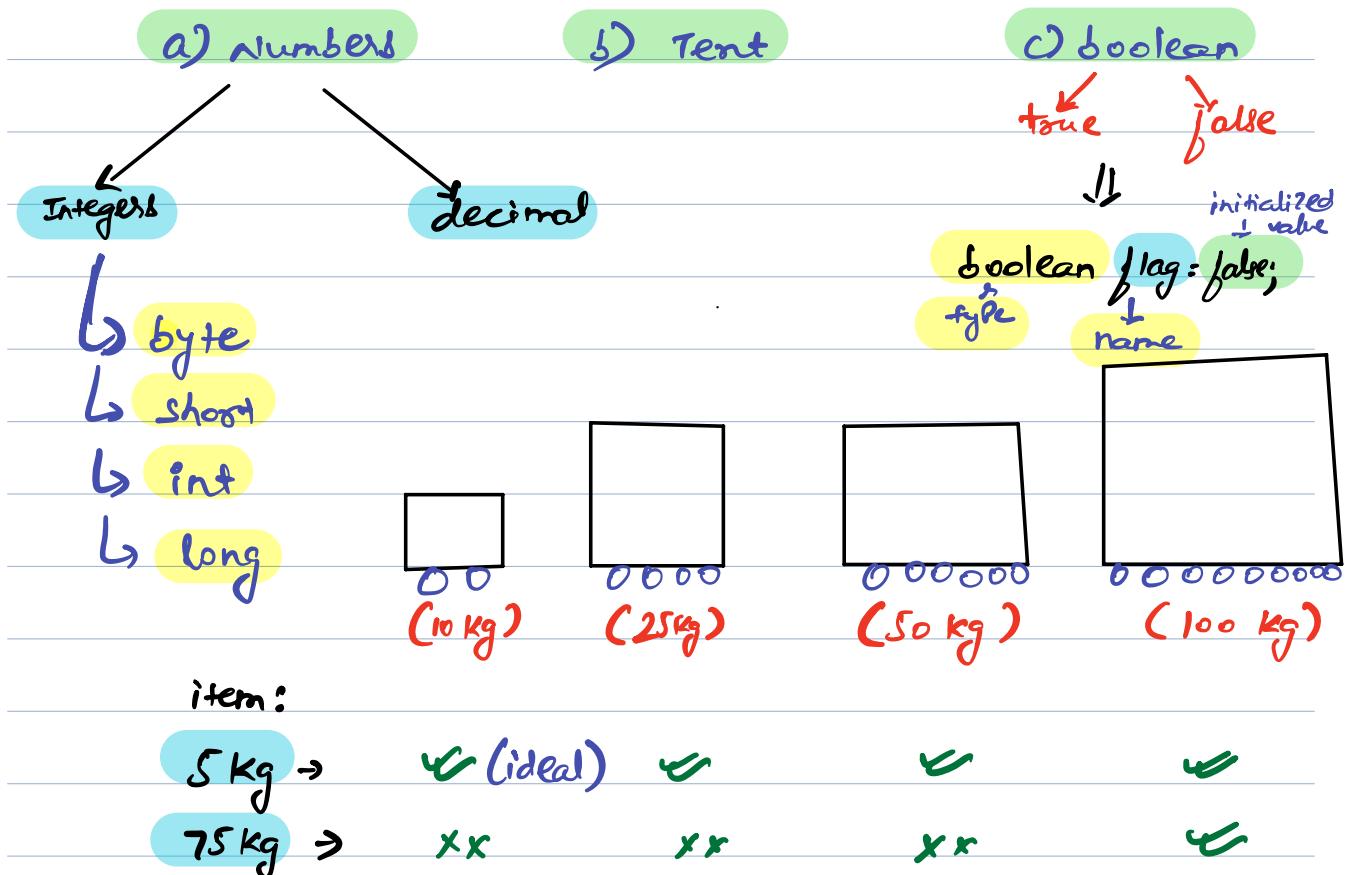
Q How many of above are correct variable names?

Quiz 12:

int Static = 60; → error

System.out.println(Static);

Different Categories of data



Range / capacity

Capacity	byte short int → $\{-2^{31} \dots 2^{31}\}$ long → $\{-2^{63} \dots 2^{63}\}$	$-2^7 \dots 2^7 - 1 : \{-128, 127\}$
range		$-2^{15} \dots 2^{15} - 1 : \{-32768, 32767\}$
are different		$-2^{31} \dots 2^{31} - 1 : \{-214\dots, 214748\dots\}$
		$-2^{63} \dots 2^{63} - 1 : \{ \dots, \dots \}$

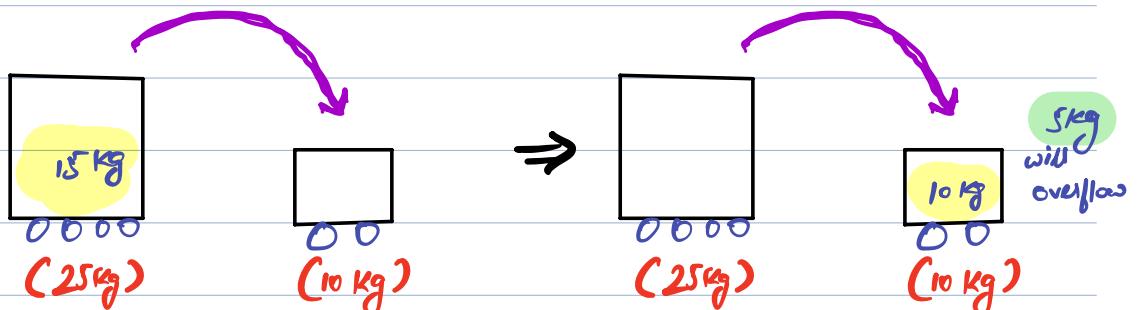
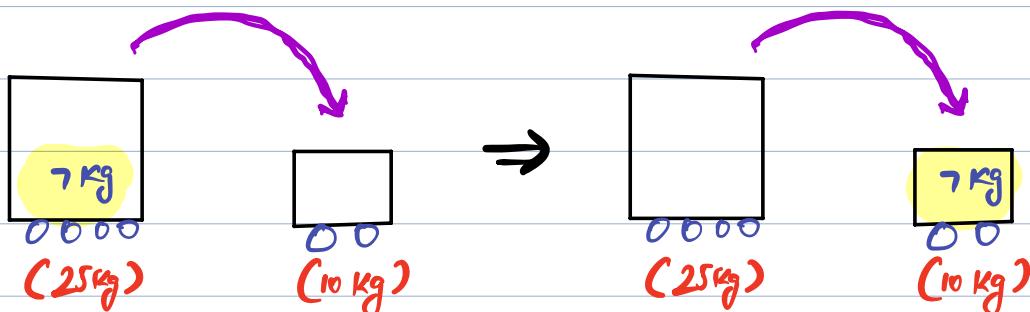
Complete explanation is in Bit manipulation Class.

↳ The default value is int. To initialize long you need long $\alpha = 100000000000L$.

Break till 9:29 PM

↳ you didn't come this far only to come this far.

→ Type Casting



Approach

int $\rightarrow \{-2^{31} \text{ to } 2^{31}-1\} \rightarrow \{-2 \times 10^9 \text{ to } 2 \times 10^9\}$
long $\rightarrow \{-2^{63} \text{ to } 2^{63}-1\} \rightarrow \{-2 \times 10^{18} \text{ to } 2 \times 10^{18}\}$

ex1: int n = 100

long y = n; \rightarrow implicit

System.out.println(y);

ex2:

long n = 1000;

explicitly \leftarrow int y = (int)n;

System.out.println(y); \rightarrow 1000

1000
n

1000
y

Quiz 13

int $a = 10000;$

long $b = a;$

System.out.println (b);

6 10000

Quiz 14:

long $a = 100000;$ → int $y = (int)a$

int $y = a;$ → Typecast?

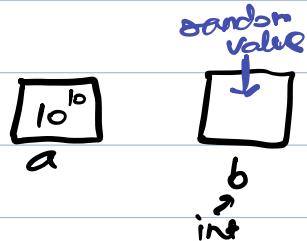
System.out.println (y);

Quiz 15:

long $a = 10^{10};$

int $b = (int)a;$

System.out.println (b);



* reading Inputs

↳ Scanner ^{name} scn = new Scanner (System.in);
int n = scn.nextInt();
System.out.println (n);

Quiz 16:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

System.out.println (n);

b error

(Small S in
System)

Quiz 17:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

small S

System.out.println (n);

b error

Quiz 18:

input: 30

Scanner Scn = new Scanner (System.in);

int n: Scn. nextInt();

System.out.println (n);

b 30

Quiz 19:

Input: 24 30

Scanner Scn = new Scanner (System.in);

int x = Scn.nextInt(); ← 24

int y = Scn.nextInt(); ← 30

System.out.println(x); → 24

Quiz 20:

Input: 24 30

Scanner Scn = new Scanner (System.in);

int x = Scn.nextInt();

int y = Scn.nextInt();

int z = Scn.nextInt();

System.out.println(x+y+z);

error

↳ Done with today's class

Quiz 21 :

float $n = 3.4f;$

double $y = n;$

System.out.println(y);

Quiz 22 :

int $n = 40;$

double $y = n;$

System.out.println(y);

Quiz 23 :

double $n = 3.45;$

→ error

int $y = n;$

System.out.println(y);

Smart Power

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        System.out.println(power(a,n));
    }

    public static long power (int a, int n){
        if (n == 1){
            return a;
        }

        long temp = power(a,n/2);
        if(n%2 == 0){
            return temp*temp;
        }else{
            return temp*temp*a;
        }
    }

}
```

C++ Code:

```
#include <iostream>

long power(int a, int n) {
    if (n == 1) {
        return a;
    }

    long temp = power(a, n / 2);
    if (n % 2 == 0) {
```

```
        return temp * temp;
    } else {
        return temp * temp * a;
    }
}

int main() {
    int a, n;
    std::cin >> a >> n;

    std::cout << power(a, n) << std::endl;

    return 0;
}
```

Python Code:

```
def power(a, n):
    if n == 1:
        return a

    temp = power(a, n // 2)
    if n % 2 == 0:
        return temp * temp
    else:
        return temp * temp * a

if __name__ == "__main__":
    a, n = map(int, input().split())

    print(power(a, n))
```

Is Palindrome

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String st = scn.nextLine();
        System.out.println(IsPalin(st,0,st.length()-1));
    }

    public static boolean IsPalin(String st, int s, int e){
        if(s >= e){
            return true;
        }

        if(st.charAt(s)==st.charAt(e)){
            boolean temp = IsPalin(st,s+1,e-1);
            return temp;
        }else{
            return false;
        }
    }
}
```

C++ Code:

```
#include <iostream>
#include <string>

bool IsPalin(const std::string& st, int s, int e) {
    if (s >= e) {
        return true;
    }

    if (st[s] == st[e]) {
        bool temp = IsPalin(st, s + 1, e - 1);
        return temp;
    } else {
        return false;
    }
}
```

```

    }
}

int main() {
    std::string st;
    std::getline(std::cin, st);

    std::cout << (IsPalin(st, 0, st.length() - 1) ? "true" : "false") << std::endl;

    return 0;
}

```

Python Code:

```

def IsPalin(st, s, e):
    if s >= e:
        return True

    if st[s] == st[e]:
        temp = IsPalin(st, s + 1, e - 1)
        return temp
    else:
        return False

if __name__ == "__main__":
    st = input()

    print("true" if IsPalin(st, 0, len(st) - 1) else "false")

```

Count of aaa_HW

Solution Vid:

<https://youtu.be/VWL9dMiO0MY>

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        String st = scn.next();
        System.out.println(count1(st,0,0));
        System.out.println(count2(st,0,0));
    }

    private static int count2(String st, int vidx, int count) {
        if(vidx>=st.length()-2){
            return count;
        }
        if(st.charAt(vidx)=='a' &&
st.charAt(vidx)==st.charAt(vidx+1)&&st.charAt(vidx+1)==st.charAt(vidx+2)){
            int temp = count2(st,vidx+3,count+1);
            return temp;
        }
        else{
            int temp = count2(st,vidx+1,count);
            return temp;
        }
    }

    private static int count1(String st, int vidx, int count) {
        if(vidx==st.length()-2){
            return count;
        }
        if(st.charAt(vidx)=='a' &&
st.charAt(vidx)==st.charAt(vidx+1)&&st.charAt(vidx+1)==st.charAt(vidx+2)){
            int temp=count1(st,vidx+1,count+1);
            return temp;
        }else{
```

```

        int temp=count1(st,vidx+1,count);
        return temp;
    }
}
}

```

C++ Code:

```

#include <iostream>
#include <string>

int count2(const std::string& st, int vidx, int count) {
    if (vidx >= st.length() - 2) {
        return count;
    }
    if (st[vidx] == 'a' && st[vidx] == st[vidx + 1] && st[vidx + 1] == st[vidx + 2]) {
        int temp = count2(st, vidx + 3, count + 1);
        return temp;
    } else {
        int temp = count2(st, vidx + 1, count);
        return temp;
    }
}

int count1(const std::string& st, int vidx, int count) {
    if (vidx == st.length() - 2) {
        return count;
    }
    if (st[vidx] == 'a' && st[vidx] == st[vidx + 1] && st[vidx + 1] == st[vidx + 2]) {
        int temp = count1(st, vidx + 1, count + 1);
        return temp;
    } else {
        int temp = count1(st, vidx + 1, count);
        return temp;
    }
}

int main() {
    std::string st;
    std::cin >> st;

    std::cout << count1(st, 0, 0) << std::endl;
    std::cout << count2(st, 0, 0) << std::endl;

    return 0;
}

```

Python Code:

```
def count2(st, vidx, count):
    if vidx >= len(st) - 2:
        return count
    if st[vidx] == 'a' and st[vidx] == st[vidx + 1] and st[vidx + 1] == st[vidx + 2]:
        temp = count2(st, vidx + 3, count + 1)
        return temp
    else:
        temp = count2(st, vidx + 1, count)
        return temp

def count1(st, vidx, count):
    if vidx == len(st) - 2:
        return count
    if st[vidx] == 'a' and st[vidx] == st[vidx + 1] and st[vidx + 1] == st[vidx + 2]:
        temp = count1(st, vidx + 1, count + 1)
        return temp
    else:
        temp = count1(st, vidx + 1, count)
        return temp

if __name__ == "__main__":
    st = input()

    print(count1(st, 0, 0))
    print(count2(st, 0, 0))
```

Check Sorted_HW

Solution Vid:

https://youtu.be/_eMQzWCnEyk

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int m = scn.nextInt();
        int[] arr = new int[m];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = scn.nextInt();
        }
        System.out.println(sorted(arr, 0));
    }

    private static boolean sorted(int[] arr, int vidx) {
        if (vidx == arr.length - 1) {
            return true;
        }

        boolean check = sorted(arr, vidx + 1);

        if(check == false){
            return false;
        }else{
            if(arr[vidx] <= arr[vidx+1]){
                return true;
            }else{
                return false;
            }
        }
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>

bool sorted(const std::vector<int>& arr, int vidx) {
    if (vidx == arr.size() - 1) {
        return true;
```

```

}

bool check = sorted(arr, vidx + 1);

if (!check) {
    return false;
} else {
    if (arr[vidx] <= arr[vidx + 1]) {
        return true;
    } else {
        return false;
    }
}

int main() {
    int m;
    std::cin >> m;
    std::vector<int> arr(m);

    for (int i = 0; i < arr.size(); i++) {
        std::cin >> arr[i];
    }

    std::cout << sorted(arr, 0) << std::endl;
}

return 0;
}

```

Python Code:

```

def sorted(arr, vidx):
    if vidx == len(arr) - 1:
        return True

    check = sorted(arr, vidx + 1)

    if not check:
        return False
    else:
        if arr[vidx] <= arr[vidx + 1]:
            return True
        else:
            return False

```

```

if __name__ == "__main__":
    m = int(input())
    arr = list(map(int, input().split()))

    print(int(sorted(arr, 0)))

```

First Index_HW

Solution Vid:

<https://youtu.be/L6blvkbX6XM>

Java Code:

```

import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int m = scn.nextInt();
        int[] arr = new int[m];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = scn.nextInt();
        }
        int n = scn.nextInt();
        int z = firstindex(arr, 0, n);
        System.out.println(z);
    }

    private static int firstindex(int[] arr, int vdx, int n) {
        if (vdx == arr.length) {
            return -1;
        }

        int index = firstindex(arr, vdx + 1, n);
        if(arr[vdx] == n){
            return vdx;
        }else{
            return index;
        }
    }
}

```

```
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <vector>  
  
int firstIndex(const std::vector<int>& arr, int vdx, int n) {  
    if (vdx == arr.size()) {  
        return -1;  
    }  
  
    int index = firstIndex(arr, vdx + 1, n);  
    if (arr[vdx] == n) {  
        return vdx;  
    } else {  
        return index;  
    }  
}  
  
int main() {  
    int m;  
    std::cin >> m;  
    std::vector<int> arr(m);  
  
    for (int i = 0; i < arr.size(); i++) {  
        std::cin >> arr[i];  
    }  
  
    int n;  
    std::cin >> n;  
  
    int z = firstIndex(arr, 0, n);  
    std::cout << z << std::endl;  
  
    return 0;  
}
```

Python Code:

```
def first_index(arr, vdx, n):  
    if vdx == len(arr):  
        return -1
```

```
index = first_index(arr, vdx + 1, n)
if arr[vdx] == n:
    return vdx
else:
    return index

if __name__ == "__main__":
    m = int(input())
    arr = list(map(int, input().split()))

n = int(input())

z = first_index(arr, 0, n)
print(z)
```



Today's agenda

- ↳ understanding sorting
- ↳ Problems on sorting
- ↳ Sorting techniques



AlgoPrep



Sorting: Arranging data in increasing /decreasing,
on what Parameter.

Ex1: 2 4 10 15 27 → true

Ex2: 20 7 3 -5 -8 → dec order → true

Ex3: 1 2 3 7 4 9 6 → Not sorted on the basis of value.
#factors: 1 2 2 2 3 3 4 → sorted on the basis of factor count.

bubble sort
Selection sort
insertion sort
merge sort
quick sort
bucket sort
radix sort
etc.

Steps to Solve Problem:

1 2 3 4
↓
Sort the array

↓
inbuilt Sorting function.

↓
Arrays.sort(arr); → inc. order

T.C: $O(n \log n)$ S.C: $O(1)$
Worst T.C: $O(n^2)$

↳ How in levelup.



Q) Order of Removal

Given n elements at every step remove an array element. Cost to remove element = Sum of array elements present. Find min cost to remove all elements.

Note: Add cost first and then remove.

$$\text{Ex1: } \text{arr}[3] = \{ \underset{0}{\cancel{x}}, \underset{1}{\cancel{x}}, \underset{2}{\cancel{x}} \} \quad \text{remove } 3 : 10$$

remove 2 : 7

remove 5 : $\frac{5}{22}$

$$\text{arr}[3] = \{ \underset{0}{\cancel{x}}, \underset{1}{\cancel{x}}, \underset{2}{\cancel{x}} \} \quad \text{remove } 2 : 10$$

remove 5 : 8

remove 3 : $\frac{3}{21}$

$$\text{Ex2: } \text{arr}[4] = \{ \underset{0}{\cancel{4}}, \underset{1}{\cancel{x}}, \underset{2}{\cancel{x}}, \underset{3}{\cancel{x}} \}$$

remove 7 :

remove 2 :

remove 6 :

remove 4 :

$$4 + 6 + 2 + 7$$

$$4 + 6 + 2$$

$$4 + 6$$

$$4$$

max Contribution: 0th index \rightarrow min

2nd max Contri: 1st index \rightarrow 2nd min

:

Array should be sorted in inc. order &

remove from left.



$$arr[4] = \{ \overset{0}{4} \overset{1}{6} \overset{2}{2} \overset{3}{7} \}$$



$$\{ \overset{0}{2} \overset{1}{4} \overset{2}{6} \overset{3}{7} \}$$

remove 7 :

$$2 + 4 + 6 + 7$$

adding frequency

remove 6 :

$$2 + 4 + 6$$

$$= N - i$$

remove 4 :

$$2 + 4$$

$$6 * 2$$

remove 2 :

$$2$$

$$4 * 3$$

$$2 + 4$$

//Pseudo Code

int orderOfRemoval (int arr[N]) {

 Arrays.sort (arr);

 int ans = 0;

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

 int temp = arr[i] * (N - i);

 ans = ans + temp;

 }

 return ans;

T.C: $O(N \log N) + O(N)$

~~$= O(N \log N + N)$~~

$= O(N \log N)$

S.C: $O(N)$

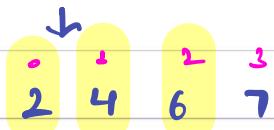
3



Tracing

```
int orderOfRemoval(int arr[N]) {
    Arrays.sort(arr);
    int ans = 0;
    for (int i = 0; i < N; i++) {
        int temp = arr[i] * (N - i);
        ans = ans + temp;
    }
    return ans;
}
```

$$arr[4] = \{ \begin{smallmatrix} 0 & 1 & 2 & 3 \\ 4 & 6 & 2 & 7 \end{smallmatrix} \}$$



$$ans = 0 + 8 + 12 + 12 + 7 = 39$$

0 $temp$

1 $2 * 4$

2 $4 * 3$

3 $6 * 2$

4 $7 * 1$



AlgoPrep



Q) Good Integer (only distinct)

Given $\text{arr}[n]$, calculate no. of good integers.

An element is Said to be good if

{No. of element < ele == ele itself}
 $\Rightarrow \text{arr}[i] == \text{arr}[j]$

Ex: $\{ -1^0, -4^1, 3^2, 5^3, -15^4, 4^5 \} \Rightarrow \text{ans: } 3$
less: $2^0, 1^1, 3^2, 5^3, 0^4, 4^5$

Ex: $\{ -1^0, -4^1, 3^2, 5^3, -15^4, 4^5 \}$
↓ sort in inc. order
 $\{ -15^0, -4^1, -1^2, 3^3, 4^4, 5^5 \}$

Count of elements smaller $\text{arr}[i] = i$
Element itself = $\text{arr}[i]$



//Pseudo Code

```
int goodIntegers (int arr[N]) {  
    Arrays.sort (arr);  
    int Count = 0;
```

T.C: $O(n \log n + n)$
 $= O(n \log n)$

S.C: $O(n)$

```
for (int i=0; i<n; i++) {  
    if (arr[i] == i) {Count++;}  
}
```

return Count;



{No. of element \leq ele \neq ele itself}
 \downarrow
 $i = \text{arr}[i]$

Good Integers : { Data can repeat }

last idea won't work directly.



Ex1: { 0 2 2 3 3 3 5 } → ans = 3
#less: 0 1 1 3 3 5

Ex2: { -4 -2 3 3 5 5 5 5 7 8 8 8 8 10 11 12 }
#less: 0 1 2 2 4 4 4 4 8 8 8 8 10 11 12
↳ ans: 3

Obs1: if element is the first occ. if $\text{arr}[i] \neq \text{arr}[i-1]$
 $i = \text{arr}[i]$
Count of ele < ele

Obs2: if element is the repeat element.

↳ count of ele < ele will remain same
as count of first occ.

Note: To check the first occ: $\text{arr}[i] \neq \text{arr}[i-1]$



//Pseudo Code

```
int goodInteger (int arr[N]) {  
    Arrays.sort (arr);  
    int count = 0;
```

int lessCount = 0;

//H.W → handle for 0th index

```
for (int i=1; i<N; i++) {  
    if (arr[i] != arr[i-1]) {  
        lessCount = i;  
    }  
    else if // nothing  
}
```

```
if (arr[i] == lessCount) {  
    count++;
```

}

3



int Count = 0;

int lessCount = 0;

```
for (int i=1; i<N; i++) {
    if (arr[i] != arr[i-1]) {
        lessCount = i;
    } else {
        // Nothing
    }
}
```

arr: { -4 2 3 3 5 5 5 5 8 8 8 10 12 }
less: 0 1 2 2 4 4 4 4 8 8 8 11 12

Count = 0

lessCount = 0

i	lessCount
1	1
2	2
3	2
4	4
5	4
6	4
7	4
8	8

Break till 9:50 PM



11 Sorting techniques

① Bubble sort

↳ Sort the array in asc. order but we can swap adjacent elements only.

$$\text{arr}(8) = \{ 5 \ 7 \ 5 \ 4 \ 10 \ -2 \ 6 \ 3 \}$$

$$\text{iter 0: } \{ 5 \ 7 \ 5 \ 4 \ 10 \ -2 \ 6 \ 3 \} \rightarrow \{ 0, n-2 \}$$

$$\text{iter 1: } \{ 5 \ 5 \ 4 \ 7 \ -2 \ 6 \ 3 \ 10 \} \quad \begin{matrix} \downarrow \\ i \end{matrix}$$
$$\{ 5 \ 5 \ 4 \ 7 \ -2 \ 6 \ 3 \ 10 \} \rightarrow \{ 0, n-3 \}$$

iter X:

↳ $n-1$ iterations



II Pseudo Code

```
void bubblesort (int arr[n]) {
```

```
    for (int i=0; i<n-1; i++) { // n-1 iterations
```

T.C: $O(n^2)$

S.C: $O(1)$

```
        for (int j=0; j<n-1-i; j++) {
```

```
            if (arr[j] > arr[j+1]) {
```

```
                int temp = arr[j];
```

```
                arr[j] = arr[j+1];
```

```
                arr[j+1] = temp;
```

 }

 }

 }



3



```

void bubbleSort (int arr[n]) {
    for (int i=0; i<N-1; i++) { 1 N-1 iterations
        for (int j=0; j<N-1-i; j++) {
            if (arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}

```

$\text{arr}(8) = \{ 5, 2, 2, 3, 4, 5, 6, 7 \}$

i
0
 j
 $\downarrow 0 \leq 6$
 $\cancel{8} \rightarrow 0$

$\downarrow j$
 $\downarrow 0, 5$

$\{ 5, 4, 5, -2, 6, 3, 7, 10 \}$

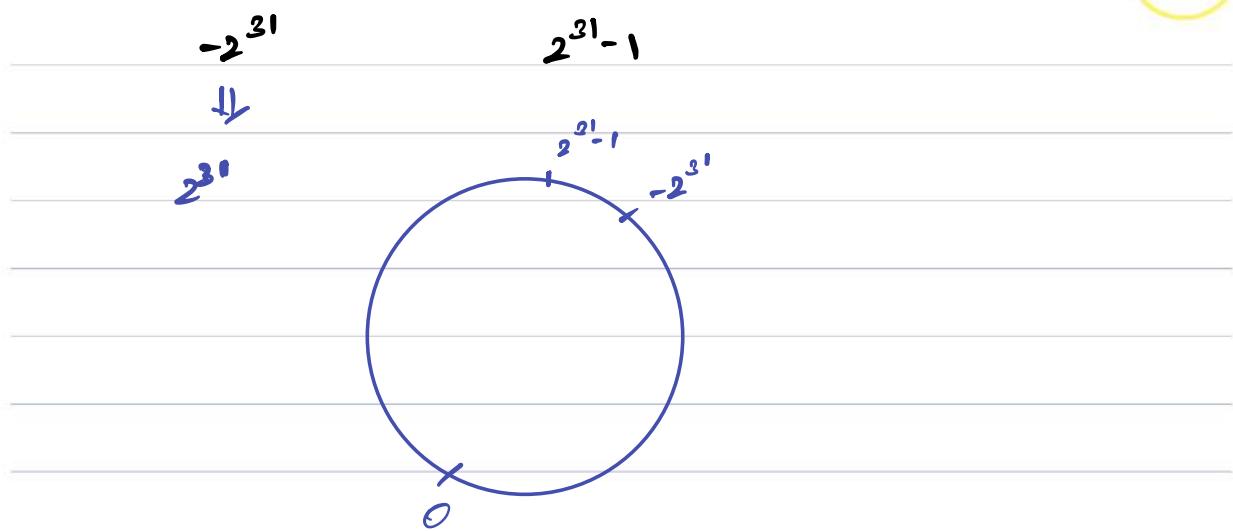
N elements

$N-1$ elements fix

\downarrow
Nth will be fixed
auto

$[0, N-2] \rightarrow N-1$ iterations

you didn't come this far only to come this far.



Order of Removal

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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();
        }
        System.out.println(orderOfRemoval(arr));
    }

    public static int orderOfRemoval(int[] arr){
        int n = arr.length;
        Arrays.sort(arr);
        int ans =0;

        for(int i=0;i<arr.length;i++){
            int temp = arr[i] * (n-i);
            ans = ans + temp;
        }

        return ans;
    }
}
```

C++ Code:

```
#include <iostream>
#include <algorithm>

int orderOfRemoval(int arr[], int n) {
    std::sort(arr, arr + n);
    int ans = 0;
```

```

        for (int i = 0; i < n; i++) {
            int temp = arr[i] * (n - i);
            ans += temp;
        }

        return ans;
    }

int main() {
    int n;
    std::cin >> n;

    int arr[n];
    for (int i = 0; i < n; i++) {
        std::cin >> arr[i];
    }

    std::cout << orderOfRemoval(arr, n) << std::endl;

    return 0;
}

```

Python Code:

```

def orderOfRemoval(arr):
    n = len(arr)
    arr.sort()
    ans = 0

    for i in range(n):
        temp = arr[i] * (n - i)
        ans += temp

    return ans

if __name__ == "__main__":
    n = int(input())
    arr = list(map(int, input().split()))

    print(orderOfRemoval(arr))

```

Good Integers Distinct

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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();
        }
        System.out.println(GoodIntegers(arr));
    }

    public static int GoodIntegers(int[] arr){
        int n = arr.length;
        Arrays.sort(arr);
        int ans =0;

        for(int i=0;i<arr.length;i++){
            if(arr[i] == i){
                ans++;
            }
        }

        return ans;
    }
}
```

C++ Code:

```
#include <iostream>
#include <algorithm>

int GoodIntegers(int arr[], int n) {
    std::sort(arr, arr + n);
    int ans = 0;
```

```

        for (int i = 0; i < n; i++) {
            if (arr[i] == i) {
                ans++;
            }
        }

        return ans;
    }

int main() {
    int n;
    std::cin >> n;

    int arr[n];
    for (int i = 0; i < n; i++) {
        std::cin >> arr[i];
    }

    std::cout << GoodIntegers(arr, n) << std::endl;

    return 0;
}

```

Python Code:

```

def GoodIntegers(arr):
    n = len(arr)
    arr.sort()
    ans = 0

    for i in range(n):
        if arr[i] == i:
            ans += 1

    return ans

if __name__ == "__main__":
    n = int(input())
    arr = list(map(int, input().split()))

    print(GoodIntegers(arr))

```

Good Integers Duplicate

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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();
        }
        System.out.println(GoodIntegers(arr));
    }

    public static int GoodIntegers(int[] arr){
        int n = arr.length;
        Arrays.sort(arr);
        int ans =0;
        int lesscount = 0;

        if(arr[0] == 0){
            ans++;
        }
        for(int i=1;i<arr.length;i++){
            if(arr[i] != arr[i-1]){
                lesscount = i;
            }
            if(arr[i] == lesscount){
                ans++;
            }
        }

        return ans;
    }
}
```

C++ Code:

```
#include <iostream>
#include <algorithm>

int GoodIntegers(int arr[], int n) {
    std::sort(arr, arr + n);
    int ans = 0;
    int lesscount = 0;

    if (arr[0] == 0) {
        ans++;
    }

    for (int i = 1; i < n; i++) {
        if (arr[i] != arr[i - 1]) {
            lesscount = i;
        }

        if (arr[i] == lesscount) {
            ans++;
        }
    }

    return ans;
}

int main() {
    int n;
    std::cin >> n;

    int arr[n];
    for (int i = 0; i < n; i++) {
        std::cin >> arr[i];
    }

    std::cout << GoodIntegers(arr, n) << std::endl;

    return 0;
}
```

Python Code:

```
def GoodIntegers(arr):
    arr.sort()
    n = len(arr)
    ans = 0
    lesscount = 0

    if arr[0] == 0:
        ans += 1

    for i in range(1, n):
        if arr[i] != arr[i - 1]:
            lesscount = i

        if arr[i] == lesscount:
            ans += 1

    return ans

if __name__ == "__main__":
    n = int(input())
    arr = list(map(int, input().split()))

    print(GoodIntegers(arr))
```

Bubble Sort

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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();
}
BubbleSort(arr);

for(int i=0;i<n;i++){
    System.out.print(arr[i]+" ");
}
}

public static void BubbleSort(int[] arr){
    int n = arr.length;

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

```

C++ Code:

```

#include <iostream>

void BubbleSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - 1 - i; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

int main() {
    int n;
    std::cin >> n;

```

```

int arr[n];
for (int i = 0; i < n; i++) {
    std::cin >> arr[i];
}

BubbleSort(arr, n);

for (int i = 0; i < n; i++) {
    std::cout << arr[i] << " ";
}

return 0;
}

```

Python Code:

```

def BubbleSort(arr):
    n = len(arr)

    for i in range(n - 1):
        for j in range(n - 1 - i):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

if __name__ == "__main__":
    n = int(input())
    arr = list(map(int, input().split()))

    BubbleSort(arr)

    for num in arr:
        print(num, end=" ")

```



Today's agenda

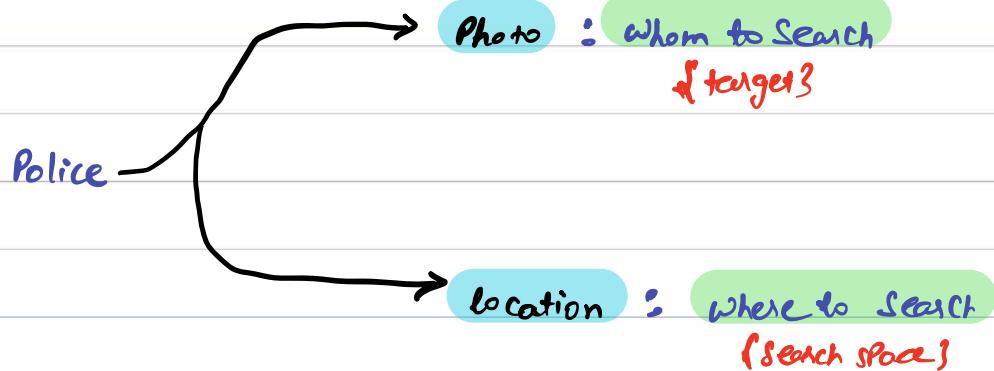
- ↳ Searching basics
- ↳ why mid at half
- ↳ search in sorted array
- ↳ floor in a sorted array
- ↳ Every element occurs twice except for 1.



AlgoPrep



Story:



Ex:

(whom to search)

word (Algorithm)

Contact number (HR)

(where to search)

{Newspaper | Book | Dict}

{Telephone directory | LinkedIn}

↳ your search space is ordered, searching becomes easier.

Note: Binary Search can be possibly applied even if array is not sorted.

Divide your search space in 2 halves, if we can discard 1 half, we can apply BS.



Q) Given a sorted array search if K is present or not?

array: { 4 7 10 13 15 20 21 24 26 28 } K=13
↓
true

1/idea 1

↳ linear search.

T.C: $O(n)$

S.C: $O(1)$

1/idea 2

→ binary search ↳ search space: array

↳ target: K

Case 1:



if (array[mid] == k) { return true; }

Case 2:



if (array[mid] < k) {

discard left side / search on right.

}



Case 3 :



$\text{if } (\text{arr}[m] > K) \{$

discard right side / Search on left.

}

$\text{arr}[10]: \{ 4 \ 7 \ 10 \ 13 \ 15 \ 20 \ 21 \ 24 \ 26 \ 28 \} \quad K=13$

lo

hi

$$m = \frac{lo+hi}{2}$$

0

4

$\text{if } (\text{arr}[m] > K):$ Search on left; $hi=m-1$

0

1

$\text{if } (\text{arr}[m] < K):$ Search on right; $lo=m+1$

2

2

$\text{if } (\text{arr}[m] < K):$ Search on right; $lo=m+1$

3

3

$\text{if } (\text{arr}[m] == K):$ return true;

$\text{arr}[10]: \{ 4 \ 7 \ 10 \ 14 \ 15 \ 20 \ 21 \ 24 \ 26 \ 28 \} \quad K=13$

lo

hi

$$m = \frac{lo+hi}{2}$$

0

4

$\text{if } (\text{arr}[m] > K):$ Search on left; $hi=m-1$

0

1

$\text{if } (\text{arr}[m] < K):$ Search on right; $lo=m+1$

2

2

$\text{if } (\text{arr}[m] < K):$ Search on right; $lo=m+1$

3

3

$\text{if } (\text{arr}[m] > K):$ Search on left; $hi=m-1$

3

2

→ exit



// Pseudo Code

```
boolean Search (int arr[], int k){
```

```
    int lo = 0;
```

```
    int hi = N-1;
```

```
    while (lo <= hi) {
```

```
        int m = (lo + hi) / 2;
```

```
        if (arr[m] == k) {
```

```
            return true;
```

```
}
```

```
        else if (arr[m] < k) {
```

```
            lo = m+1;
```

```
}
```

```
        else {
```

```
            hi = m-1;
```

```
}
```

```
    return false;
```

T.C: $O(\log N)$

S.C: $O(1)$



3

3





Q) Given a sorted arr[N], find floor of given num k.

↳ just smaller (greatest no. $\leq k$ in arr[]) or equal

Ex: arr[9] = { -4 3 4 7 10 11 12 15 19 }

K=5 : 4

K=7 : 7

K=11 : 11

1/idea1

↳ linear search

T.C: $O(n)$

S.C: $O(1)$

1/idea2

↳ binary search

Case 1:



if (arr[mid] == k) { return k; }

Case 2:



if (arr[mid] < k) {

ans = arr[mid];

discard left / go to right

}



Case 3:



if ($\text{arr}[mid] > K$) {

(discard right) go to left

}

// Pseudo Code

```
int floor (int arr[], int k) {  
    int lo = 0;  
    int hi = N-1;  
    int ans = -∞;
```

T.C: $O(\log n)$

S.C: $O(1)$

while ($lo \leq hi$) {

int m = $(lo + hi) / 2$;

if ($\text{arr}[m] == k$) {

return k;

}

else if ($\text{arr}[m] < k$) {

ans = arr[m]; lo = m+1;

} else {

hi = m-1;

return ans;



Ex: $\text{arr}[g] = \{-4, 3, 4, 7, 10, 11, 12, 15, 19\}$ $k=6$

$\text{ans} = -\infty$

lo hi m

0 8 4 : if $(\text{arr}[m] > k) \rightarrow$ go to left : $hi = m-1$

0 3 1 : if $(\text{arr}[m] < k) \rightarrow \text{ans} = 3 \rightarrow$ go to right
 $lo = m+1$

2 3 2 : if $(\text{arr}[m] < k) \rightarrow \text{ans} = 4 \rightarrow$ go to right
 $lo = m+1$

3 3 3 : if $(\text{arr}[m] > k) \rightarrow$ go to left
 $hi = m-1$



Ans: 4

→ Break till 9:44 pm



Q) Every element occurs twice except for 1, find unique element.

Note: duplicates are adjacent to each other

Ex: `arr[15]: 4 4 1 1 9 9 11 11 20 7 7 3 3 5 5`

1/idea1

b) Take xor of all elements.

T.C: $O(n)$

S.C: $O(1)$

1/idea2

b) Binary Search

`arr[15]:`

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
4	4	1	1	9	9	11	11	20	7	7	3	3	5	5

Pre Single occ: Numbers are starting from even index

Post Single occ: Numbers are starting from odd index



Case 1:



| if (arr[mid] != arr[mid+1] && arr[mid] != arr[mid+1]) {
| | return arr[mid];
| }
|

Case 2: my mid is at first occurrence

| if (mid % 2 == 0) {
| | discard left | go to right
| }
|



Case 3:

my mid is at first occurrence

| if (mid % 2 == 1) {
| | reject right | go to left
| }
|

Tracing



lo hi mid *
0 14 7

$m = 6$ $mx.2 == 0$, go to right, $lo = mid + 2$

8 14 11 $m = 11$ $mx.2 == 1$, go to left, $hi = mid - 1$

8 10 9 $m = 9$ $mx.2 == 1$, go to left, $hi = mid - 1$

8 8 8 → return arr[mid];

* How to make sure that mid lands at 1st occ.

if (arr[mid] == arr[mid - 1]) <

mid--;

3

else {

// No Change

X 2



// Pseudo Code

```
int unique (int arr[n]) {
    // 0th index if (arr[0] != arr[1]) { return arr[0]; }

    // last index if (arr[n-1] != arr[n-2]) { return arr[n-1]; }

    int lo = 2;
    int hi = n-3;

    while (lo <= hi) {
        int mid = (lo+hi)/2;

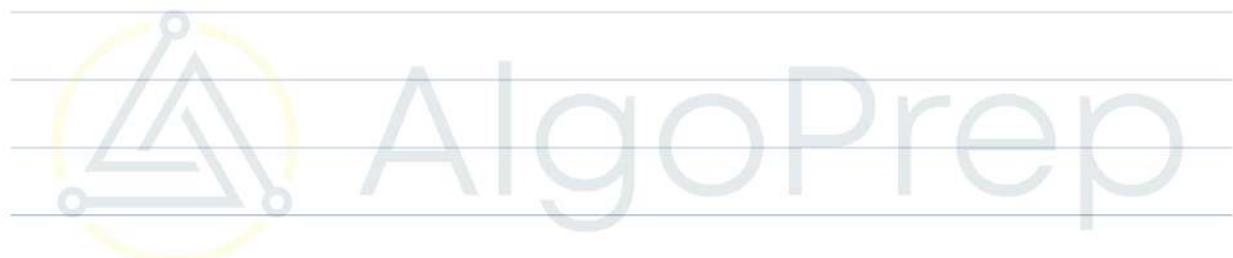
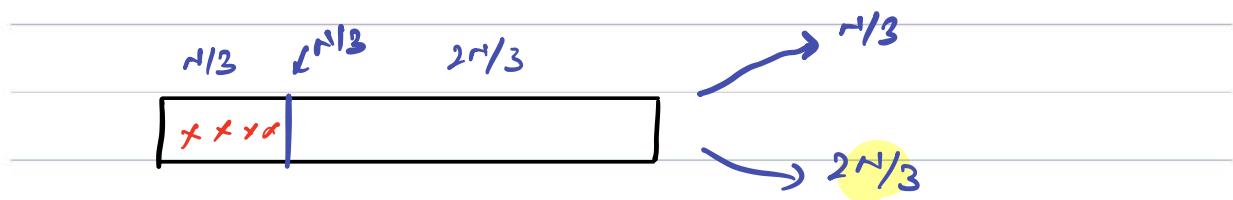
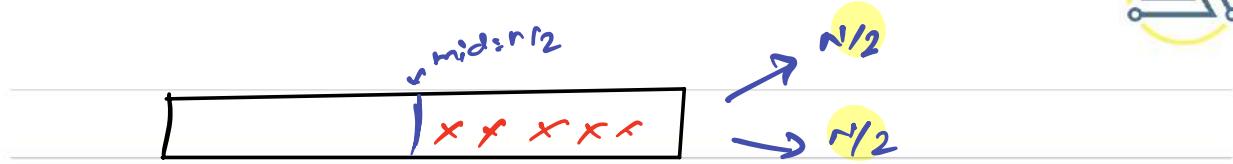
        if (arr[mid] != arr[mid-1] && arr[mid] != arr[mid+1]) {
            return arr[mid];
        }
    }

    if (arr[mid] == arr[mid-1]) {
        mid--;
    }

    if (mid%2 == 0) {
        lo = mid+2;
    } else {
        hi = mid-1;
    }

    return -1;
}
```

T.C: $O(\log N)$
S.C: $O(1)$



Search in a Sorted Array

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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 k = scn.nextInt();
        System.out.println(isPresent(arr,k));
    }

    public static boolean isPresent(int[] arr, int k){
        int lo = 0;
        int hi = arr.length - 1;

        while(lo <= hi){
            int m = (lo+hi)/2;

            if(arr[m] == k){
                return true;
            }else if(arr[m] < k){
                lo = m + 1;
            }else{
                hi = m-1;
            }
        }

        return false;
    }
}
```

C++ Code:

```
#include <iostream>
using namespace std;

bool isPresent(int arr[], int n, int k) {
    int lo = 0;
    int hi = n - 1;

    while (lo <= hi) {
        int m = (lo + hi) / 2;

        if (arr[m] == k) {
            return true;
        } else if (arr[m] < k) {
            lo = m + 1;
        } else {
            hi = m - 1;
        }
    }

    return false;
}

int main() {
    int n;
    cin >> n;

    int arr[n];
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int k;
    cin >> k;

    cout << (isPresent(arr, n, k) ? "Present" : "Not Present") << endl;

    return 0;
}
```

Python Code:

```
def is_present(arr, k):
    lo = 0
    hi = len(arr) - 1

    while lo <= hi:
        m = (lo + hi) // 2

        if arr[m] == k:
            return True
        elif arr[m] < k:
            lo = m + 1
        else:
            hi = m - 1

    return False

n = int(input())
arr = list(map(int, input().split()))

k = int(input())

print("Present" if is_present(arr, k) else "Not Present")
```

Floor in a Sorted Array

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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 k = scn.nextInt();
        System.out.println(Floor(arr,k));
    }

    public static int Floor(int[] arr, int k){
        int lo = 0;
        int hi = arr.length - 1;
        int ans = Integer.MIN_VALUE;

        while(lo <= hi){
            int m = (lo+hi)/2;

            if(arr[m] == k){
                return k;
            }else if(arr[m] < k){
                ans = arr[m];
                lo = m + 1;
            }else{
                hi = m-1;
            }
        }

        return ans;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
using namespace std;

int Floor(vector<int>& arr, int k) {
    int lo = 0;
    int hi = arr.size() - 1;
    int ans = INT_MIN;

    while (lo <= hi) {
        int m = (lo + hi) / 2;

        if (arr[m] == k) {
            return k;
        } else if (arr[m] < k) {
            ans = arr[m];
            lo = m + 1;
        } else {
            hi = m - 1;
        }
    }

    return ans;
}

int main() {
    int n;
    cin >> n;

    vector<int> arr(n);
    for (int i = 0; i < n; i++) {
        cin >> arr[i];
    }

    int k;
    cin >> k;

    cout << Floor(arr, k) << endl;

    return 0;
}
```

Python Code:

```
def Floor(arr, k):
    lo = 0
    hi = len(arr) - 1
    ans = float('-inf')

    while lo <= hi:
        m = (lo + hi) // 2

        if arr[m] == k:
            return k
        elif arr[m] < k:
            ans = arr[m]
            lo = m + 1
        else:
            hi = m - 1

    return ans

n = int(input())
arr = list(map(int, input().split()))
k = int(input())

print(Floor(arr, k))
```

Single Element in Array

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        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();
        }

        System.out.println(singleNonDuplicate(arr));
    }

    public static int singleNonDuplicate(int[] nums) {
        if(nums.length == 1){
            return nums[0];
        }
        if(nums[0] != nums[1]){
            return nums[0];
        }
        if(nums[nums.length-1] != nums[nums.length - 2]){
            return nums[nums.length-1];
        }
        int left = 2;
        int right = nums.length - 3;

        while(left < right){
            int middle = (left+right)/2;

            if(nums[middle-1]!= nums[middle] && nums[middle]!= nums[middle+1]){
                if(nums[middle-1]== nums[middle+1]){
                    left = middle + 1;
                } else {
                    right = middle;
                }
            } else {
                left = right;
            }
        }
        return nums[left];
    }
}
```

```

        return nums[middle];
    }

    if(nums[middle] == nums[middle - 1]){
        middle--;
    }
    if(middle % 2 == 0){
        left = middle + 2;
    }else{
        right = middle - 1;
    }
}

return nums[left];
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
using namespace std;

int singleNonDuplicate(vector<int>& nums) {
    if(nums.size() == 1){
        return nums[0];
    }
    if(nums[0] != nums[1]){
        return nums[0];
    }
    if(nums[nums.size()-1] != nums[nums.size() - 2]){
        return nums[nums.size()-1];
    }
    int left = 2;
    int right = nums.size() - 3;

    while(left < right){
        int middle = (left + right) / 2;

        if(nums[middle-1] != nums[middle] && nums[middle] != nums[middle+1]){
            return nums[middle];
        }

        if(nums[middle] == nums[middle - 1]){

```

```

        middle--;
    }
    if(middle % 2 == 0){
        left = middle + 2;
    } else {
        right = middle - 1;
    }
}

return nums[left];
}

int main() {
    int n;
    cin >> n;
    vector<int> nums(n);
    for(int i = 0; i < n; i++){
        cin >> nums[i];
    }
    cout << singleNonDuplicate(nums) << endl;
    return 0;
}

```

Python Code:

```

def singleNonDuplicate(nums):
    if len(nums) == 1:
        return nums[0]
    if nums[0] != nums[1]:
        return nums[0]
    if nums[-1] != nums[-2]:
        return nums[-1]
    left = 2
    right = len(nums) - 3

    while left < right:
        middle = (left + right) // 2

        if nums[middle - 1] != nums[middle] and nums[middle] != nums[middle + 1]:
            return nums[middle]

        if nums[middle] == nums[middle - 1]:
            middle -= 1
        if middle % 2 == 0:
            left = middle + 2

```

```

else:
    right = middle - 1

return nums[left]

n = int(input())
nums = list(map(int, input().split()))
print(singleNonDuplicate(nums))

```

Peak Element

Solution Vid:

Youtube Link: <https://youtu.be/0HqoT8GGbeU>

Algoprep Link: <https://learn.algoprep.in/s/courses/648afad5e4b08bd5923fce6e/take>

Java Code:

```

class Solution {
    public int findPeakElement(int[] nums) {
        if(nums.length == 1) {
            return 0;
        }

        if(nums[0] > nums[1]) {
            return 0;
        }

        if(nums[nums.length - 1] > nums[nums.length - 2]) {
            return nums.length - 1;
        }

        int left = 1;
        int right = nums.length - 2;

        while (left <= right) {
            int middle = (left + right) / 2;
            if(nums[middle] > nums[middle - 1] && nums[middle] > nums[middle+1]) {
                return middle;
            }

            if(nums[middle] < nums[middle - 1]) {

```

```

        right = middle - 1;
    }else {
        left = middle + 1;
    }
}
}

return -1;
}
}
}

```

C++ Code:

```

class Solution {
public:
    int findPeakElement(vector<int>& nums) {
        if(nums.size() == 1){
            return 0;
        }

        if(nums[0] > nums[1]){
            return 0;
        }
        if(nums[nums.size() - 1] > nums[nums.size() - 2]){
            return nums.size() - 1;
        }

        int left = 1;
        int right = nums.size() - 2;

        while (left <= right) {
            int middle = (left + right) / 2;
            if(nums[middle] > nums[middle - 1] && nums[middle] > nums[middle + 1]){
                return middle;
            }

            if(nums[middle] < nums[middle - 1]) {
                right = middle - 1;
            } else {
                left = middle + 1;
            }
        }
        return -1;
    }
};

```

Python Code:

```
class Solution:
    def findPeakElement(self, nums: List[int]) -> int:
        if len(nums) == 1:
            return 0

        if nums[0] > nums[1]:
            return 0
        if nums[-1] > nums[-2]:
            return len(nums) - 1

        left = 1
        right = len(nums) - 2

        while left <= right:
            middle = (left + right) // 2
            if nums[middle] > nums[middle - 1] and nums[middle] > nums[middle + 1]:
                return middle

            if nums[middle] < nums[middle - 1]:
                right = middle - 1
            else:
                left = middle + 1
        return -1
```



Today's agenda

- ↳ Classes and objects
- ↳ Constructors
- ↳ Nested class
- ↳ LinkedList Intro
- ↳ Point linkedlist



AlgoPrep



// mid Calculation

lo

'int'

hi

'int'

$$\text{int } m = (\text{lo} + \text{hi}) / 2$$

XX

-2^{31}

lo

hi

$$\text{int } m = \text{lo} + \frac{(\text{hi} - \text{lo})}{2}$$

✓

↳ prevent overflow

$2^{31} - 1$

lo

'int'

hi

$2^{31} - 1$

$$m = \frac{\underline{1} + 2^{31} - 1}{2} \rightarrow 2^{31}$$

$$\text{int } m = \text{lo} + \frac{(\text{hi} - \text{lo})}{2}$$

lo

'int'

hi

$2^{31} - 1$

$$m = \underline{1} + \frac{(2^{31} - 1 - 1)}{2}$$



11 Classes and objects

int, Char, String, double

b) int n = 20;

→ To combine different data types and data structures.

*

Public static class Pair {
 int n;
 int y;

3

Pair p1 = new Pair();

Pair p2 = new Pair();

n = 0
y = 50

System.out.println(p2.y); → 50

Pair p3 = new Pair();

p2.y = 50;

System.out.println(p2.y); → 50

n = 0
y = 0



* class: It is a blueprint.

* object: real instance of blue print

Class home {

 int roomCount

 int floorCount

 int area

 String Color

 boolean Pool

3

Sharry

home Sharry:

 new home()

 Sharry.roomCount=4

 Sharry.floorCount=2

 Sharry.Color="red"

 Pool=true;

roomCount=4
floorCount=2
area=0
Color="Red"
Pool=false

Sharry

Chandra Sai

home Sai =

 new home();

Adarsh

home Adarsh

=new home();

Sai



11 Constructors → To make your life easy.

```
Public static class Pair {  
    int x;  
    int y;
```

3

```
Pair p1 = new Pair();  
p1.x = 10;  
p1.y = 20;
```

10
20

$$x = 10$$
$$y = 20$$

p1

```
Public static class Pair {  
    int x;  
    int y;
```

```
    Pair (int v1, int v2) {  
        x = v1;  
        y = v2;
```

```
    Pair () {
```

3

```
    Pair (int v1) {
```

3

```
Pair p1 = new Pair(10, 20);
```

$$x = 10$$
$$y = 20$$

p1

```
Pair p2 = new Pair();
```

$$x = 0$$
$$y = 0$$

`int n = "Hello"`] \rightarrow while assigning, type should be same.
Pair P1 = 10;



Break till 9:28 PM

// Nested class

```
Class Node {  
    int val;  
    Node next;
```

```
Node (int v){  
    val = v;  
}
```

}

Node n1 = new Node(10);

int val = 10;
Node next = null;
n2

n1

Node n2 = new Node(20);

int val = 20
node.next = null

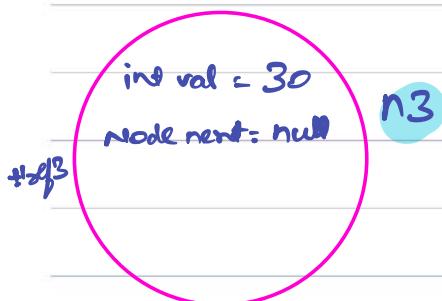
n2

`n1.next = n2;`

`Print (n1.next.val);` \rightarrow 20



Node n₃ = new Node(30); Node n₁ = new Node(10);



n₁.next = n₂;

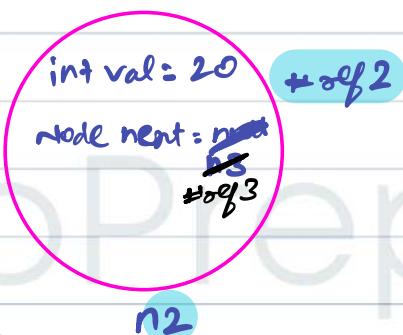
n₂.next = n₃;

Node n₂ = new Node(20);

Point(n₂.val); → 20

Point(n₂.next.next); → null

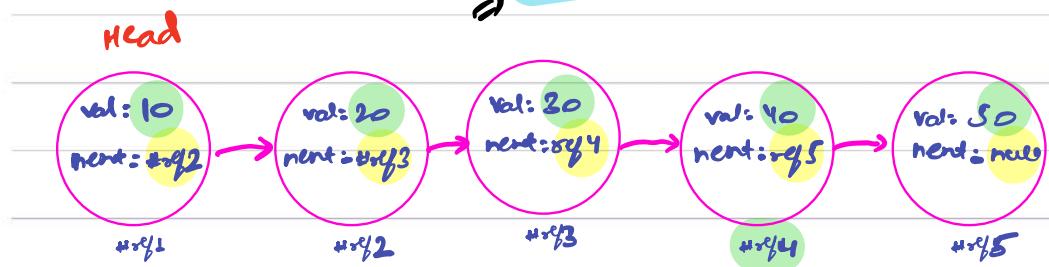
Point(n₁.next.next.val); → 30



Point(n₁.next); → reference of n₂



→ Linked list



Point (Head.val); → 10

Point (Head.next.val); → 20

Point (Head.next.next.val); → 20

Point (Head.next.next.next.val); → 40

Point (Head.next.next.next.next.val); → 50

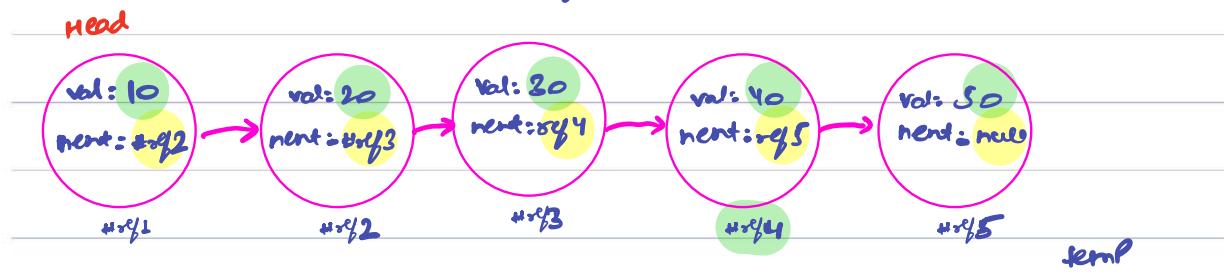


AlgoPrep



Q) Point linkedlist

Given head of the LL Point the elements.



Void PointLinkedList (Node head){

Node temp = head;

while (temp!=null) {

Point (temp.val);

temp = temp.next;

10 20 30
40 50

T.C: O(n)

S.C: O(1)

}



Today's agenda

↳ insert in a linkedlist

↳ delete in a linkedlist

↳ Reverse the linkedlist

↳ Find mid

↳ floyd cycle.



AlgoPrep

```
class Node {
```

```
    int val;
```

```
    Node next;
```

```
} Node (int data) {
```

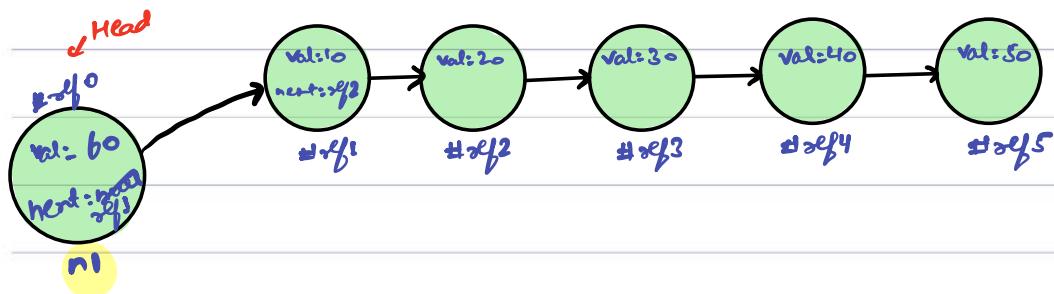
```
    val = data;
```

3



Q) insert in a linkedlist

insert at head
l1 v=60



void **insert at Start** (Node head, int v){

T.C: **O(1)**

S.C: **O(1)**

Node *n1* = new Node (v);

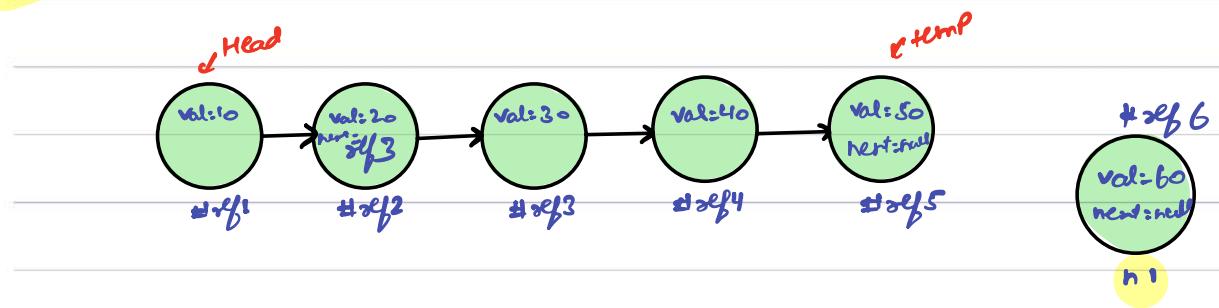
n1.next = *head*;

Head = *n1*;

3

insert after last node
6 y=60

temp.next = null
out jad



void insertatlast (Node head, int v){

Node n1 = new Node (v);

Node temp = head;

while (temp.next != null) {

temp = temp.next;

temp.next = n1;

3

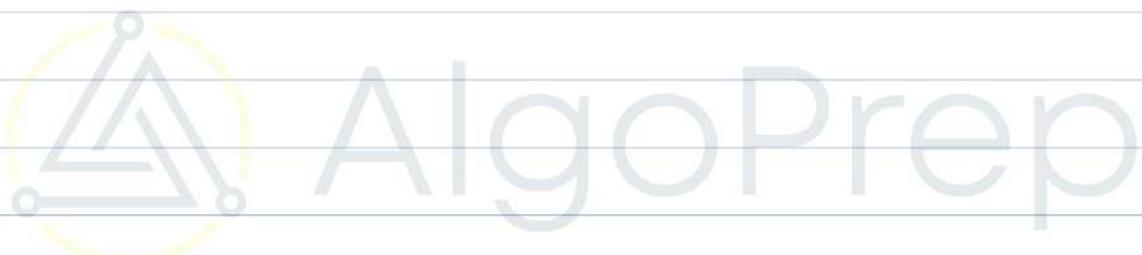
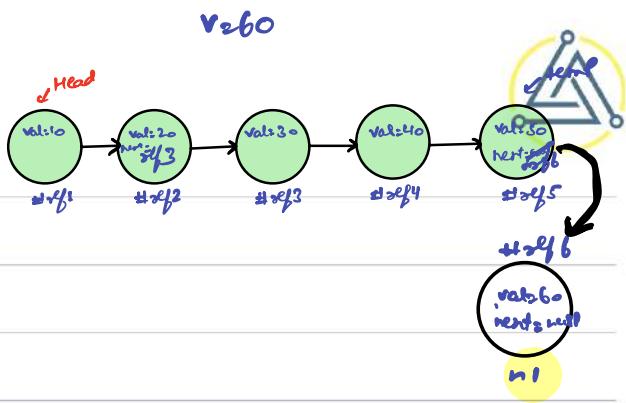


ArgoPrep

```

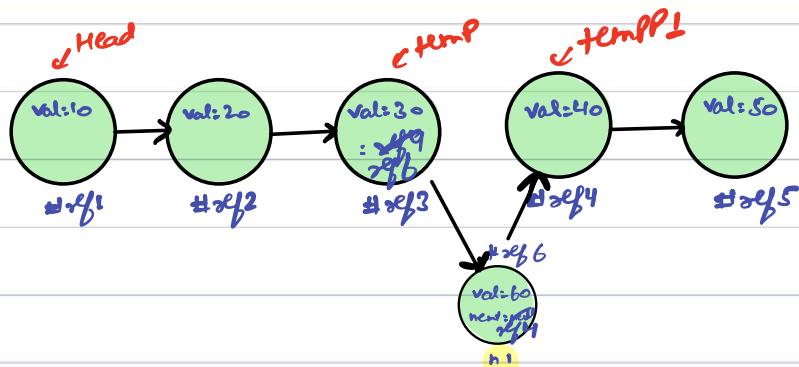
void insertAtlast (Node head, int v) {
    Node n1 = new Node(v);
    Node temp = head;
    while (temp.next != null) {
        temp = temp.next;
    }
    temp.next = n1;
}

```

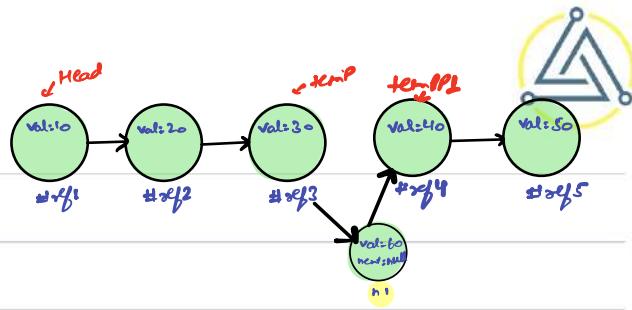




||insert at index → k=3 , v=60



```
void insertAt(Node head, int v, int k){  
    Node n1 = new Node(v);  
    Node temp = head;  
    // K-1 jumps;  
    for(int i=1; i<=k-1; i++) {  
        temp = temp.next;  
    }  
    Node tempL = temp.next;  
    temp.next = n1;  
    n1.next = tempL;
```



```

void insertat (Node head, int v, int k) {
    Node n1 = new Node (v);
    Node temp = head;
    // K-1 jumps;
    for (int i=1; i<=k-1; i++) {
        temp = temp.next;
    }
    Node tempP1 = temp.next;
    temp.next = n1;
    n1.next = tempP1;
}

```

Edge cases

↳ head := null

↳ k == 0

↳ k == size of LL (add at front)

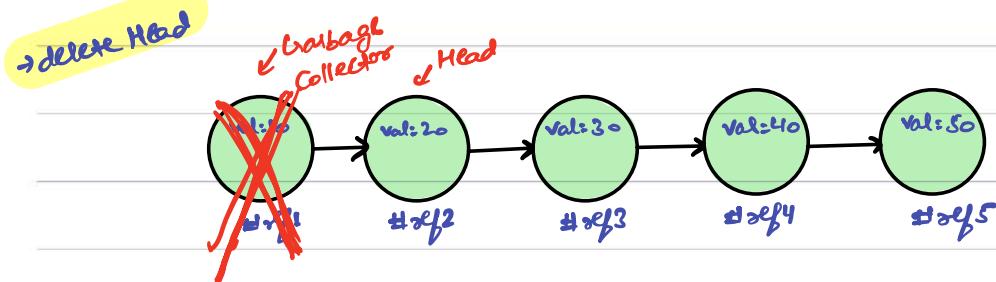
→ null pointer exception



AlgoPrep



a) Delete in a linkedlist



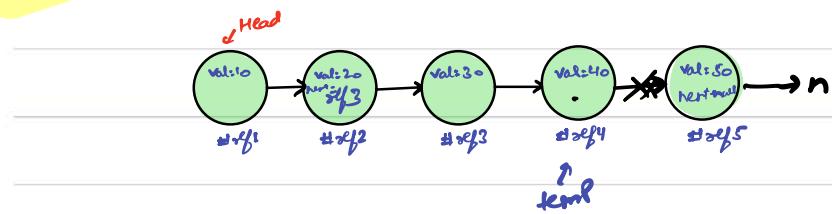
↳ `head = head.next;`



AlgoPrep



→ delete last



deleteLast (Node head){

 Node temp = head;

T.C: $O(n)$

S.C: $O(1)$

 while (temp.next != null) {

 temp = temp.next;

 3

 temp.next = null;

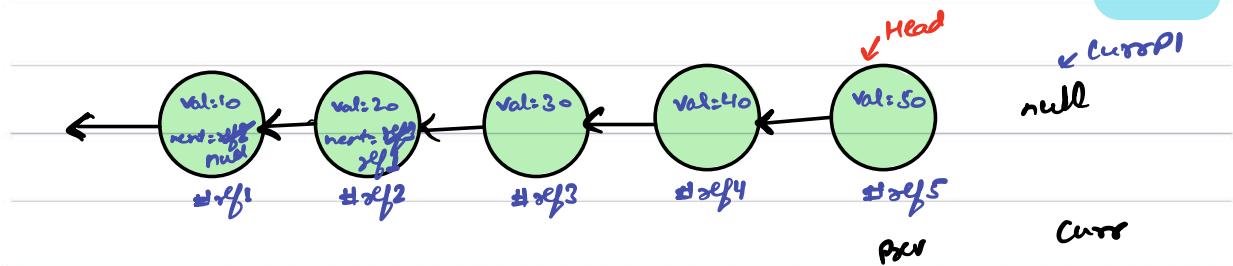
3

Break till 9:37 PM



Q) Reverse a linkedlist

→ Simple recursion
just to Point in
reverse.



Void reverse (Node head)

```

Node curr = head;
Node prev = null;
while(curr != null) {
    Node currPl = curr.next;
    curr.next = prev;
    prev = curr;
    curr = currPl;
}

```

T.C: $O(n)$

S.C: $O(1)$

head = prev;

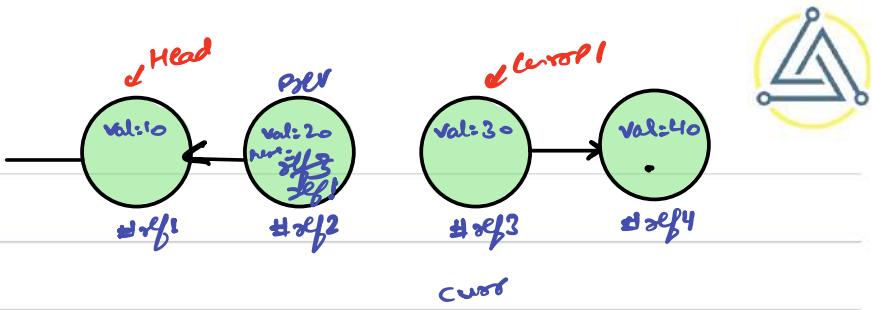
3

Edgecases

↳ head := null

↳ k := 0

↳ k := size of LL (added later)



Node Curr = head;

Node Prev = null;

while(Curr != null) {

 Node CurOp1 = Curr.next;

 Curr.next = Prev;

 Prev = Curr;

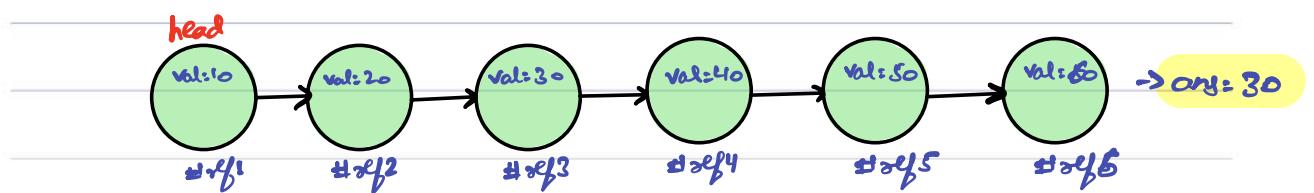
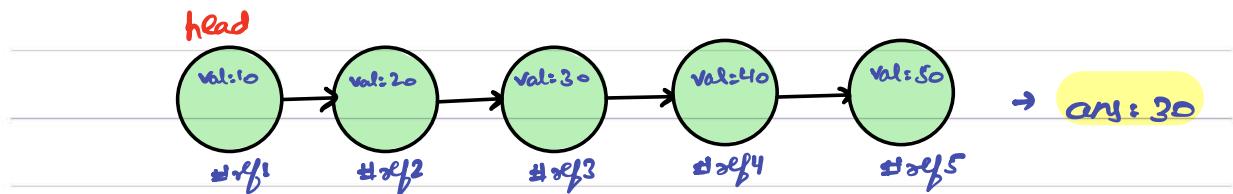
 Curr = CurOp1;

}

head = Prev;



Q) Given head of a linkedlist, find the mid of it.



//idea1

b) iterate and find out size of linkedlist And then
in 2nd iteration return $\frac{\text{size}+1}{2}$ element.

T.C: $O(N)$

S.C: $O(1)$

track = 1 Km

//idea2

do it in single iteration

Jainil

n steps

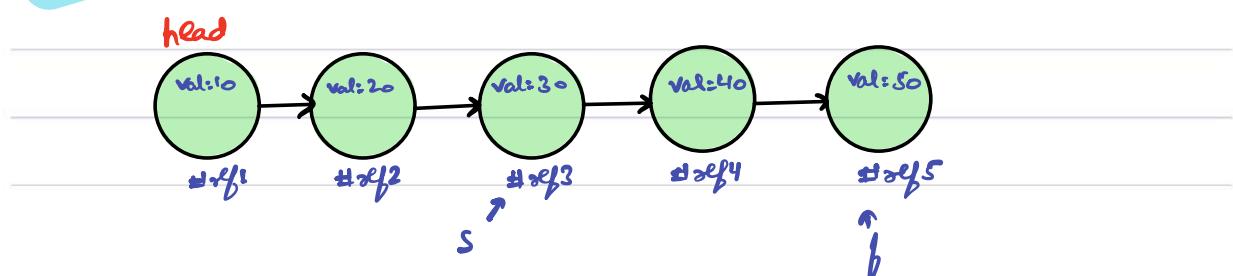
↓
500 m

Omneesh

$2n$ steps

↓
finish line

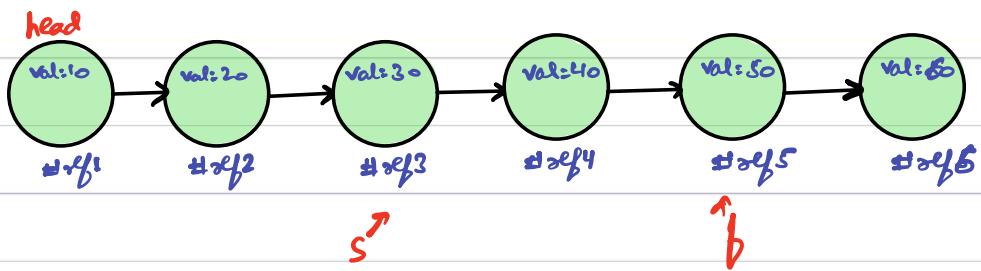
odd length



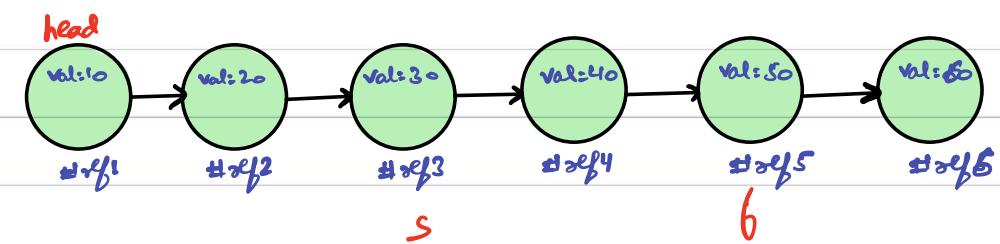
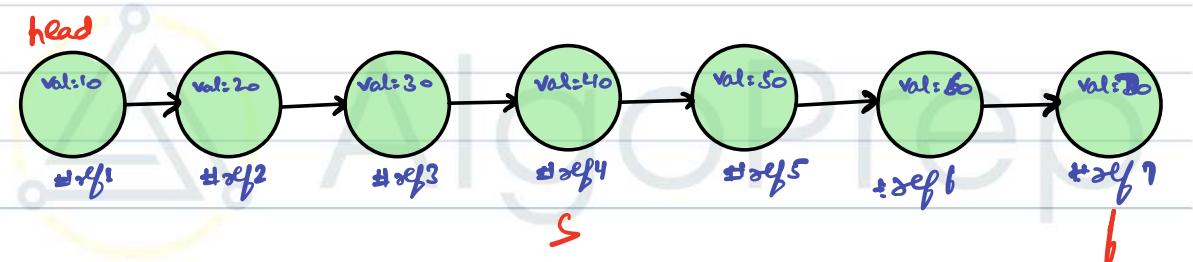
Obs: $f.\text{next} == \text{null} \rightarrow \text{exit}$, S is our answer



even length



Obs: $f.\text{next} == \text{null} \rightarrow \text{exit}$, S is our answer



Obs: $f.\text{next} == \text{null} \rightarrow \text{exit}$



II Pseudo Code

Node mid (Node head) {

 Node *s* = head;

 Node *f* = head;

 while (*f*.next != null && *f*.next.next != null) {

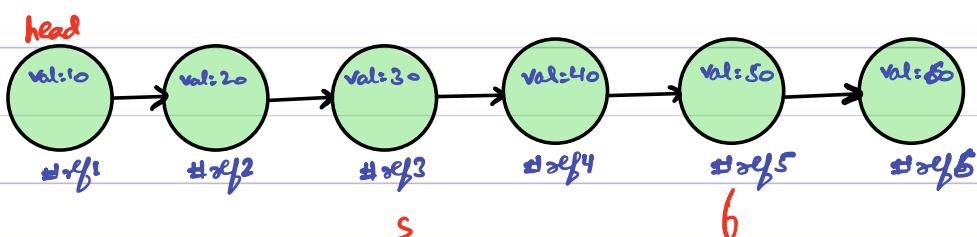
s = *s*.next;

f = *f*.next.next;

}

 return *s*;

}



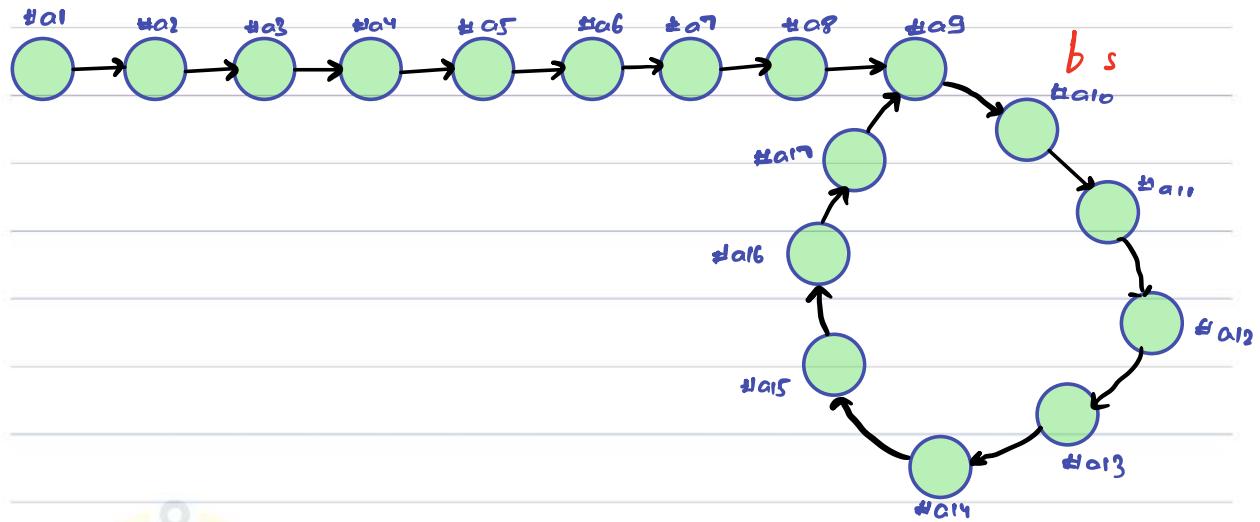
Broke till 10:45PM

Floyd Cycle



Q) Given a linked list, check for cycle & return the starting point if exists.

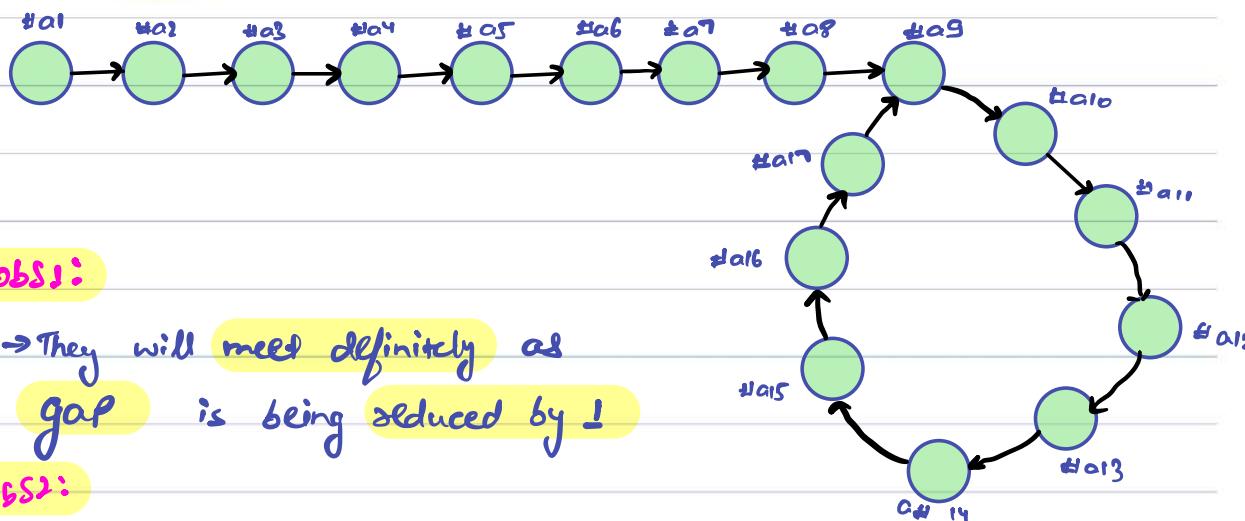
head



↳ If s meets f , then we have cycle.

head

100



Observation:

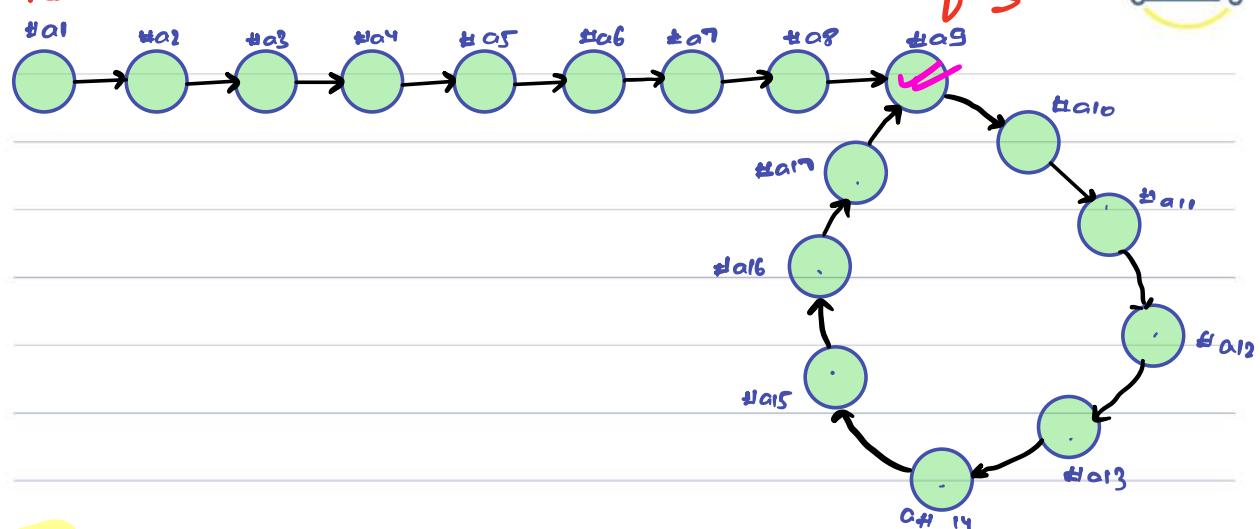
→ They will meet definitely as gap is being reduced by 1

Observation:

↳ Slow will meet fast before completing cycle.

Ent:

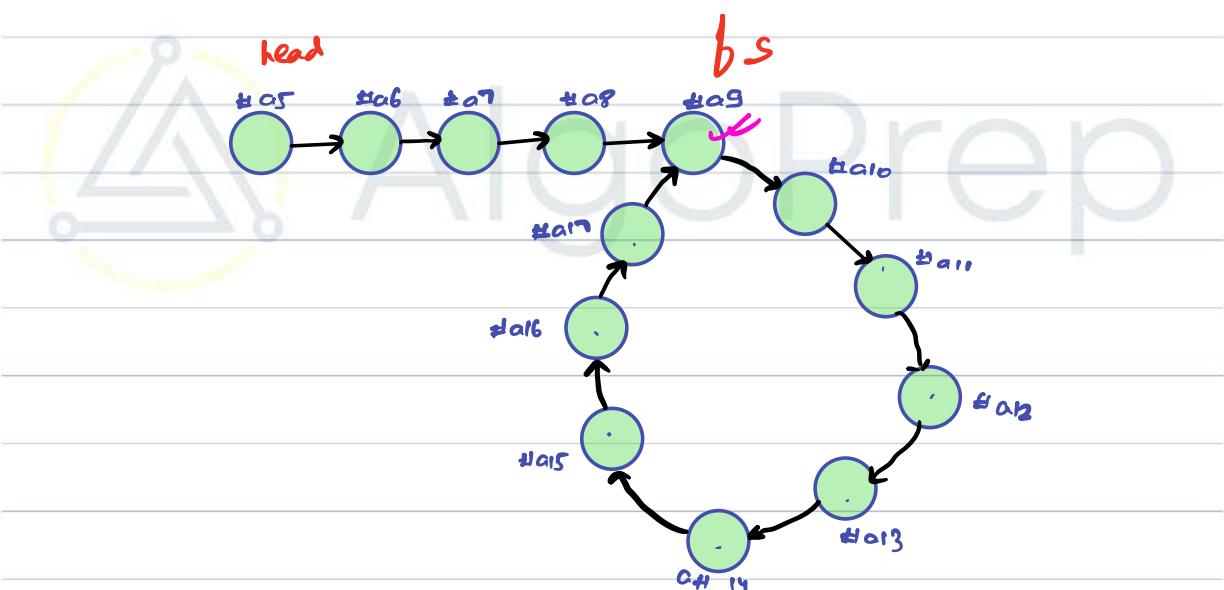
head



Bn2:

Chain = 4

Cycle = 9



Steps:

Slow & fast pointer

↳ ① after some time they will meet.

② Pick one pointer & Put it equal to head.

③ Start jumping one step at a time with both
S & f, They will meet magically at Start
Point of cycle.



//Pseudo Code

Node removecycle (Node head){

 Node *s* = head; // Check here

 Node *b* = head;

 while (*s* != *f*) {

f = *f*.next.next;

s = *s*.next;

 check,
b == null ||
b.next == null

f = head;

 while (*f* != *s*) {

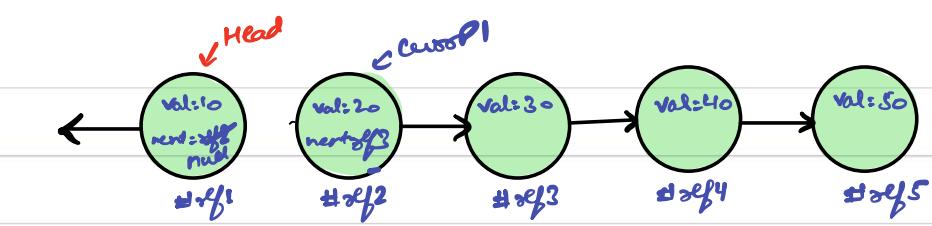
s = *s*.next;

f = *f*.next;

 return *b*;

3

W H Y ??



↑
Perv ↑
Curor

```
while ( ) {  
    Node CurorPl = Curor.next;  
    Curor.next = Perv;  
    Perv = Curor;  
    Curor = CurorPl;  
}
```



AlgoPrep

Delete Node

Java Code:

```
Node deleteNode(Node head, int x) {  
    if(head==null){  
        return null;  
    }  
    Node y=head;  
    if(x==1&&y.next==null){  
        return null;  
    }  
    if(x==1&&y.next!=null){  
        head=head.next;  
        return head;  
    }  
    Node temp=null;  
    for(int i=1;i<x-1;i++){  
        y=y.next;  
    }  
    temp=y.next.next;  
    Node del=y.next;  
    del.next=null;  
    y.next=temp;  
    return head;  
  
}
```

C++ Code:

```
struct Node {  
    int data;  
    Node* next;  
};  
  
Node* deleteNode(Node* head, int x) {  
    if (head == nullptr) {  
        return nullptr;  
    }  
    Node* y = head;  
  
    if (x == 1 && y->next == nullptr) {  
        delete y;  
        return nullptr;
```

```

    }

if (x == 1 && y->next != nullptr) {
    head = head->next;
    delete y;
    return head;
}

Node* temp = nullptr;
for (int i = 1; i < x - 1; i++) {
    y = y->next;
}
temp = y->next->next;
Node* del = y->next;
del->next = nullptr;
y->next = temp;
delete del;

return head;
}

```

Python Code:

```

class ListNode:
    def __init__(self, data=0, next=None):
        self.data = data
        self.next = next

def deleteNode(head, x):
    if head is None:
        return None
    y = head

    if x == 1 and y.next is None:
        return None

    if x == 1 and y.next is not None:
        head = head.next
        return head

    temp = None
    for i in range(1, x - 1):
        y = y.next
    temp = y.next.next
    del_node = y.next

```

```

del_node.next = None
y.next = temp

return head

```

Reverse LinkedList

Java Code:

```

class Solution {
public ListNode reverseList(ListNode head) {
if(head == null){
return null;
}

if(head.next == null){
return head;
}

ListNode prev = null;
ListNode curr = head;
while(curr != null){

ListNode currp1 = curr.next;
curr.next = prev;
prev = curr;
curr = currp1;

}
return prev;
}
}

```

C++ Code:

```

class Solution {
public:
    ListNode* reverseList(ListNode* head) {
        if (head == nullptr) {
            return nullptr;
        }

```

```

    }

    if (head->next == nullptr) {
        return head;
    }

    ListNode* prev = nullptr;
    ListNode* curr = head;

    while (curr != nullptr) {
        ListNode* next_node = curr->next;
        curr->next = prev;
        prev = curr;
        curr = next_node;
    }

    return prev;
}
};


```

Python Code:

```

class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

class Solution:
    def reverseList(self, head: ListNode) -> ListNode:
        if not head:
            return None

        if not head.next:
            return head

        prev = None
        curr = head

        while curr:
            next_node = curr.next
            curr.next = prev
            prev = curr
            curr = next_node

        return prev

```

Middle of the LinkedList

Java Code:

```
public ListNode middleNode(ListNode head) {  
    ListNode slow = head, fast = head;  
    while (fast != null && fast.next != null) {  
        slow = slow.next;  
        fast = fast.next.next;  
    }  
    return slow;  
}
```

C++ Code:

```
Unset  
ListNode* middleNode(ListNode* head) {  
    ListNode *slow = head, *fast = head;  
    while (fast && fast->next)  
        slow = slow->next, fast = fast->next->next;  
    return slow;  
}
```

Python Code:

```
Python  
def middleNode(self, head):  
    slow = fast = head  
    while fast and fast.next:  
        slow = slow.next  
        fast = fast.next.next  
    return slow
```

Floyd Cycle

Java Code:

```
public ListNode detectCycle(ListNode head) {
    ListNode slow = head, fast = head;
    while (fast != null && fast.next != null) {
        slow = slow.next;
        fast = fast.next.next;

        if (slow == fast) break;
    }

    if (fast == null || fast.next == null) return null;

    fast = head;
    while (fast != slow) {
        fast = fast.next;
        slow = slow.next;
    }
    return fast;
}
```

C++ Code:

```
struct ListNode {
    int val;
    ListNode *next;
    ListNode(int x) : val(x), next(nullptr) {}
};

class Solution {
public:
    ListNode *detectCycle(ListNode *head) {
        ListNode *slow = head, *fast = head;
        while (fast != nullptr && fast->next != nullptr) {
            slow = slow->next;
            fast = fast->next->next;

            if (slow == fast) break;
        }

        if (fast == nullptr || fast->next == nullptr) return nullptr;
```

```

fast = head;
while (fast != slow) {
    fast = fast->next;
    slow = slow->next;
}
return fast;
};


```

Python Code:

```

class ListNode:
    def __init__(self, val=0):
        self.val = val
        self.next = None

class Solution:
    def detectCycle(self, head):
        slow = head
        fast = head
        while fast is not None and fast.next is not None:
            slow = slow.next
            fast = fast.next.next

            if slow == fast:
                break

        if fast is None or fast.next is None:
            return None

        fast = head
        while fast != slow:
            fast = fast.next
            slow = slow.next

        return fast

```

Delete Node in a LinkedList_HW

Solution Vid:

<https://youtu.be/HReo4x-dJss>

Java Code:

```
public void deleteNode(ListNode node) {  
    node.val = node.next.val;  
    node.next = node.next.next;  
}
```

C++ Code:

```
JavaScript  
void deleteNode(ListNode* node) {  
    auto next = node->next;  
    *node = *next;  
    delete next;  
}
```

Python Code:

```
Unset  
def deleteNode(self, node):  
    node.val = node.next.val  
    node.next = node.next.next
```

Remove Duplicates_HW

Solution Vid:

<https://youtu.be/1y8dcee0ixA>

Java Code:

```
public ListNode deleteDuplicates(ListNode head) {  
    if(head == null){  
        return null;  
    }  
  
    ListNode current = head;  
    while (current.next != null) {  
        if (current.val == current.next.val) {  
            ListNode currentp2 = current.next.next;  
            current.next = currentp2;  
        } else {  
            current = current.next;  
        }  
    }  
    return head;  
}
```

C++ Code:

```
struct ListNode {  
    int val;  
    ListNode* next;  
    ListNode(int val) : val(val), next(nullptr) {}  
};  
  
class Solution {  
public:  
    ListNode* deleteDuplicates(ListNode* head) {  
        ListNode* current = head;  
        while (current != nullptr && current->next != nullptr) {  
            if (current->val == current->next->val) {  
                ListNode* currentp2 = current->next->next;  
                delete current->next;  
                current->next = currentp2;  
            } else {  
                current = current->next;  
            }  
        }  
        return head;  
    }
```

};

Python Code:

```
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

class Solution:
    def deleteDuplicates(self, head: ListNode) -> ListNode:
        current = head
        while current is not None and current.next is not None:
            if current.val == current.next.val:
                current.next = current.next.next
            else:
                current = current.next
        return head
```

ReOrder List_HW

Solution Vid:

<https://youtu.be/J3JDugg-GpY>

Java Code:

```
public void reorderList(ListNode head) {
    if(head==null||head.next==null) return;
    ListNode s=head;
    ListNode f=head;
    while(f.next!=null&&f.next.next!=null) {
        s=s.next;
        f=f.next.next;
    }
    ListNode Prev=null;
    ListNode Curr=s.next;
    s.next = null;
```

```

while(Curr!=null) {
    ListNode Currp1 = Curr.next;
    Curr.next = Prev;
    Prev = Curr;
    Curr = Currp1;
}

ListNode left=head;
ListNode right=Prev;
while(right != null) {
    ListNode leftp1 = left.next;
    ListNode rightp1 = right.next;

    left.next = right;
    right.next = leftp1;
    left = leftp1;
    right = rightp1;
}
}

```

C++ Code:

```

void reorderList(ListNode* head) {
    if (!head || !head->next) return;

    ListNode* s = head;
    ListNode* f = head;
    while (f->next && f->next->next) {
        s = s->next;
        f = f->next->next;
    }

    ListNode* Prev = nullptr;
    ListNode* Curr = s->next;
    s->next = nullptr;
    while (Curr) {
        ListNode* Currp1 = Curr->next;
        Curr->next = Prev;
        Prev = Curr;
        Curr = Currp1;
    }
}

```

```

ListNode* left = head;
ListNode* right = Prev;
while (right) {
    ListNode* leftp1 = left->next;
    ListNode* rightp1 = right->next;

    left->next = right;
    right->next = leftp1;

    left = leftp1;
    right = rightp1;
}
}

```

Python Code:

```

class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def reorderList(head):
    if not head or not head.next:
        return

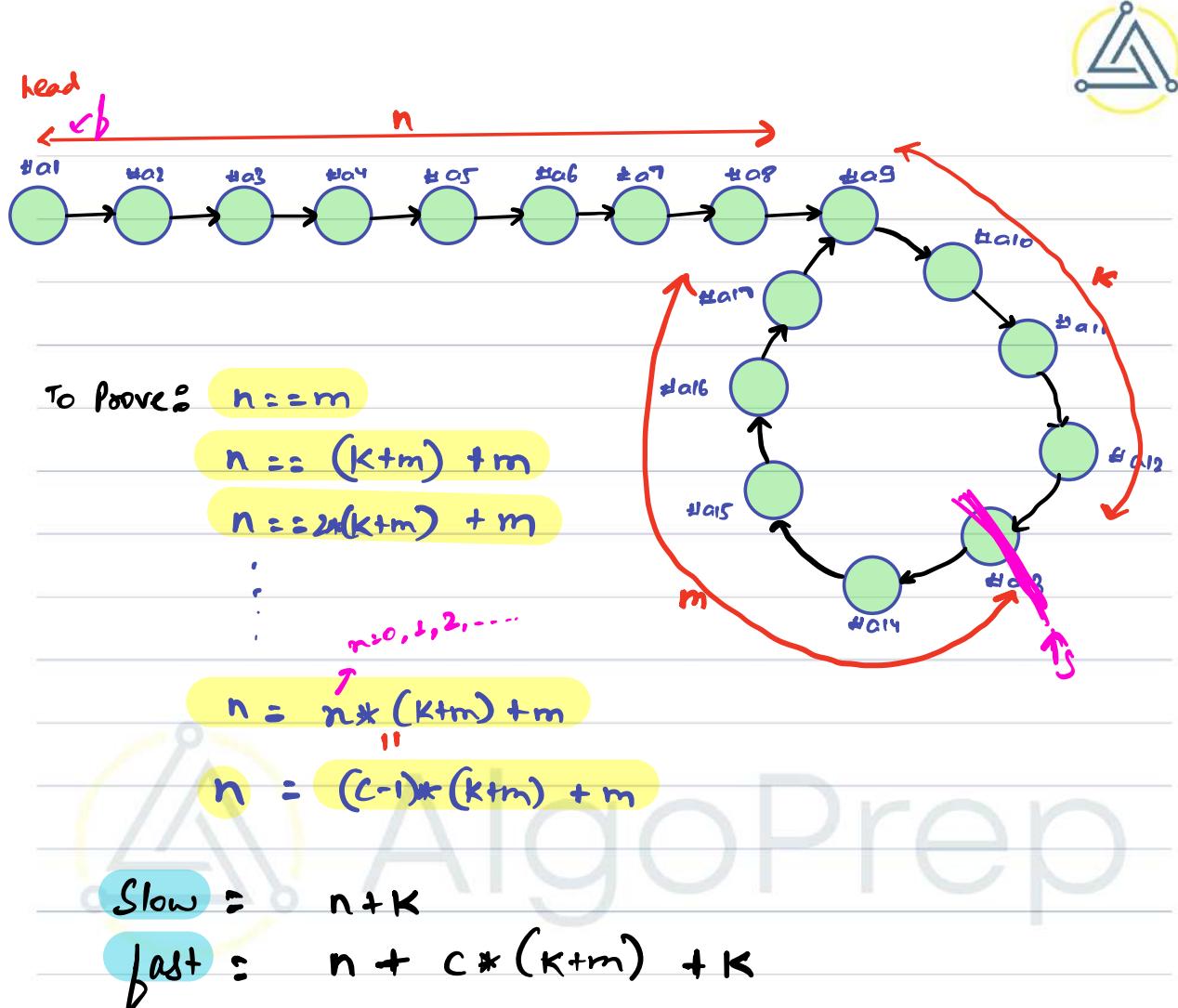
    s = head
    f = head
    while f.next and f.next.next:
        s = s.next
        f = f.next.next

    prev = None
    curr = s.next
    s.next = None
    while curr:
        currp1 = curr.next
        curr.next = prev
        prev = curr
        curr = currp1

    left = head
    right = prev

```

```
while right:  
    leftp1 = left.next  
    rightp1 = right.next  
  
    left.next = right  
    right.next = leftp1  
  
    left = leftp1  
    right = rightp1
```



$$n+k = c * (k+m)$$

$$n+k = (c-1)* (k+m) + \cancel{(k+m)}$$

$$n = (c-1)* (k+m) + m$$



Today's agenda

↳ Stacks

↳ Linkedlist as stack

↳ Remove adjacent duplicate

↳ Balanced Parentheses

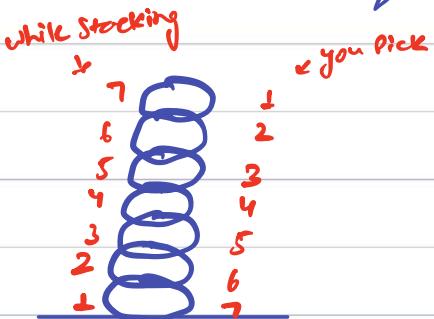
↳ Min Stack



AlgoPrep



II Stack → Last in first out.



→ Stack < Integer > St = new Stack<>();



Operations:

+·c: O(1) ← St.push(10) → add in stack

+·c: O(1) ← St.pop() → remove top from stack and return

+·c: O(1) ← St.peek() → tell you the topmost element

+·c: O(1) ← St.size() → no. of elements in stack.

LinkedList*

LRU Cache



AlgoPrep

LinkedIn

youtube insta

LinkedIn

Ex: ① Pile of Plates

② undo/redo

③ danglers in hand

Arrays [initialize]

HashMap [" "]

Stack [" "]

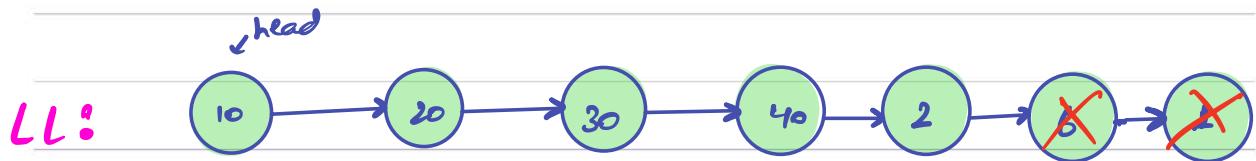
LinkedList → initialize &



→ adopted

// LinkedList as Stack

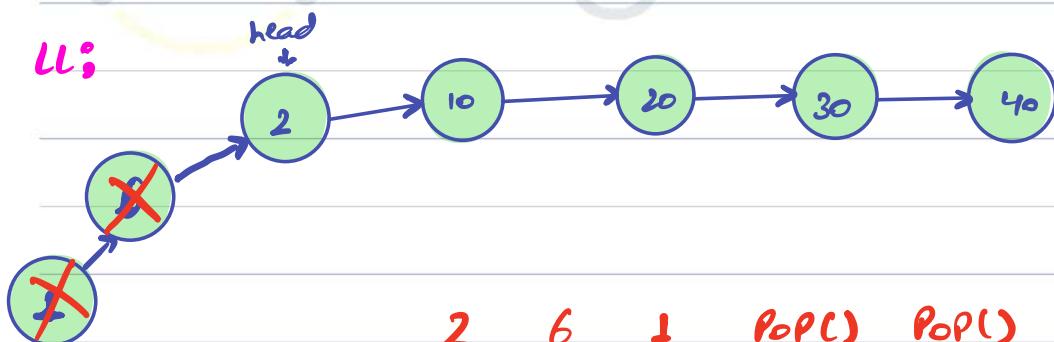
Last in first out



2 6 1 $\text{Pop}()$ $\text{Pop}()$
↓ ↓

① addlast → $O(n)$

② remove last → $O(n)$



2 6 1 $\text{Pop}()$ $\text{Pop}()$

add first → $O(1)$

remove first → $O(1)$



Q) Remove adjacent duplicate

↳ Given a string S, Remove equal Pair of adjacent characters. Return the final string.

End: ~~a b c d~~ → ad

Ex2: ~~ab~~xx~~~~xx~~~~de → ade

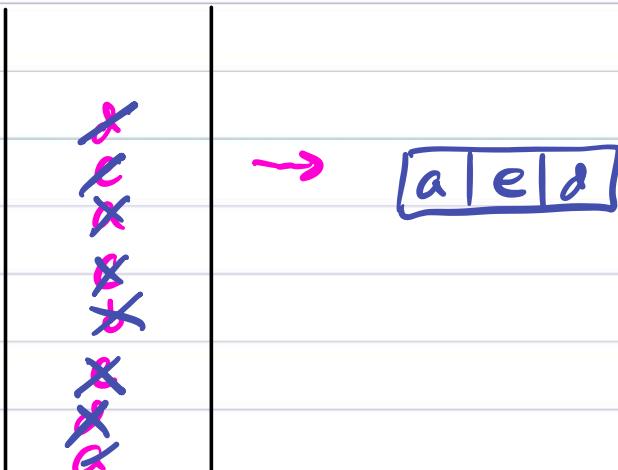
Ex3: ~~a x b e~~ → abe

Ex4: ad c & e & a & d ed
↳ adceded

Ens: a x x x d a
↳ ada

Exn!

"~~a~~~~x~~~~x~~~~b~~~~x~~~~x~~~~x~~~~x~~~~ed~~"





11pseudo code

String

Remove adjacent duplicate (String s) {

Stack <Character> st = new Stack <>();

for (int i=0; i< s.length(); i++) {

if (st.size() == 0) {

st.push(s.charAt(i));

continue;

}

if (st.peek() == s.charAt(i)) {

st.pop();

}

else {

st.push(s.charAt(i));

}

}

Char [] arr = new char [st.size()];

for (int i=arr.length-1; i>=0; i--) {

arr[i] = st.pop();

}

→ Convert arr to String & return.

3



Break till 9:45 PM



AlgoPrep



Q) Valid Parentheses

Given a String with 'c' '}' '{' '}', '[' ']', you have to find whether the String is balanced or not.

Note: balanced strings:

- ① open brackets must be closed by the same type of bracket.

s: () { } [] ()) → true

s: () { } [) → false

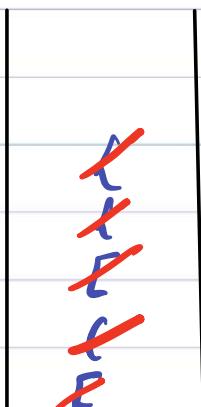
- ② opening brackets must be closed in correct order.

s: [{ }] → false

s: () { () } → false

s: [() { }] { }

?





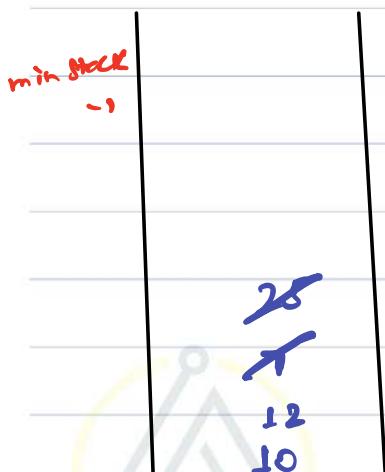
// Pseudo Code

```
boolean validParentheses (String s) {
    Stack<Character> st = new Stack<>();
    TC: O(n)
    SC: O(n)
    for (int i=0; i < s.length(); i++) {
        if (st.size() == 0) {
            st.push(s[i]);
            continue;
        }
        if (s[i] == '(' || s[i] == '[' || s[i] == '{') {
            st.push(s[i]);
        } else {
            if (s[i] == ')') {
                if (st.peek() == '(') {
                    st.pop();
                } else {
                    return false;
                }
            } else if (s[i] == ']') {
                if (st.peek() == '[') {
                    st.pop();
                } else {
                    return false;
                }
            } else if (s[i] == '}') {
                if (st.peek() == '{') {
                    st.pop();
                } else {
                    return false;
                }
            }
        }
        if (st.size() == 0) {
            return true;
        } else {
            return false;
        }
    }
}
```



Q) Min Stack

Normal Stack → $\text{Pop}()$ → top element
Min Stack → $\text{getmin}()$ → min element of stack
Complexity: $O(1)$



10 12 7 25 $\text{getmin}()$ $\text{Pop}()$ $\text{Pop}()$

↓
7

$\text{getmin}()$

↓
10

wrong

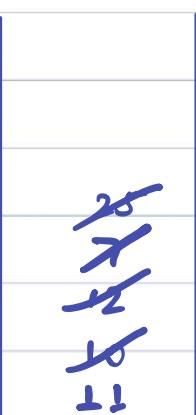
1/idea 1

11 10 12 7 25 $\text{getmin}()$ $\text{Pop}()$ $\text{Pop}()$

↓ ↓ ↓
7 25 7

$\text{Pop}()$ $\text{Pop}()$ $\text{getmin}()$

↓ ↓
12 10



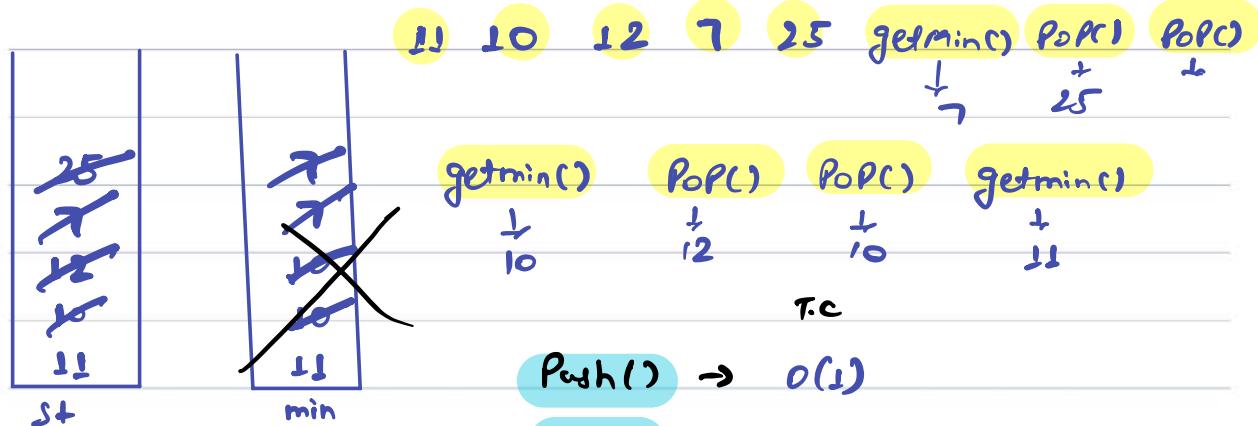
min = ~~11 12 25~~ (10)

2nd min = ~~11 25~~ (10)



Correct idea!

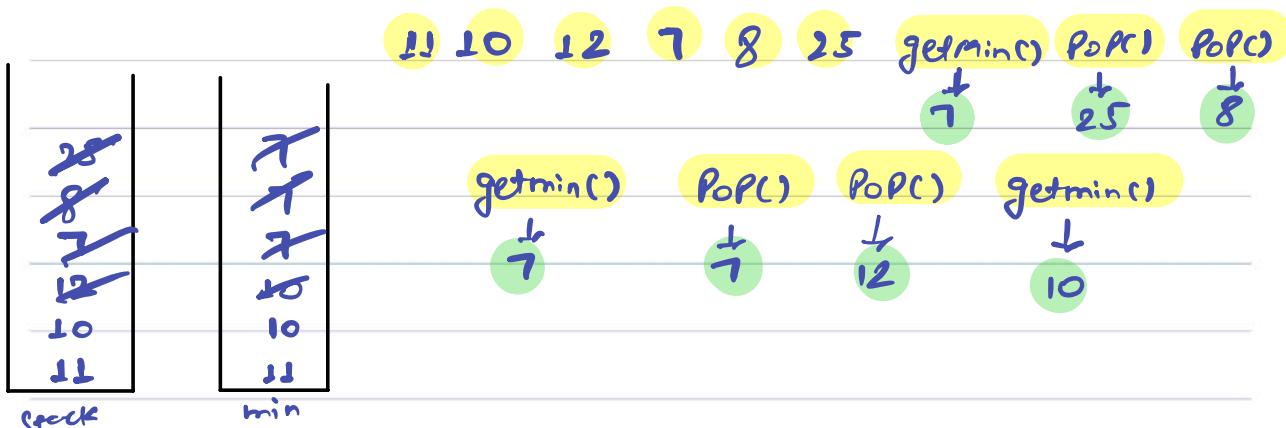
↳ we use 2 stacks



S.C. = $O(n)$

Code → HW

doubts



Remove Adjacent Duplicate

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        System.out.println(Removeadjacentduplicate(s));
    }

    public static String Removeadjacentduplicate(String s){
        Stack<Character> st = new Stack<>();

        for(int i=0;i<s.length();i++){
            if(st.size() == 0){
                st.push(s.charAt(i));
                continue;
            }

            if(st.peek() == s.charAt(i)){
                st.pop();
            }else{
                st.push(s.charAt(i));
            }
        }

        char[] arr = new char[st.size()];

        for(int i=arr.length-1;i>=0;i--){
            arr[i] = st.pop();
        }

        return String.valueOf(arr);
    }
}
```

C++ Code:

```
#include <iostream>
#include <string>
#include <stack>

std::string RemoveAdjacentDuplicate(const std::string& s) {
    std::stack<char> st;

    for (char c : s) {
        if (st.empty() || st.top() != c) {
            st.push(c);
        } else {
            st.pop();
        }
    }

    std::string result;
    while (!st.empty()) {
        result = st.top() + result;
        st.pop();
    }

    return result;
}

int main() {
    std::string s;
    std::cin >> s;

    std::cout << RemoveAdjacentDuplicate(s) << std::endl;

    return 0;
}
```

Python Code:

```
def remove_adjacent_duplicate(s):
    stack = []

    for char in s:
        if not stack or stack[-1] != char:
            stack.append(char)
        else:
            stack.pop()

    result = ''.join(stack)
    return result

if __name__ == "__main__":
    s = input()

    print(remove_adjacent_duplicate(s))
```

Valid Parentheses

Java Code:

```
class Solution {
    public boolean isValid(String s) {
        Stack<Character> stack = new Stack<>();
        int i = 0;

        while(i < s.length()) {
            char ch = s.charAt(i);

            if(stack.isEmpty() || ch == '(' || ch == '[' || ch == '{') {
                stack.push(ch);
            } else {
                if((ch == ')' && stack.peek() == '(') || (ch == ']' && stack.peek() == '[') ||
                (ch == '}' && stack.peek() == '{')) {
                    stack.pop();
                } else {
                    return false;
                }
            }
            i++;
        }

        return stack.isEmpty();
    }
}
```

```

        return false;
    }
}
i++;
}

if(stack.size() == 0) {
    return true;
} else{
    return false;
}
}
}
}

```

C++ Code:

```

#include <iostream>
#include <stack>

class Solution {
public:
    bool isValid(std::string s) {
        std::stack<char> stack;
        int i = 0;

        while (i < s.length()) {
            char ch = s[i];

            if (stack.empty() || ch == '(' || ch == '[' || ch == '{') {
                stack.push(ch);
            } else {
                if ((ch == ')' && stack.top() == '(') || (ch == ']' && stack.top() == '[') || (ch == '}' &&
stack.top() == '{')) {
                    stack.pop();
                } else {
                    return false;
                }
            }
            i++;
        }

        return stack.empty();
    }
}

```

```
};
```

Python Code:

```
class Solution:
    def isValid(self, s: str) -> bool:
        stack = []
        i = 0

        while i < len(s):
            ch = s[i]

            if not stack or ch == '(' or ch == '[' or ch == '{':
                stack.append(ch)
            else:
                if (ch == ')' and stack[-1] == '(') or (ch == ']' and stack[-1] == '[') or (ch == '}' and stack[-1] == '{'):
                    stack.pop()
                else:
                    return False
            i += 1

        return not stack
```

Min Stack

Java Code:

```
class MinStack {
    Stack <Integer> st = new Stack<>();
    Stack <Integer> min = new Stack<>();
    public MinStack() {
    }
    public void push(int val) {
        if(st.size() == 0) {
            st.push(val);
            min.push(val);
        }
        else if(val < min.peek()) {
            min.push(val);
        }
        else {
            min.push(min.peek());
        }
    }
    public int pop() {
        if(st.size() == 0) {
            return -1;
        }
        int val = st.pop();
        if(val == min.peek()) {
            min.pop();
        }
        return val;
    }
    public int top() {
        if(st.size() == 0) {
            return -1;
        }
        return st.peek();
    }
    public int getMin() {
        if(st.size() == 0) {
            return -1;
        }
        return min.peek();
    }
}
```

```

}

st.push(val);

int temp = Math.min(val, min.peek());
min.push(temp);
}

public void pop() {
st.pop();
min.pop();
}

public int top() {
return st.peek();
}

public int getMin() {
return min.peek();
}
}

```

C++ Code:

```

#include <stack>
#include <algorithm>

class MinStack {
private:
    std::stack<int> st;
    std::stack<int> minStack;

public:
    MinStack() {

    }

    void push(int val) {
        if (st.empty()) {
            st.push(val);
            minStack.push(val);
            return;
        }

        st.push(val);
        int temp = std::min(val, minStack.top());

```

```

        minStack.push(temp);
    }

void pop() {
    st.pop();
    minStack.pop();
}

int top() {
    return st.top();
}

int getMin() {
    return minStack.top();
}
};


```

Python Code:

```

class MinStack:

    def __init__(self):
        self.stack = []
        self.minStack = []

    def push(self, val: int) -> None:
        self.stack.append(val)
        if not self.minStack or val <= self.minStack[-1]:
            self.minStack.append(val)

    def pop(self) -> None:
        if self.stack:
            if self.stack[-1] == self.minStack[-1]:
                self.minStack.pop()
            self.stack.pop()

    def top(self) -> int:
        if self.stack:
            return self.stack[-1]

    def getMin(self) -> int:
        if self.minStack:
            return self.minStack[-1]

```

Asteroid Collision

Solution Vid:

<https://youtu.be/X22B9nrLQYU>

Java Code:

```
public int[] asteroidCollision(int[] asteroids) {  
    Stack<Integer> st = new Stack<>();  
    int n = asteroids.length;  
    int i=0;  
    while(i < n){  
        if(asteroids[i] < 0){  
            if(st.size()==0 || st.peek()<0){  
                st.push(asteroids[i]);  
                i++;  
            }  
            else if(st.peek() == Math.abs(asteroids[i])){  
                st.pop();  
                i++;  
            }  
            else if(st.peek() < Math.abs(asteroids[i])){  
                st.pop();  
            }  
            else if(st.peek() > Math.abs(asteroids[i])){  
                i++;  
            }  
        }else{  
            st.push(asteroids[i]);  
            i++;  
        }  
    }  
    int[] ans = new int[st.size()];  
    for(int j=ans.length-1;j>=0;j--){  
        ans[j] = st.pop();  
    }  
    return ans;  
}
```

```
}

return ans;
}
```

C++ Code:

```
#include <vector>
#include <stack>

std::vector<int> asteroidCollision(std::vector<int>& asteroids) {
    std::stack<int> st;
    int n = asteroids.size();
    int i = 0;

    while (i < n) {
        if (asteroids[i] < 0) {
            if (st.empty() || st.top() < 0) {
                st.push(asteroids[i]);
                i++;
            } else if (st.top() == abs(asteroids[i])) {
                st.pop();
                i++;
            } else if (st.top() < abs(asteroids[i])) {
                st.pop();
            } else if (st.top() > abs(asteroids[i])) {
                i++;
            }
        } else {
            st.push(asteroids[i]);
            i++;
        }
    }

    std::vector<int> ans(st.size());
    int j = ans.size() - 1;
    while (!st.empty()) {
        ans[j] = st.top();
        st.pop();
        j--;
    }

    return ans;
}
```

Python Code:

```
def asteroidCollision(asteroids):
    st = []
    n = len(asteroids)
    i = 0

    while i < n:
        if asteroids[i] < 0:
            if not st or st[-1] < 0:
                st.append(asteroids[i])
                i += 1
            elif st[-1] == abs(asteroids[i]):
                st.pop()
                i += 1
            elif st[-1] < abs(asteroids[i]):
                st.pop()
            elif st[-1] > abs(asteroids[i]):
                i += 1
        else:
            st.append(asteroids[i])
            i += 1

    return st

# Example usage:
asteroids = [5, 10, -5]
result = asteroidCollision(asteroids)
print(result) # Output: [5, 10]
```

Validate Stack

Solution Vid:

<https://youtu.be/1Qe1N2pw8gw>

Java Code:

```
class Solution {
    public boolean validateStackSequences(int[] pushed, int[] popped) {
        Stack<Integer> st = new Stack<>();
        ...
```

```

int i=0;
int j=0;

while(i<pushed.length && j<popped.length) {
    if(st.size()==0) {
        st.push(pushed[i]);
        i++;
        continue;
    }

    if(popped[j] == st.peek()) {
        st.pop();
        j++;
    } else{
        st.push(pushed[i]);
        i++;
    }
}

while(j < popped.length && popped[j] == st.peek()) {
    st.pop();
    j++;
}

if(st.size()>0) {
    return false;
} else{
    return true;
}
}
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <stack>

class Solution {

```

```

public:
    bool validateStackSequences(std::vector<int>& pushed, std::vector<int>& popped) {
        std::stack<int> st;

        int i = 0;
        int j = 0;

        while (i < pushed.size() && j < popped.size()) {
            if (st.empty()) {
                st.push(pushed[i]);
                i++;
                continue;
            }

            if (popped[j] == st.top()) {
                st.pop();
                j++;
            } else {
                st.push(pushed[i]);
                i++;
            }
        }

        while (j < popped.size() && popped[j] == st.top()) {
            st.pop();
            j++;
        }

        return st.empty();
    }
};

int main() {
    // You need to create the input arrays and call the function here.
    return 0;
}

```

Python Code:

```

class Solution:
    def validateStackSequences(self, pushed, popped):
        stack = []
        i, j = 0, 0

        while i < len(pushed) and j < len(popped):

```

```
if not stack:  
    stack.append(pushed[i])  
    i += 1  
elif popped[j] == stack[-1]:  
    stack.pop()  
    j += 1  
else:  
    stack.append(pushed[i])  
    i += 1  
  
while j < len(popped) and popped[j] == stack[-1]:  
    stack.pop()  
    j += 1  
  
return not stack
```

Example usage:

You need to create the input lists and call the function here.



Today's Agenda

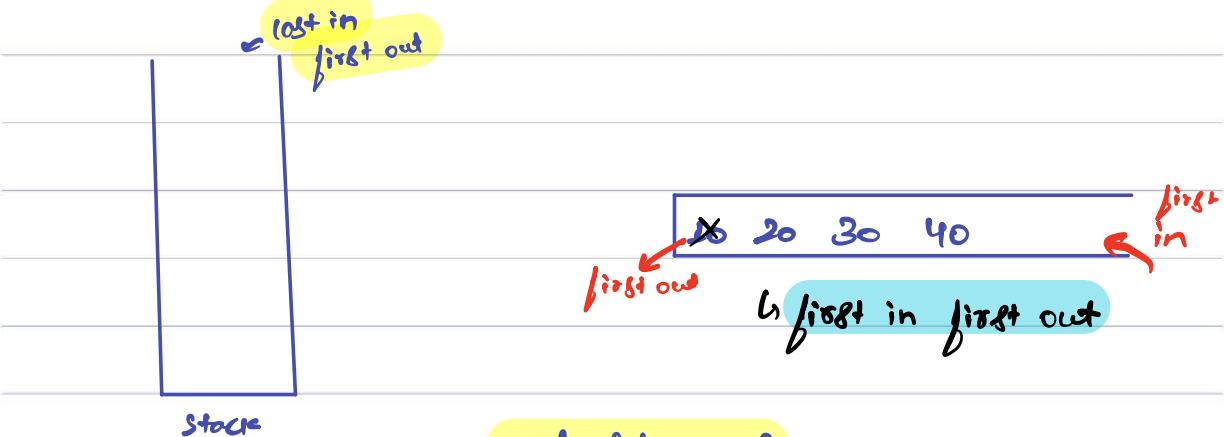
- ↳ Queue basics
- ↳ Reverse first K ele in queue
- ↳ implement queue using stack
- ↳ Kth number using only 1 & 2
- ↳ first non repeating character
- ↳ ... - - - - -



AlgoPrep



// Introduction



real life ex:

i) line

ii) task Scheduling

Queue initialize

↳ Queue < Integer > q = new LinkedList<>();

Operations:



q.add(x) → insert or at end of queue.

q.remove() → delete element from front and return.

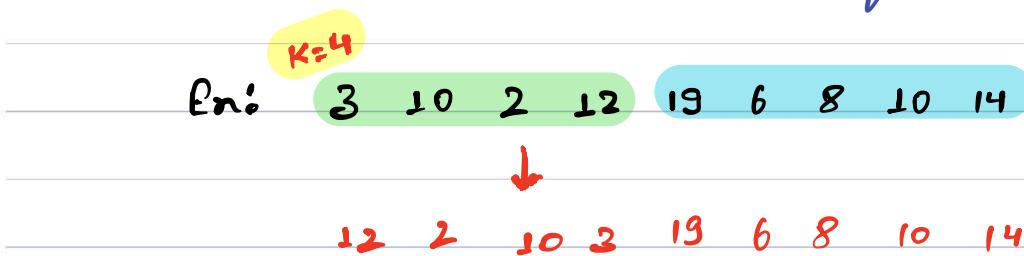
q.peek() → Return the front element.

q.size() → no. of elements.



Q) Reverse first K elements

↳ Given a Queue, Reverse its first K elements.



1/ idea

q:



Step 1:

↳ Push K elements from Queue \rightarrow Stack

Step 2:

↳ Put K elements back from Stack to Queue.

Step 3:

↳ Remove the first $n-K$ elements from the front and add at the end.



// Pseudo Code

Queue < Queue > ReverseKelements (Queue < > q, int k) {

Stack < integer > s = new Stack < >();

int n = q.size();

for (int i=1; i<=k; i++) {

s.push(q.remove());

T.C: O(k) + O(k) + O(n-k)

= O(n+k) ≈ O(n)

}

S.C: O(k)
if
k:=n

for (int i=1; i<=k; i++) {

int temp = s.pop();

q.add(temp);

3

for (int i=1; i<=n-k; i++) {

int temp = q.remove();

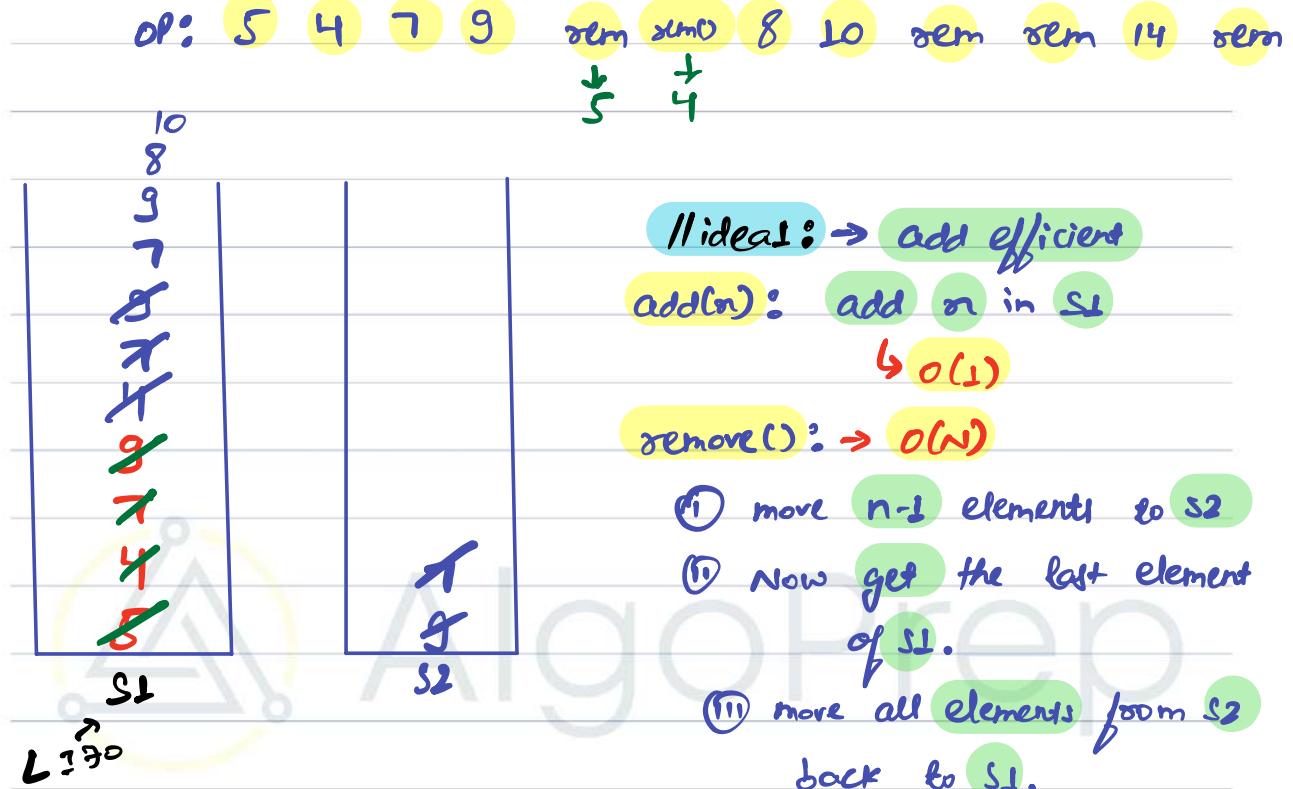
q.add(temp);

return q;

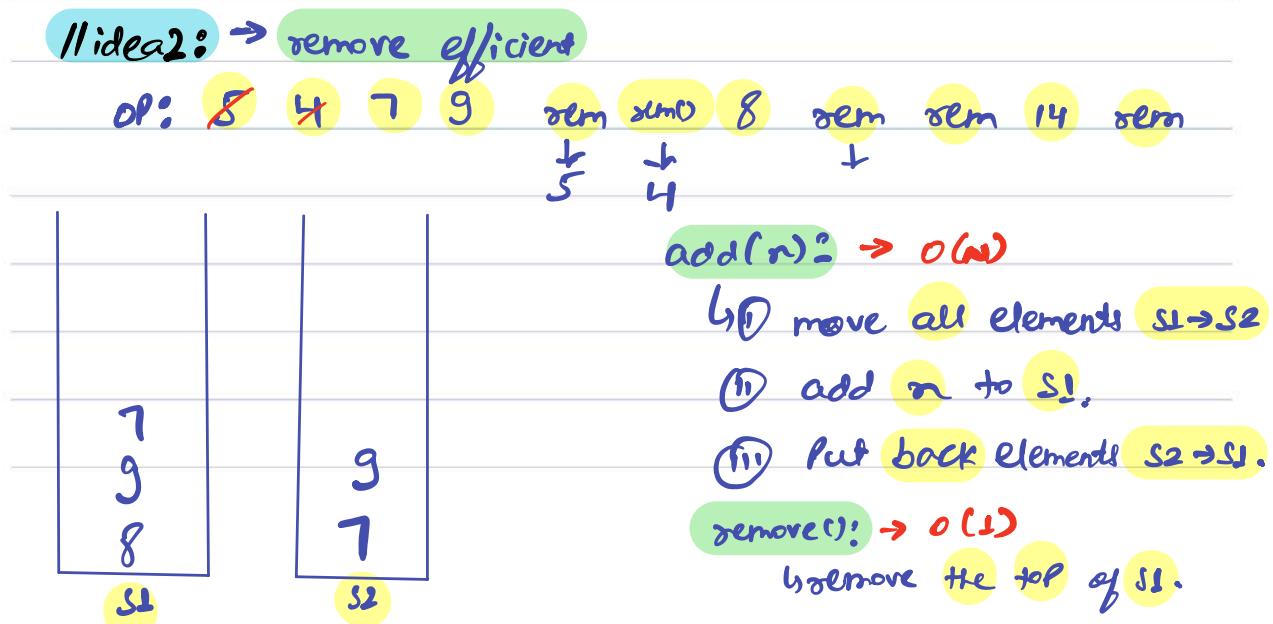
3



Q) Implement Queue using Stacks



S.C: O(1)

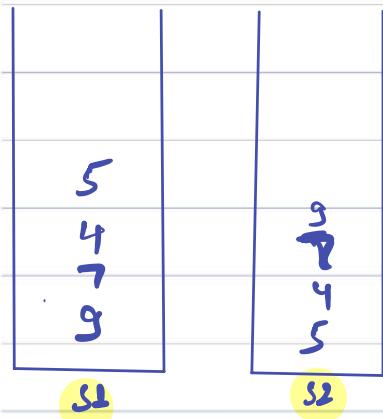


select



// idea 2: \rightarrow remove efficient

OP: 5 4 7 9 \uparrow rem sum 8 rem rem 14 rem



$\text{add}(n) \Rightarrow O(n)$

i) move all elements $S1 \rightarrow S2$

ii) add n to $S1$.

iii) Put back elements $S2 \rightarrow S1$.

$\text{remove}() \Rightarrow O(1)$

ii) remove the top of $S1$.



AlgoPrep

Break till 9:27 AM



Q) Kth Number

↳ Generate Kth number in series using digits 1 and 2 only.

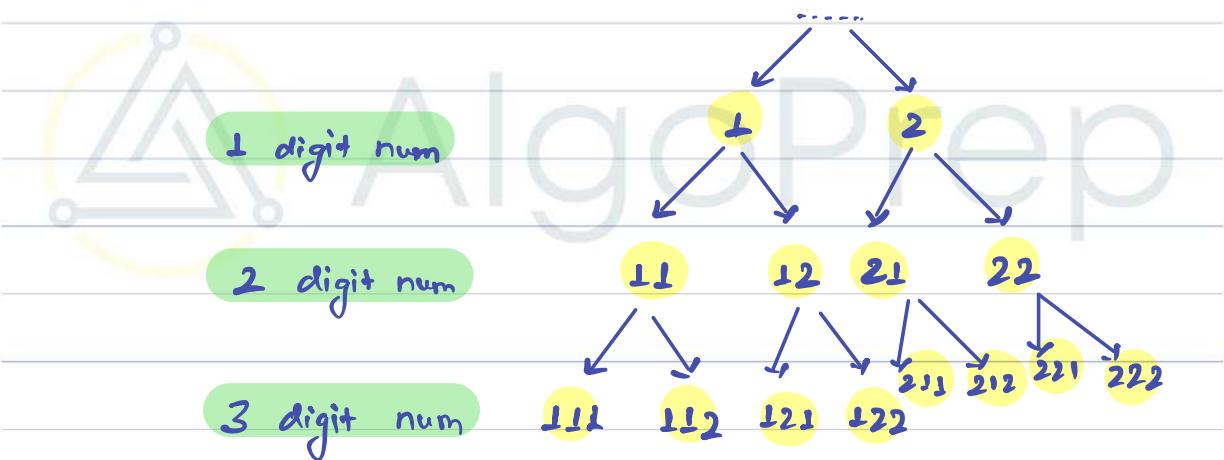
K=5

1 2 11 12 21

K=7

1 2 11 12 21 22 111

//idea



//Algorithm

K=5

q:

~~1 2 1 1 1 2 2 1 1 1 2 1 2 2~~

↳ ans = 21



// Pseudo Code

String kthnumber (int k) {

Queue < String > q = new LinkedList<>();

q.add ("1");

q.add ("2");

String ans = "";

for (int i=1; i<=k; i++) {

String temp = q.remove();

ans = temp;

q.add (temp + "1");

q.add (temp + "2");

}

return ans;

}

K=5



String Kthnumber (int k) {

Queue < String > q;

q.add ("1");

q.add ("2");

String ans = "";

for (int i=1; i<=k; i++) {

String temp = q.remove();

ans = temp;

q.add (temp + "1");

q.add (temp + "2");

}

return ans;

6. ans = 21

2 1 2 2 11 12 21 22 111 112
121 122 211 212

ans = ""

i	temp	ans
1	1	1
2	2	2
3	11	11
4	12	12
5	21	21



AlgoPrep



(Q)

- ↳ Generate k th Palindrome number in series using digits 1 and 2 only
Note: Only consider even digit numbers.

$K=5 \Rightarrow 11 \quad 22 \quad 1111 \quad 1221 \quad 2112$

Idea!

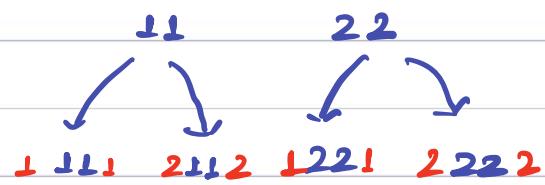
Keep generating numbers using 1 and 2, check for even Palindrome length and return k th one.

Idea 2

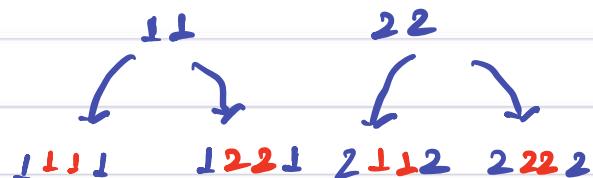
→ insert same number at end

2 digit Palindrome

4 digit Palindrome



→ insert same numbers at middle





// Pseudocode

String kthnumber (int k) {

Queue < String > q = new LinkedList<>();

q.add ("1");

q.add ("22");

String ans = "";

for (int i=1; i<=k; i++) {

String temp = q.remove();

ans = temp;

String left = temp.substring(0, temp.length/2);

String right = temp.substring($\frac{\text{temp.length}}{2}$, temp.length)

q.add (left + "1" + right);

q.add (left + "22" + right);

return ans;

↳ String s = "Hello";

s.substring (s, e);

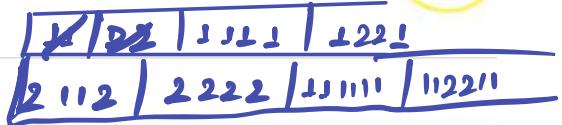
String temp1 = s.substring (0, 3) ^{inclusive} ⇒ "Hel"

String temp2 = s.substring (0, 4) ^{exclusive} ⇒ "Hell"

String temp3 = s.substring (2, 5) ^{exclusive} ⇒ "llo"



```
2. add("11");
2. add("22");
String ans = "";
for (int i=1; i<=k; i++) {
    String temp = q.dequeue();
    ans = temp;
    String left = temp.substring(0, temp.length/2);
    String right = temp.substring(temp.length/2, temp.length);
    q.add(left + "11" + right);
    q.add(left + "22" + right);
}
return ans;
```



i	temp	temp.length	left	right	ans
1	11	2	1	1	11
2	22	2	2	2	22
3	1111	4	11	11	



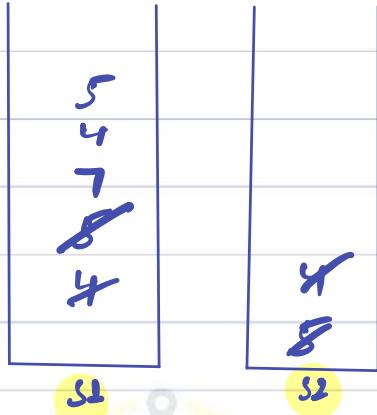
AlgoPrep



Doubts

remove efficient

OP: 5 4 7 9 ~~rem rem~~ 8 ~~rem rem~~ 14 ~~rem~~



$\text{add}(n) \Rightarrow O(n)$

↳ move all elements $S1 \rightarrow S2$

(i) add n to $S1$.

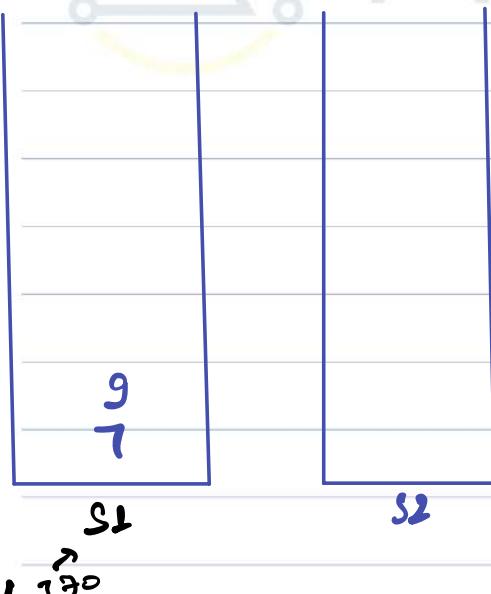
(ii) Put back elements $S2 \rightarrow S1$.

$\text{remove}() \Rightarrow O(1)$

↳ remove the top of $S1$.

add efficient

OP: 8 4 7 9 ~~rem rem~~ 8 ~~rem rem~~ 14 ~~rem~~



// ideal: \Rightarrow add efficient

$\text{add}(n) \Rightarrow$ add n in $S1$

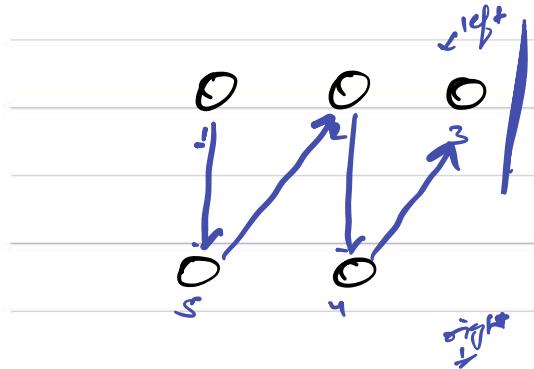
↳ $O(1)$

$\text{remove}() \Rightarrow O(n)$

(i) move $n-1$ elements to $S2$

(ii) Now get the last element of $S1$.

(iii) move all elements from $S2$ back to $S1$.



AlgoPrep

Reverse K Elements

Java Code:

```
public Queue<Integer> modifyQueue(Queue<Integer> q, int k) {  
    // add code here.  
  
    Stack<Integer> st = new Stack<>();  
  
    for(int i=1;i<=k;i++){  
        st.push(q.remove());  
    }  
  
    for(int i=1;i<=k;i++){  
        q.add(st.pop());  
    }  
  
    for(int i=1;i<=q.size()-k;i++){  
        q.add(q.remove());  
    }  
  
    return q;  
}
```

C++ Code:

```
#include <iostream>  
#include <queue>  
#include <stack>  
  
std::queue<int> modifyQueue(std::queue<int> q, int k) {  
    std::stack<int> st;  
  
    // Push the first 'k' elements into a stack  
    for (int i = 1; i <= k; i++) {  
        st.push(q.front());  
        q.pop();  
    }  
  
    // Push the elements from the stack back into the queue  
    for (int i = 1; i <= k; i++) {  
        q.push(st.top());  
        st.pop();  
    }
```

```

}

// Rotate the remaining elements in the queue by 'q.size() - k' positions
for (int i = 1; i <= q.size() - k; i++) {
    q.push(q.front());
    q.pop();
}

return q;
}

int main() {
    std::queue<int> q;
    int k;
    // Populate 'q' and 'k' with appropriate values here

    std::queue<int> modifiedQueue = modifyQueue(q, k);

    // Print the modified queue
    while (!modifiedQueue.empty()) {
        std::cout << modifiedQueue.front() << " ";
        modifiedQueue.pop();
    }

    return 0;
}

```

Python Code:

```

from queue import Queue

def modifyQueue(q, k):
    stack = []

    # Push the first 'k' elements into a stack
    for i in range(k):
        stack.append(q.get())

    # Push the elements from the stack back into the queue
    for i in range(k):
        q.put(stack.pop())

    # Rotate the remaining elements in the queue by 'q.qsize() - k' positions
    for i in range(q.qsize() - k):
        q.put(q.get())

```

```

return q

# Example usage:
q = Queue()
# Populate 'q' and 'k' with appropriate values here

modified_queue = modifyQueue(q, k)

# Print the modified queue
while not modified_queue.empty():
    print(modified_queue.get(), end=" ")

```

Queue Using Stack

Java Code:

```

class StackQueue
{
    Stack<Integer> s1 = new Stack<Integer>();
    Stack<Integer> s2 = new Stack<Integer>();

    //Function to push an element in queue by using 2 stacks.
    void Push(int x){
        // Your code here
        s1.push(x);
    }

    //Function to pop an element from queue by using 2 stacks.
    int Pop()
    {
        if(s1.size() == 0){
            return -1;
        }
        // Your code here
        int n = s1.size();
        for(int i=1;i<=n-1;i++){
            s2.push(s1.pop());
        }

        int ans = s1.pop();

        for(int i=1;i<=n-1;i++){

```

```

        s1.push(s2.pop());
    }

    return ans;
}
}

```

C++ Code:

```

#include <iostream>
#include <stack>

class StackQueue {
private:
    std::stack<int> s1;
    std::stack<int> s2;

public:
    // Function to push an element into the queue.
    void push(int x) {
        s1.push(x);
    }

    // Function to pop an element from the queue.
    int pop() {
        if (s1.empty()) {
            return -1; // Queue is empty.
        }

        // Move elements from s1 to s2 for reversal.
        while (!s1.empty()) {
            s2.push(s1.top());
            s1.pop();
        }

        // Pop the front element from s2.
        int frontElement = s2.top();
        s2.pop();

        // Move elements back from s2 to s1.
        while (!s2.empty()) {
            s1.push(s2.top());
            s2.pop();
        }
    }

    return frontElement;
}

```

```

    }
};

int main() {
    StackQueue sq;

    // Example usage:
    sq.push(1);
    sq.push(2);
    sq.push(3);

    std::cout << sq.pop() << std::endl; // Output: 1
    std::cout << sq.pop() << std::endl; // Output: 2

    sq.push(4);
    std::cout << sq.pop() << std::endl; // Output: 3
    std::cout << sq.pop() << std::endl; // Output: 4

    return 0;
}

```

Python Code:

```

class StackQueue:

    def __init__(self):
        self.s1 = []
        self.s2 = []

    # Function to push an element into the queue.
    def push(self, x: int) -> None:
        self.s1.append(x)

    # Function to pop an element from the queue.
    def pop(self) -> int:
        if not self.s1:
            return -1 # Queue is empty.

        # Move elements from s1 to s2 for reversal.
        while self.s1:
            self.s2.append(self.s1.pop())

        # Pop the front element from s2.
        front_element = self.s2.pop()

```

```

# Move elements back from s2 to s1.
while self.s2:
    self.s1.append(self.s2.pop())

return front_element

# Example usage:
sq = StackQueue()
sq.push(1)
sq.push(2)
sq.push(3)

print(sq.pop()) # Output: 1
print(sq.pop()) # Output: 2

sq.push(4)
print(sq.pop()) # Output: 3
print(sq.pop()) # Output: 4

```

Kth Number

Java Code:

```

import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
        should be named Solution. */
        Scanner scn = new Scanner(System.in);
        int k = scn.nextInt();
        System.out.println(kth(k));

    }

    public static String kth(int k){
        Queue<String> q = new LinkedList<>();
        q.add("1");
        q.add("2");

        String ans = "";

```

```

        for(int i=0;i<k;i++){
            String temp = q.remove();
            ans = temp;
            q.add(temp+"1");
            q.add(temp+"2");
        }

        return ans;
    }
}

```

C++ Code:

```

#include <iostream>
#include <queue>
#include <string>

std::string kth(int k) {
    std::queue<std::string> q;
    q.push("1");
    q.push("2");

    std::string ans;
    for (int i = 0; i < k; i++) {
        std::string temp = q.front();
        ans = temp;
        q.pop();
        q.push(temp + "1");
        q.push(temp + "2");
    }

    return ans;
}

int main() {
    int k;
    std::cin >> k;

    std::string result = kth(k);
    std::cout << result << std::endl;

    return 0;
}

```

Python Code:

```
from collections import deque

def kth(k):
    q = deque()
    q.append("1")
    q.append("2")

    ans = ""
    for i in range(k):
        temp = q.popleft()
        ans = temp
        q.append(temp + "1")
        q.append(temp + "2")

    return ans

if __name__ == "__main__":
    k = int(input())
    result = kth(k)
    print(result)
```

Even Palindrome kth Number

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        System.out.println(KPalindrome(k));

    }

    public static String KPalindrome(int k){
        Queue<String> q = new LinkedList<>();
        q.add("11");
```

```

q.add("22");

String ans = "";

for(int i=1;i<=k;i++){
    String temp = q.remove();
    ans = temp;
    String left = temp.substring(0,temp.length()/2);
    String right = temp.substring(temp.length()/2,temp.length());

    q.add(left+"11"+right);
    q.add(left+"22"+right);

}

return ans;
}
}

```

C++ Code:

```

#include <iostream>
#include <queue>
#include <string>

std::string KPalindrome(int k) {
    std::queue<std::string> q;
    q.push("11");
    q.push("22");

    std::string ans = "";

    for (int i = 1; i <= k; i++) {
        std::string temp = q.front();
        ans = temp;
        q.pop();
        std::string left = temp.substr(0, temp.length() / 2);
        std::string right = temp.substr(temp.length() / 2);

        q.push(left + "11" + right);
        q.push(left + "22" + right);
    }

    return ans;
}

```

```

int main() {
    int k;
    std::cin >> k;

    std::string result = KPalindrome(k);
    std::cout << result << std::endl;

    return 0;
}

```

Python Code:

```

from collections import deque

def KPalindrome(k):
    q = deque()
    q.append("11")
    q.append("22")

    ans = ""

    for i in range(1, k + 1):
        temp = q.popleft()
        ans = temp
        left = temp[:len(temp) // 2]
        right = temp[len(temp) // 2:]

        q.append(left + "11" + right)
        q.append(left + "22" + right)

    return ans

if __name__ == "__main__":
    k = int(input())
    result = KPalindrome(k)
    print(result)

```

Implement Stack Using Queues_HW

Solution Vid:

<https://youtu.be/W-aJKz1uFBs>

Java Code:

```
class MyStack {  
    Queue<Integer> q = new LinkedList<>();  
    public MyStack() {}  
  
    public void push(int x) {  
        q.add(x);  
    }  
  
    public int pop() {  
        int n = q.size();  
        for(int i=1;i<=n-1;i++) {  
            q.add(q.remove());  
        }  
  
        return q.remove();  
    }  
  
    public int top() {  
        int n = q.size();  
        for(int i=1;i<=n-1;i++) {  
            q.add(q.remove());  
        }  
  
        int ans = q.peek();  
        q.add(q.remove());  
        return ans;  
    }  
  
    public boolean empty() {  
        return q.size() == 0;  
    }  
}
```

C++ Code:

```
#include <iostream>
#include <queue>

class MyStack {
private:
    std::queue<int> q;

public:
    MyStack() {

    }

    void push(int x) {
        q.push(x);
    }

    int pop() {
        int n = q.size();
        for (int i = 1; i <= n - 1; i++) {
            q.push(q.front());
            q.pop();
        }

        int poppedElement = q.front();
        q.pop();

        return poppedElement;
    }

    int top() {
        int n = q.size();
        for (int i = 1; i <= n - 1; i++) {
            q.push(q.front());
            q.pop();
        }

        int topElement = q.front();
        q.push(q.front());
        q.pop();

        return topElement;
    }

    bool empty() {
        return q.empty();
    }
}
```

```

    }
};

int main() {
    MyStack stack;

    stack.push(1);
    stack.push(2);
    std::cout << stack.top() << std::endl; // Output: 2
    std::cout << stack.pop() << std::endl; // Output: 2
    std::cout << stack.empty() << std::endl; // Output: 0 (false)

    return 0;
}

```

Python Code:

```

from collections import deque

class MyStack:

    def __init__(self):
        self.q = deque()

    def push(self, x: int) -> None:
        self.q.append(x)
        n = len(self.q)
        for _ in range(n - 1):
            self.q.append(self.q.popleft())

    def pop(self) -> int:
        return self.q.popleft()

    def top(self) -> int:
        return self.q[0]

    def empty(self) -> bool:
        return len(self.q) == 0

# Example usage:
stack = MyStack()
stack.push(1)
stack.push(2)
print(stack.top()) # Output: 2
print(stack.pop()) # Output: 2

```

```
print(stack.empty()) # Output: False
```

First Non repeating Character_HW

Solution Vid:

<https://youtu.be/2dLbbLSXDI8>

Java Code:

```
class Solution
{
    public String FirstNonRepeating(String A)
    {
        // code here

        Queue<Character> q = new LinkedList<>();
        HashMap<Character, Integer> hm = new HashMap<>();
        char[] ans = new char[A.length()];

        for(int i=0;i<A.length();i++){
            char c = A.charAt(i);

            q.add(c);
            if(hm.containsKey(c) == true){
                int temp = hm.get(c);
                hm.put(c,temp+1);
            }else{
                hm.put(c,1);
            }

            while(q.size()>0 && hm.get(q.peek())> 1){
                q.remove();
            }

            if(q.size()>0){
                ans[i] = q.peek();
            }else{
                ans[i] = '#';
            }
        }
    }
}
```

```

    }

    return String.valueOf(ans);

}
}

```

C++ Code:

```

#include <iostream>
#include <queue>
#include <unordered_map>
#include <string>

class Solution {
public:
    std::string FirstNonRepeating(std::string A) {
        std::queue<char> q;
        std::unordered_map<char, int> hm;
        std::string ans(A.length(), ' ');

        for (int i = 0; i < A.length(); i++) {
            char c = A[i];

            q.push(c);
            if (hm.find(c) != hm.end()) {
                int temp = hm[c];
                hm[c] = temp + 1;
            } else {
                hm[c] = 1;
            }

            while (!q.empty() && hm[q.front()] > 1) {
                q.pop();
            }

            if (!q.empty()) {
                ans[i] = q.front();
            } else {
                ans[i] = '#';
            }
        }

        return ans;
    }
}

```

```

    }
};

int main() {
    Solution solution;
    std::string input;
    std::cin >> input;

    std::string result = solution.FirstNonRepeating(input);
    std::cout << result << std::endl;

    return 0;
}

```

Python Code:

```

from collections import deque, defaultdict

class Solution:

    def FirstNonRepeating(self, A: str) -> str:
        q = deque()
        hm = defaultdict(int)
        ans = [' '] * len(A)

        for i in range(len(A)):
            c = A[i]

            q.append(c)
            hm[c] += 1

            while q and hm[q[0]] > 1:
                q.popleft()

            if q:
                ans[i] = q[0]
            else:
                ans[i] = '#'

        return ''.join(ans)

# Example usage:
solution = Solution()
input_str = input()
result = solution.FirstNonRepeating(input_str)

```

```
print(result)
```



Today's agenda

↳ Trees Intro

↳ Naming convention

↳ Tree traversal

↳ Basic tree Problems



AlgoPrep



Linear

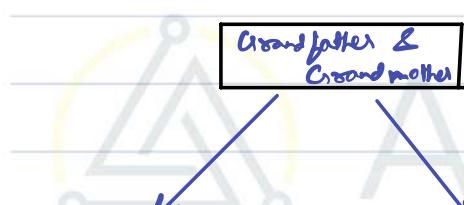
Arrays →

--	--	--	--	--

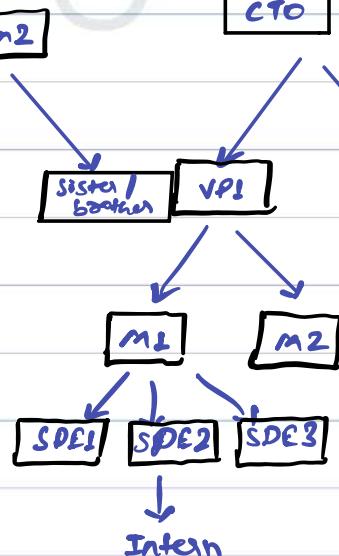
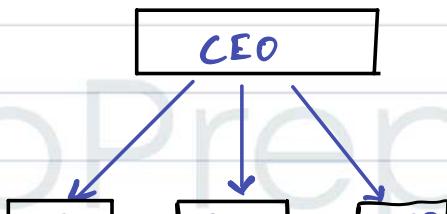
, LL, Hashmap, Stack, queues.

Hierarchical

family tree



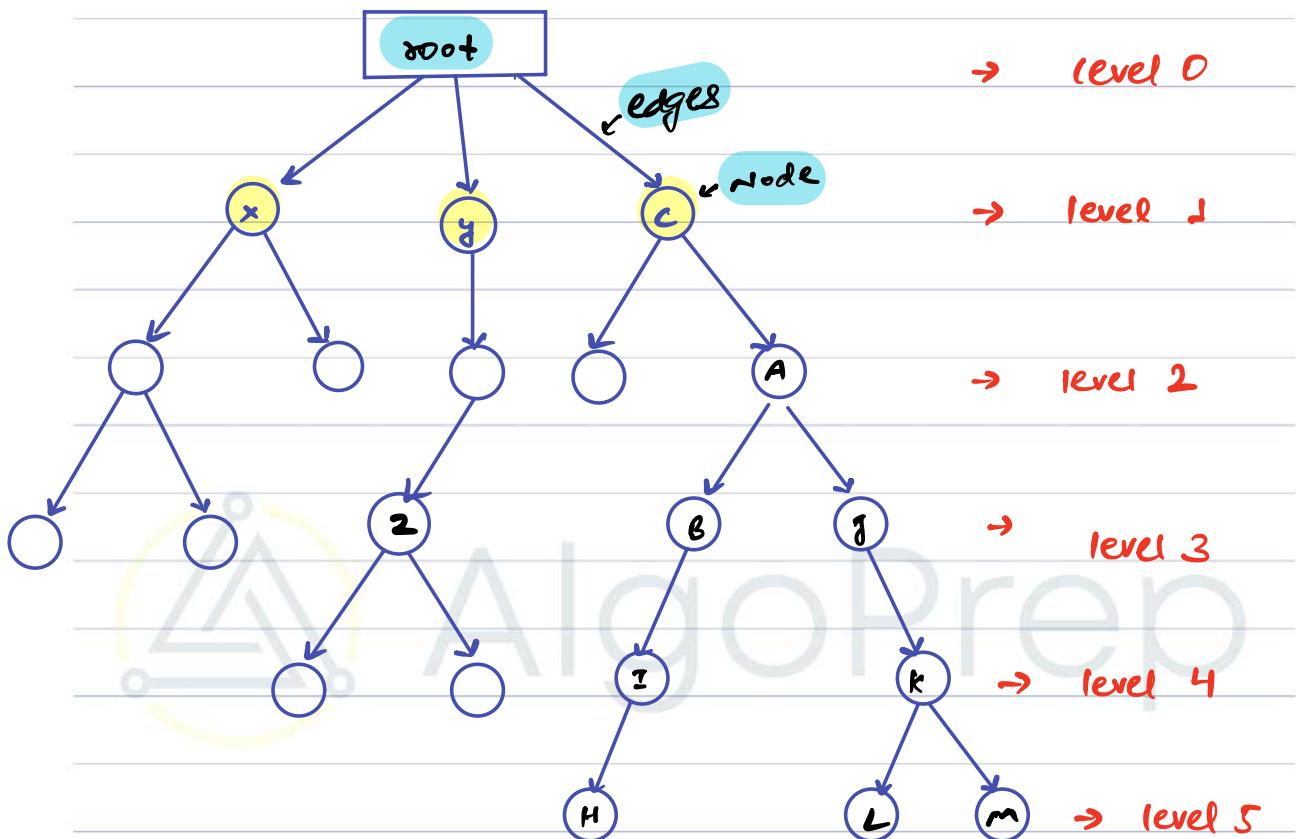
Org structure





// Tree

↳ Naming Convention



Parent & Child : **A** is Parent of **B**, **B** is child of **A**

Ancestors : All the nodes in the Path from root node to that node.

Descendents : All the nodes below the given node.

leaf Nodes : Nodes without children

Siblings : Nodes with same parent

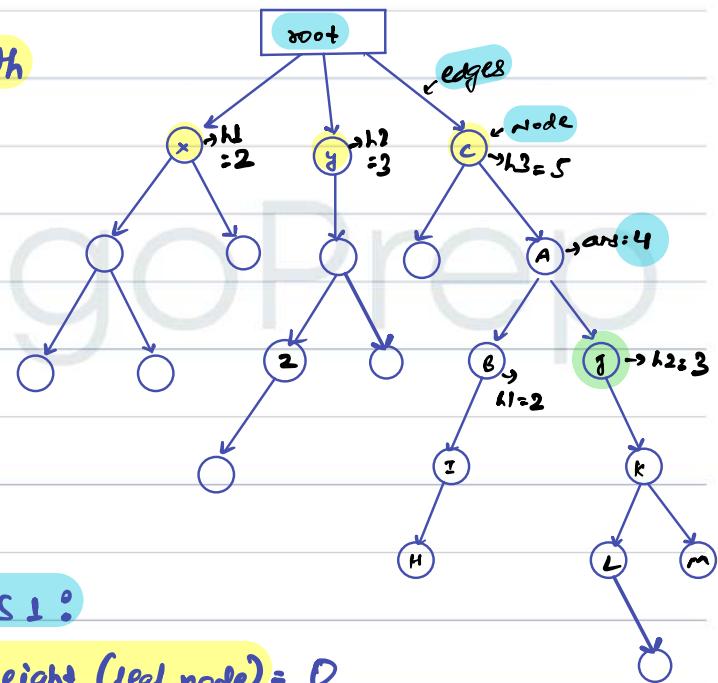


* Property of tree :

- ↳ we will have only 1 root Node.
- ↳ for every Node, there can be only 1 Parent.
- ↳ Cycle Not allowed

* Height (Node)

↳ length of longest Path
from node to any of its
descendent leaf nodes.



$$\text{Height}(j) = 2$$

Obs 1 :

Height (leaf node) = 0

$$\text{Height}(y) = 3$$

Obs 2 :

$$H(\text{Node}) = \max(\text{Height of Child nodes}) + 1$$

$$\text{Height}(A) = 4$$

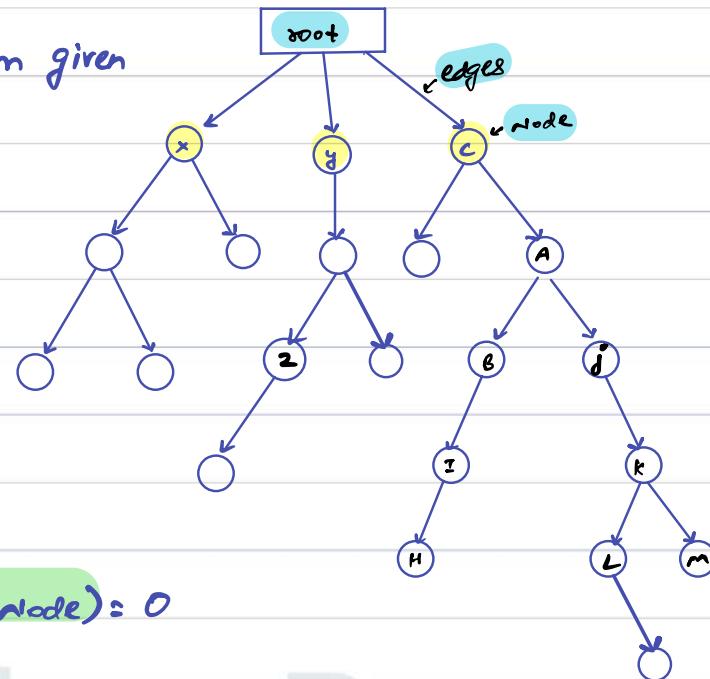


* Depth (Node)

↳ length of Path from given node to root node.

$$\text{Depth}(j) = 3$$

$$\text{Depth}(n) = 5$$

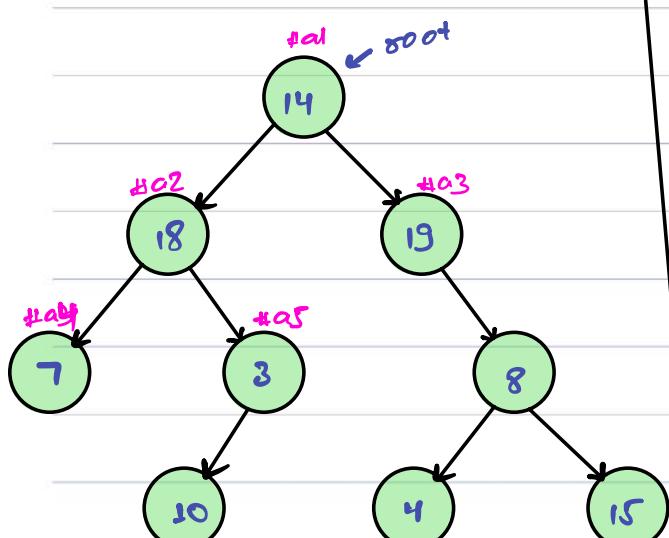


Obs 1: $\text{depth}(\text{Root node}) = 0$

Obs 2: $\text{depth}(\text{node}) = \text{depth}(\text{Parent node}) + 1$



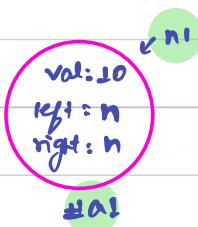
Binary tree: Every node can have atmost 2 child Nodes.



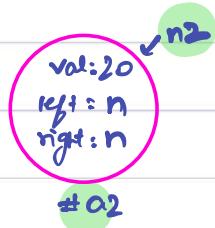
Class node L

```
int val;
Node left;
Node right;
node (int n){  
    val=n;
```

Node n1 = new Node (10);



Node n2 = new Node (20);

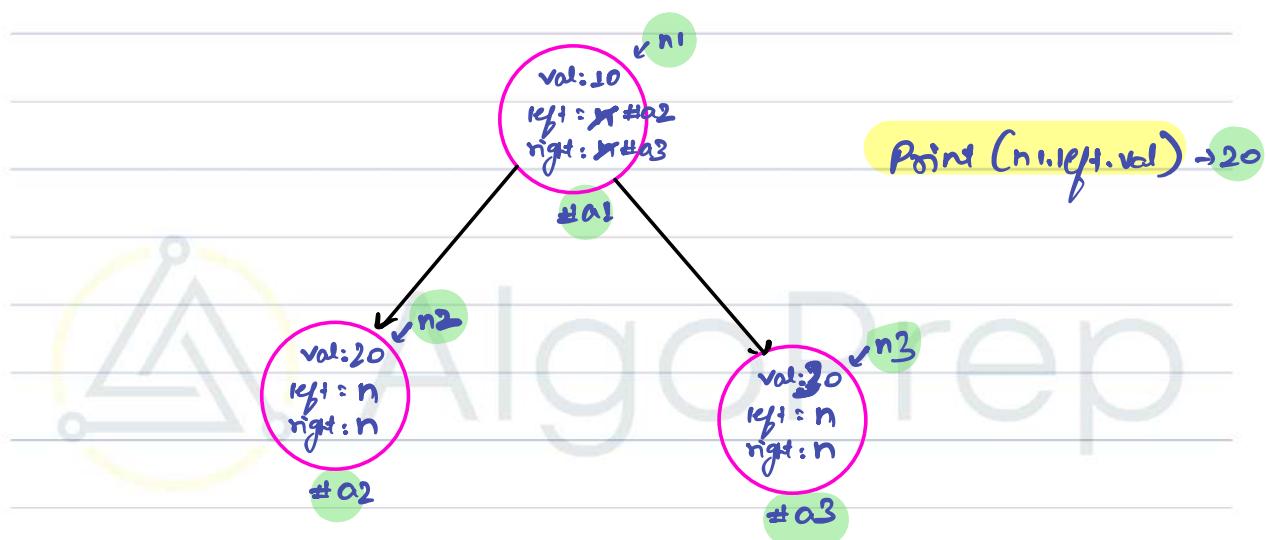
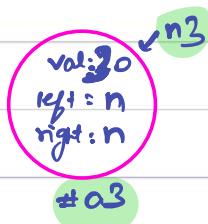


Class node L

```
int val;
Node left;
Node right;
node (int n){  
    val=n;
```



Node n₃ = new Node(30);



n₁.left = n₂;

n₁.right = n₃;

→ Don't worry about tree construction. (Serialization & deserialization)
↓
Solve the Problem for constructed tree.



//Tree traversal

breadth first search

↳ level order traversal (BFS)

b Recursion traversal

↳ Pre/Post/In CDFS

depth first search

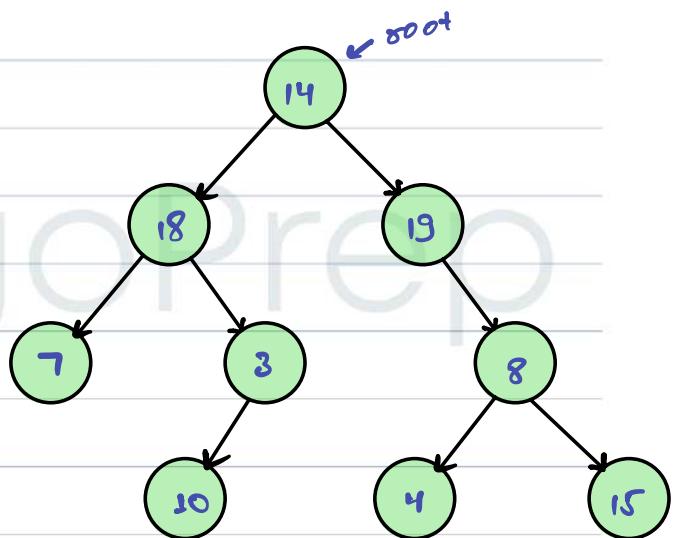
void traversal (Node root){

if (root == null) {

return;

traversal (root.left);

traversal (root.right);



}

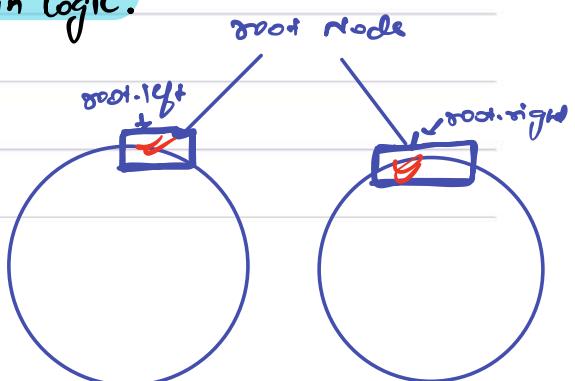
faith: Given root node, travel

every descendent of root node.

basecase:

if (node == null) {
return;

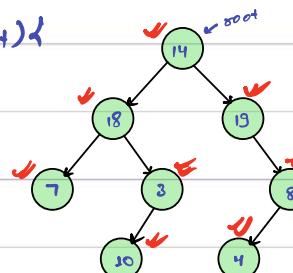
main logic:



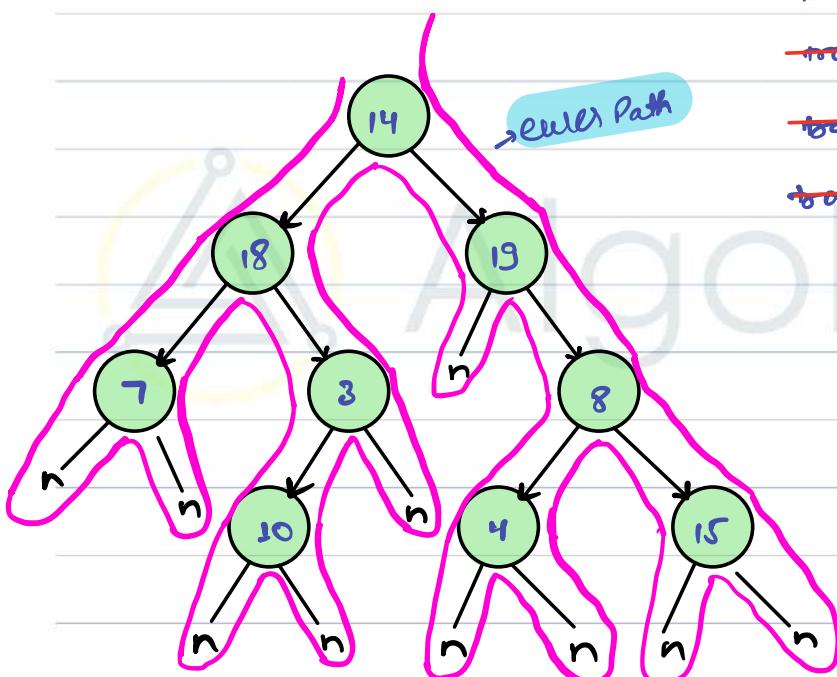
```

void traversal ( Node root) {
    1 if (root == null) {
    2     return;
    2 traversal (root.left);
    2 traversal (root.right);
}

```



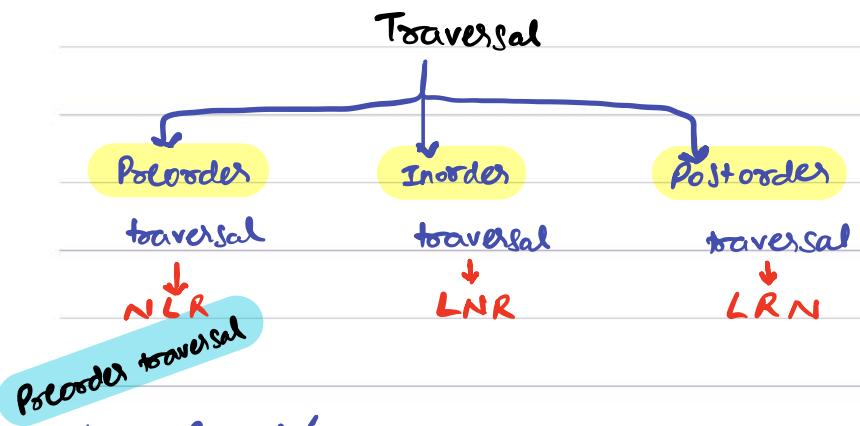
trav	root = 4	123
trav	root = 8	123
trav	root = null	1
trav	root = 19	123
trav	root = null	1
trav	root = null	1
trav	root = 10	123
trav	root = 2	123
trav	root = null	1
trav	root = null	1
trav	root = 7	123
trav	root = 8	123
trav	root = 14	123



↳ How many times you visited a Single node?
 ↳ 3 times

T.C: $O(3 \times n) \approx O(n)$

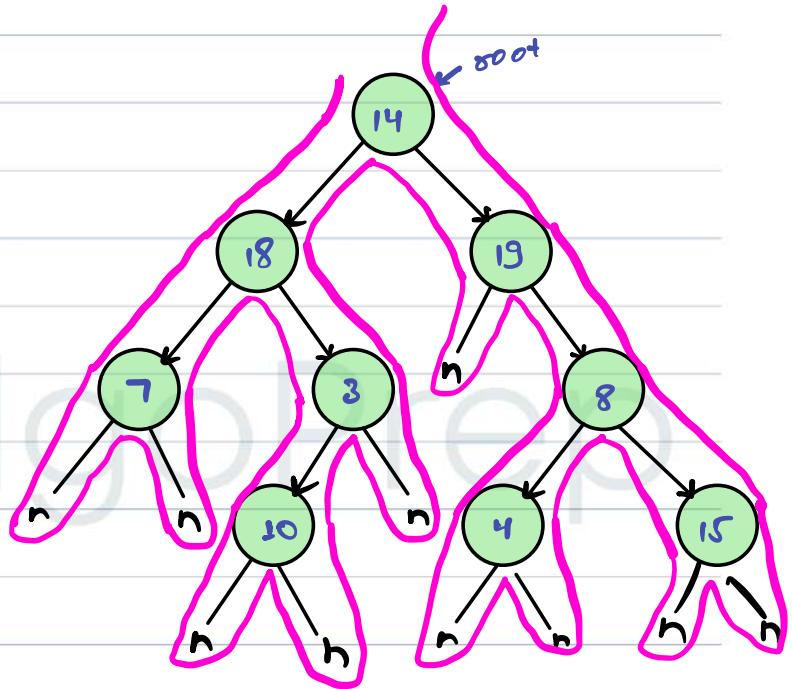
Break till 10:05 PM



```

void Preorder(Node root){
    if (root == null) {
        return;
    }
    System.out.print(root.val);
    traversal (root.left);
    traversal (root.right);
}
    
```

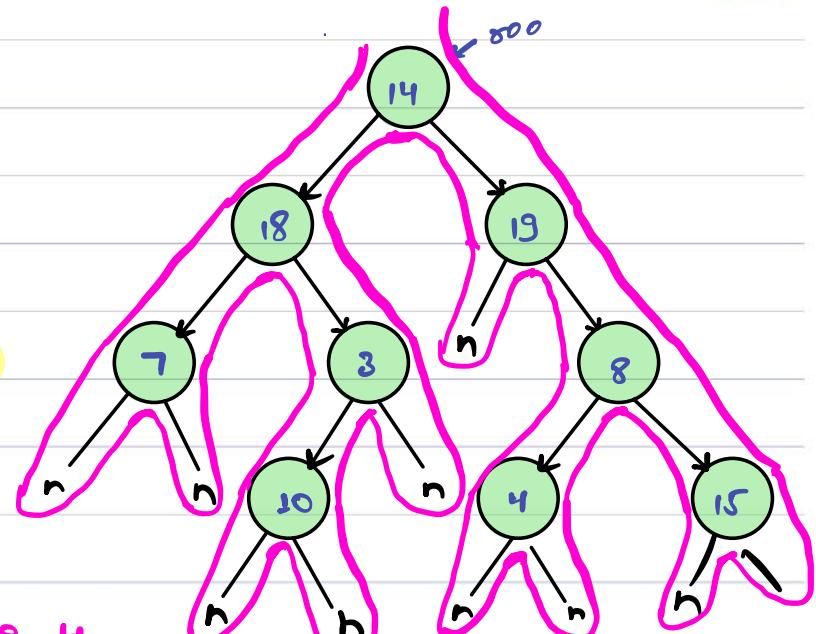
14 18 7 3 10
19 8 4 15



Inorder traversal
↳ LNR



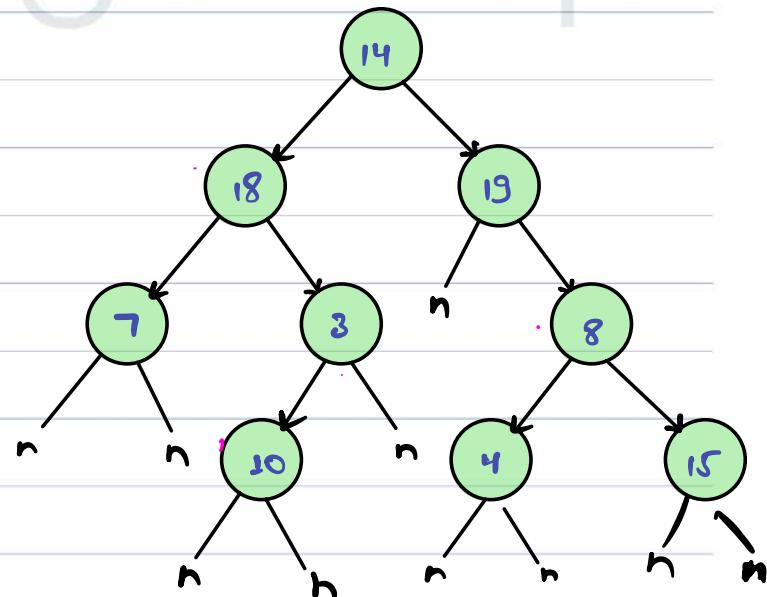
```
void Inorder ( Node root) {
    1 if (root == null) {
    2     return;
    3
    4 traversal (root.left);
    5 System.out.print (root.val);
    6 traversal (root.right);
    7
}
```

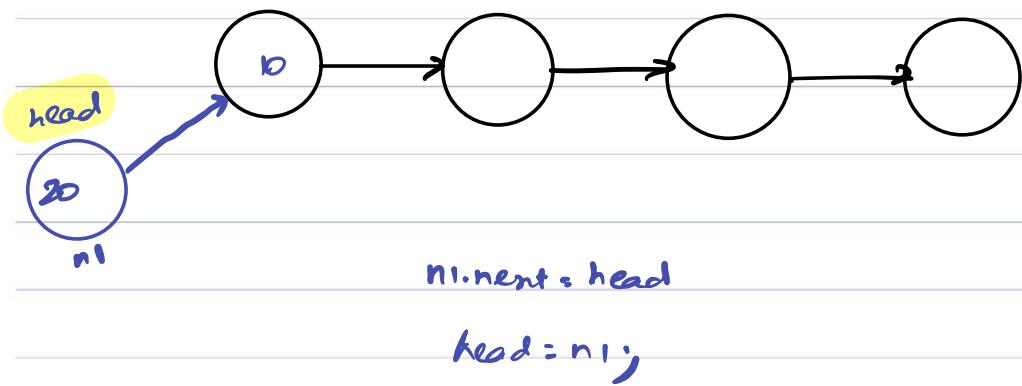


7 18 10 3 14 19 4
8 15

Postorder traversal
↳ LRN

```
void Postorder( Node root) {
    1 if (root == null) {
    2     return;
    3
    4 traversal (root.left);
    5 traversal (root.right);
    6 System.out.print (root.val);
    7
}
```





AlgoPrep

PreOrder Traversal

Java Code:

```
class Solution {  
    public List<Integer> preorderTraversal(TreeNode root) {  
        List<Integer> pre = new ArrayList<>();  
        preHelper(root, pre);  
        return pre;  
    }  
  
    public void preHelper(TreeNode root, List<Integer> pre) {  
        if (root == null) return;  
        pre.add(root.val);  
        preHelper(root.left, pre);  
        preHelper(root.right, pre);  
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <vector>  
  
struct TreeNode {  
    int val;  
    TreeNode* left;  
    TreeNode* right;  
    TreeNode(int x) : val(x), left(NULL), right(NULL) {}  
};  
  
class Solution {  
public:  
    std::vector<int> preorderTraversal(TreeNode* root) {  
        std::vector<int> pre;  
        preHelper(root, pre);  
        return pre;  
    }  
  
    void preHelper(TreeNode* root, std::vector<int>& pre) {  
        if (root == nullptr) return;  
        pre.push_back(root->val);
```

```

        preHelper(root->left, pre);
        preHelper(root->right, pre);
    }
};

int main() {
    // You need to create the binary tree and call the function here.
    return 0;
}

```

Python Code:

```

class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Solution:
    def preorderTraversal(self, root: TreeNode):
        pre = []
        self.pre_helper(root, pre)
        return pre

    def pre_helper(self, root: TreeNode, pre):
        if root is None:
            return
        pre.append(root.val)
        self.pre_helper(root.left, pre)
        self.pre_helper(root.right, pre)

# Example usage:
# You need to create the binary tree and call the function here.

```

Inorder Traversal

Java Code:

```
class Solution {  
    public List<Integer> inorderTraversal(TreeNode root) {  
        List<Integer> pre = new ArrayList<>();  
        preHelper(root, pre);  
        return pre;  
    }  
  
    public void preHelper(TreeNode root, List<Integer> pre) {  
        if (root == null) return;  
        preHelper(root.left, pre);  
        pre.add(root.val);  
        preHelper(root.right, pre);  
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <vector>  
  
struct TreeNode {  
    int val;  
    TreeNode* left;  
    TreeNode* right;  
    TreeNode(int x) : val(x), left(NULL), right(NULL) {}  
};  
  
class Solution {  
public:  
    std::vector<int> inorderTraversal(TreeNode* root) {  
        std::vector<int> in;  
        inorderHelper(root, in);  
        return in;  
    }  
  
    void inorderHelper(TreeNode* root, std::vector<int>& in) {
```

```

    if (root == nullptr) return;
    inorderHelper(root->left, in);
    in.push_back(root->val);
    inorderHelper(root->right, in);
}
};

int main() {
    // You need to create the binary tree and call the function here.
    return 0;
}

```

Python Code:

```

class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Solution:
    def inorderTraversal(self, root: TreeNode):
        in_order = []
        self.inorder_helper(root, in_order)
        return in_order

    def inorder_helper(self, root: TreeNode, in_order):
        if root is None:
            return
        self.inorder_helper(root.left, in_order)
        in_order.append(root.val)
        self.inorder_helper(root.right, in_order)

# Example usage:
# You need to create the binary tree and call the function here.

```

PostOrder Traversal

Java Code:

```
class Solution {  
    public List<Integer> postorderTraversal(TreeNode root) {  
        List<Integer> pre = new ArrayList<>();  
        preHelper(root, pre);  
        return pre;  
    }  
  
    public void preHelper(TreeNode root, List<Integer> pre) {  
        if (root == null) return;  
        preHelper(root.left, pre);  
        preHelper(root.right, pre);  
        pre.add(root.val);  
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <vector>  
  
struct TreeNode {  
    int val;  
    TreeNode* left;  
    TreeNode* right;  
    TreeNode(int x) : val(x), left(NULL), right(NULL) {}  
};  
  
class Solution {  
public:  
    std::vector<int> postorderTraversal(TreeNode* root) {  
        std::vector<int> post;  
        postorderHelper(root, post);  
        return post;  
    }  
  
    void postorderHelper(TreeNode* root, std::vector<int>& post) {  
        if (root == nullptr) return;  
        postorderHelper(root->left, post);  
        postorderHelper(root->right, post);  
        post.push_back(root->val);  
    }  
}
```

```

        post.push_back(root->val);
    }
};

int main() {
    // You need to create the binary tree and call the function here.
    return 0;
}

```

Python Code:

```

class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Solution:
    def postorderTraversal(self, root: TreeNode):
        post_order = []
        self.postorderHelper(root, post_order)
        return post_order

    def postorderHelper(self, root: TreeNode, post_order):
        if root is None:
            return
        self.postorderHelper(root.left, post_order)
        self.postorderHelper(root.right, post_order)
        post_order.append(root.val)

# Example usage:
# You need to create the binary tree and call the function here.

```



Today's Agenda

↳ Arraylist

↳ Size of the tree

↳ Sum of all nodes

↳ level order

↳ reverse level order



AlgoPrep



Arrays: `int[] arr = new int[10];`

→ in background is nothing but array

ArrayList: → dynamic array.

`List<Integer> ls = new ArrayList<>();`

ls 0 1 2 3 4
10 20 30 40 50

ls.add(10); → O(1)

ls.add(20);

ls.add(30);

ls.add(40);

ls.add(50);

ls.size(); → 5

ls.remove(idn) → O(1) ^{last idn}
O(n) ← other idn

iterate →
for (int i=0; i < ls.size(); i++) {
}



Q) Size of a tree

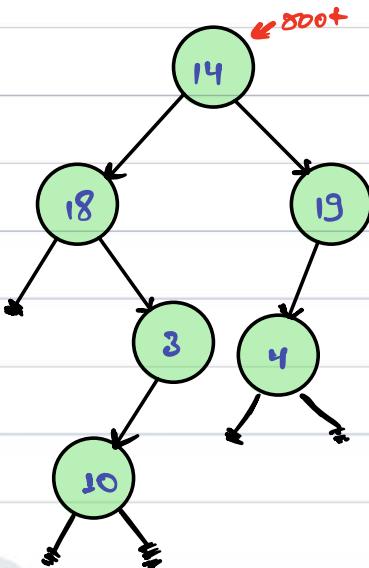
↳ Given a tree, calculate no. of nodes in it.

```
int size(node root) {
    if (root == null) { return 0; }

    int n = size(root.left);
    int y = size(root.right);
    return n+y+1;
}
```

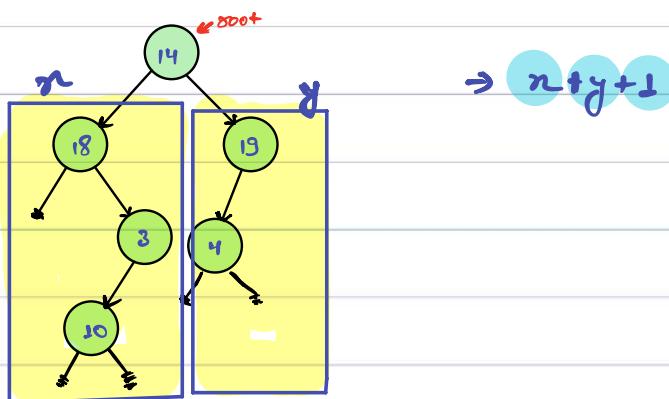
T.C: $O(N)$

S.C: $O(N)$



Faith: Given root node, find & return no. of nodes in that tree.

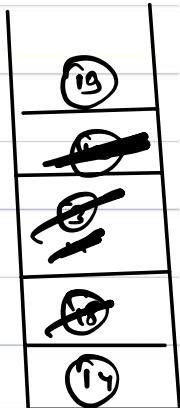
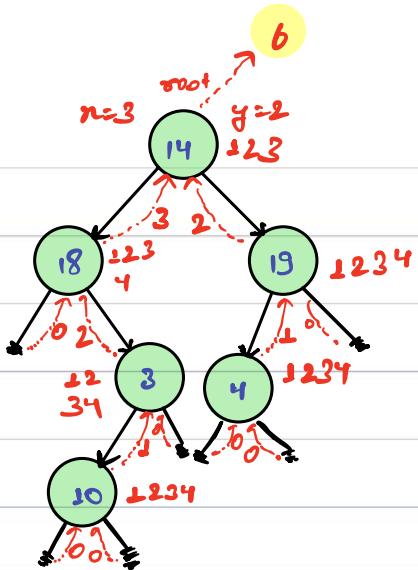
Main logic:



base case:



```
int size(node root) {  
    1 if (root == null) {return 0;}  
  
    2 int n = size(root.left);  
    3 int y = size(root.right);  
    4 return n+y+1;  
}
```



S.C: $O(\text{No. of Levels}) \approx O(\log n)$



AlgoPrep

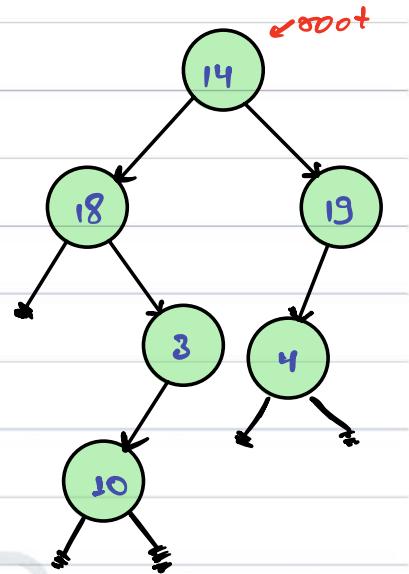


Q) Sum of a tree

Given a tree, calculate sum of all nodes data in it.

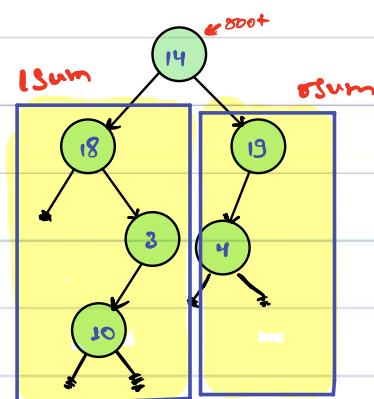
```
int sum (Node root){  
    if (root == null) { return 0; }  
}
```

```
int lsum = sum (root.left);  
int rsum = sum (root.right);  
return lsum + rsum + root.val;
```



Faith: Given `root` node, find & return sum of nodes in that tree.

Main logic:

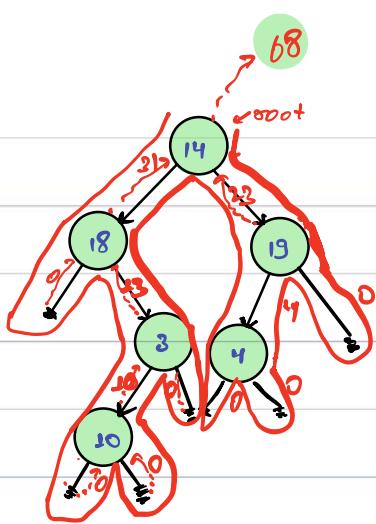


$\rightarrow lsum + rsum + root.val$

base case:



```
int sum (Node *root) {  
    if (root == null) {return 0;}  
  
    int lsum = sum (root.left);  
    int rsum = sum (root.right);  
    return lsum + rsum + root.val;  
}
```

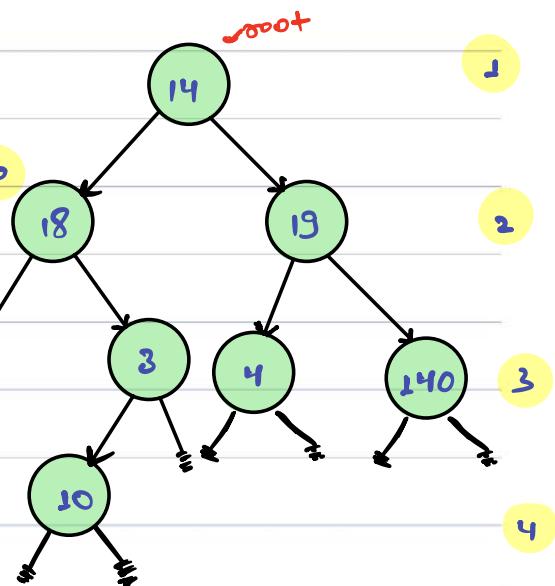


AlgoPrep



Q) Level order of tree

level order:
node queue:



Output:

14 18 19 3 4 140 10

II) Pseudo code

```

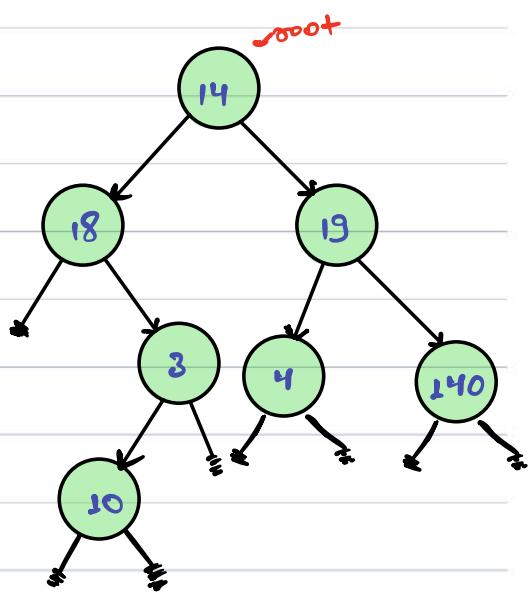
void levelOrder (Node root) {
    Queue < Node > q;
    q.add (root);
    while (q.size() > 0) {
        Node temp = q.remove();
        System.out.print (temp.val);
        if (temp.left != null) {
            q.add (temp.left);
        }
        if (temp.right != null) {
            q.add (temp.right);
        }
    }
}
    
```

T.C: O(n)

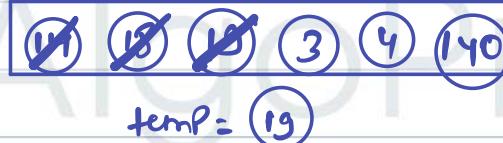
S.C: O(n)



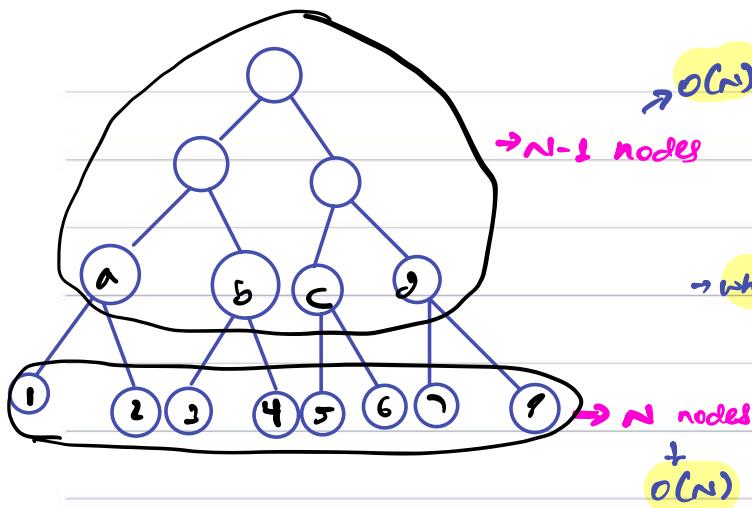
```
void levelOrder (Node root) {  
    Queue<Node> q;  
    q.add (root);  
  
    while (q.size() > 0) {  
        Node temp = q.remove();  
        System.out.print (temp.val);  
        if (temp.left != null) {  
            q.add (temp.left);  
        }  
        if (temp.right != null) {  
            q.add (temp.right);  
        }  
    }  
}
```



2



14 18 19



Total no. nodes = $2n-1$

$\downarrow O(n)$

→ why? → HW

6 8 7 9 1 2 3 4 5 6 7 8

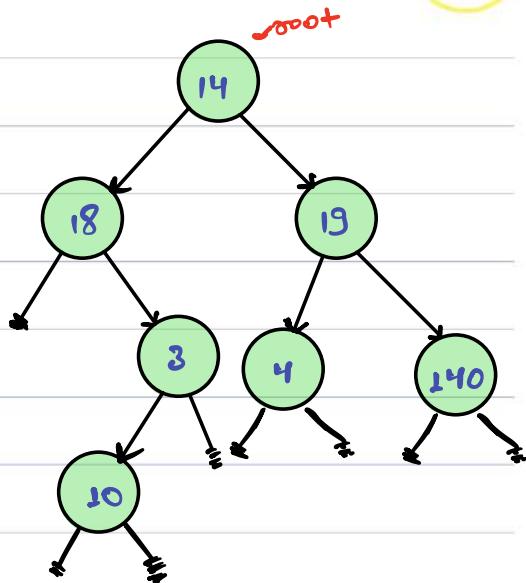


Break till 9:37 PM



Q) Level order of tree 2

14
18 19
3 4 140
10



Void levelOrder (Node root) {

Queue < Node > q;
q.add (root);

while (q.size () > 0) {

int n = q.size ();

T.C: O(n)

S.C: O(n)

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

Node temp = q.remove();

System.out.print (temp.val);

if (temp.left != null) {

q.add (temp.left);

if (temp.right != null) {

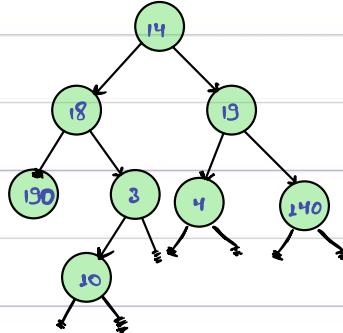
q.add (temp.right);

}
System.out.println();



```

while (q.size() > 0) {
    int n = q.size();
    for (int i = 1; i <= n; i++) {
        Node temp = q.remove();
        System.out.print(temp.val);
        if (temp.left != null) {
            q.add(temp.left);
        }
        if (temp.right != null) {
            q.add(temp.right);
        }
    }
    System.out.println();
}
    
```



q []
~~14~~ ~~18~~ ~~19~~ ~~190~~ ~~3~~ ~~4~~ ~~140~~
~~10~~

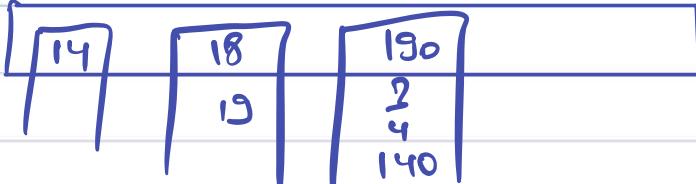
n = 1

14
18 19
190 3 4 140
10



→ List < List < Integer > > ans;

ans:





```
List<List<Integer>> levelOrder (Node root) {
```

```
    Queue < Node > q;
```

```
    q.add (root);
```

```
    List<List<Integer>> ans = new ArrayList<>();
```

```
    while (q.size () > 0) {
```

```
        int n = q.size ();
```

```
        List<Integer> levelAns = new ArrayList<>();
```

```
        for (int i = 1; i <= n; i++) {
```

```
            Node temp = q.remove();
```

```
            levelAns.add (temp.val);
```

```
            if (temp.left != null) {
```

```
                q.add (temp.left);
```

```
            if (temp.right != null) {
```

```
                q.add (temp.right);
```

```
}
```

```
ans.add (levelAns);
```

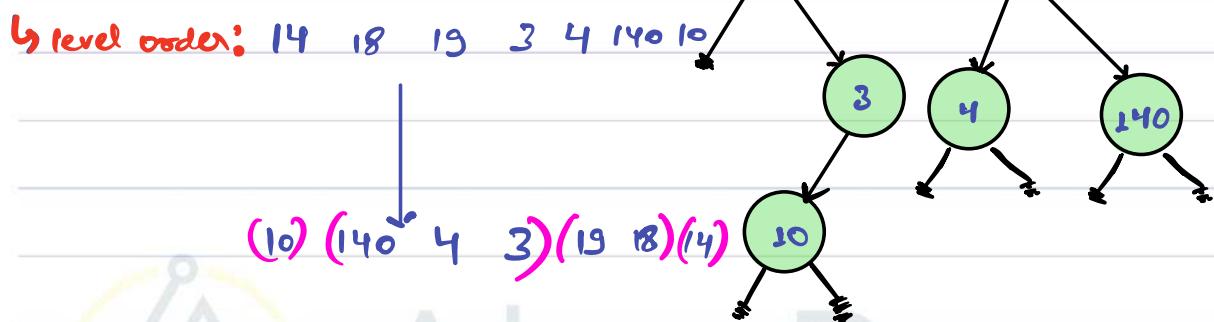
```
return ans;
```



Q) Reverse level order

↳ Point the level order from last level to first.

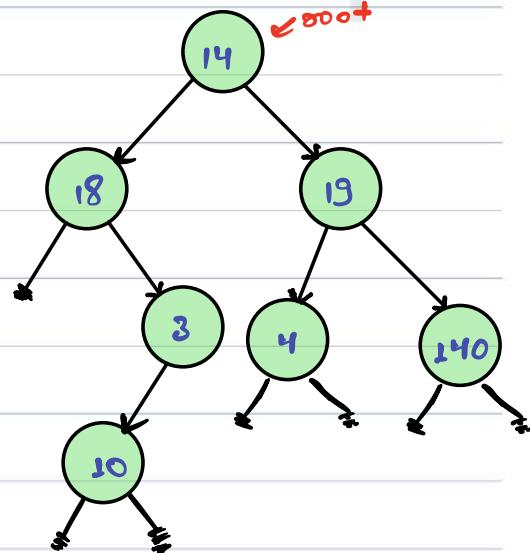
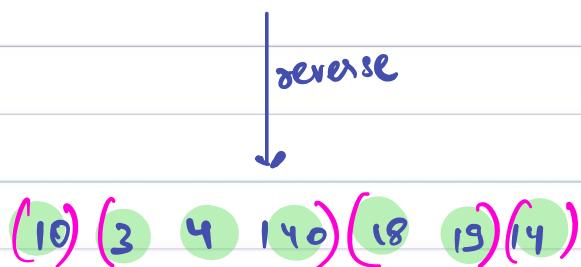
Output: (10) (3 4 140) (18 19) (14)



R-L

level order:

↳ 14 19 18 140 4 3 10





11PSuedo Code

b HW



AlgoPrep

Size of Tree

Java Code:

```
class Tree
{
    public static int getSize(Node root)
    {
        if (root == null)
            return 0;

        int a = getSize(root.left);
        int b = getSize(root.right);

        return a+b+1;
    }
}
```

C++ Code:

```
#include <iostream>

class Node {
public:
    int data;
    Node* left;
    Node* right;

    Node(int value) : data(value), left(nullptr), right(nullptr) {}

};

class Tree {
public:
    static int getSize(Node* root) {
        if (root == nullptr)
            return 0;

        int a = getSize(root->left);
        int b = getSize(root->right);

        return a + b + 1;
    }
}
```

```
};

int main() {
    // You can create a sample binary tree and test the getSize function here
    return 0;
}
```

Python Code:

```
class Node:
    def __init__(self, value):
        self.data = value
        self.left = None
        self.right = None

class Tree:
    @staticmethod
    def get_size(root):
        if root is None:
            return 0

        a = Tree.get_size(root.left)
        b = Tree.get_size(root.right)

        return a + b + 1
```

Sum of Tree

Java Code:

```
class BinaryTree
{
    static int sumBT(Node head){
        if (head == null)
            return 0;

        int a = sumBT(head.left);
        int b = sumBT(head.right);

        return a+b+head.data;
    }
}
```

C++ Code:

```
#include <iostream>

class Node {
public:
    int data;
    Node* left;
    Node* right;

    Node(int value) : data(value), left(nullptr), right(nullptr) {}
};

class BinaryTree {
public:
    static int sumBT(Node* head) {
        if (head == nullptr)
            return 0;
```

```

int a = sumBT(head->left);
int b = sumBT(head->right);

return a + b + head->data;
}

};

int main() {
    // You can create a sample binary tree and test the sumBT function here
    return 0;
}

```

Python Code:

```

class Node:
    def __init__(self, value):
        self.data = value
        self.left = None
        self.right = None

class BinaryTree:
    @staticmethod
    def sumBT(head):
        if head is None:
            return 0

        a = BinaryTree.sumBT(head.left)
        b = BinaryTree.sumBT(head.right)

        return a + b + head.data

```

LevelOrder of Tree

Java Code:

```
Java
class Solution {
    public List<List<Integer>> levelOrder(TreeNode root) {
        Queue<TreeNode> queue = new LinkedList<TreeNode>();
        List<List<Integer>> wrapList = new
        LinkedList<List<Integer>>();

        if(root == null) return wrapList;

        queue.offer(root);
        while(!queue.isEmpty()){
            int size = queue.size();
            List<Integer> subList = new LinkedList<Integer>();
            for(int i=1; i <= size; i++) {
                if(queue.peek().left != null)
                    queue.offer(queue.peek().left);
                if(queue.peek().right != null)
                    queue.offer(queue.peek().right);
                subList.add(queue.poll().val);
            }
            wrapList.add(subList);
        }
        return wrapList;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
#include <queue>

using namespace std;

// Definition for a binary tree node.
struct TreeNode {
    int val;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int x) : val(x), left(NULL), right(NULL) {}
};

class Solution {
public:
    vector<vector<int>> levelOrder(TreeNode* root) {
        vector<vector<int>> wrapList;

        if (root == nullptr) return wrapList;

        queue<TreeNode*> q;
        q.push(root);

        while (!q.empty()) {
            int size = q.size();
            vector<int> subList;

            for (int i = 0; i < size; i++) {
                TreeNode* node = q.front();
                q.pop();

                subList.push_back(node->val);

                if (node->left) q.push(node->left);
                if (node->right) q.push(node->right);
            }

            wrapList.push_back(subList);
        }

        return wrapList;
    };
}
```

Python Code:

```
from collections import deque

# Definition for a binary tree node.
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Solution:
    def levelOrder(self, root: TreeNode) -> List[List[int]]:
        wrapList = []

        if not root:
            return wrapList

        queue = deque()
        queue.append(root)

        while queue:
            size = len(queue)
            subList = []

            for _ in range(size):
                node = queue.popleft()
                subList.append(node.val)

                if node.left:
                    queue.append(node.left)
                if node.right:
                    queue.append(node.right)

            wrapList.append(subList)

    return wrapList
```

Size of Tree

Java Code:

```
class Tree
{
    public ArrayList<Integer> reverseLevelOrder(Node node)
    {
        // code here
        Stack<Node> s = new Stack<Node>();

        Queue<Node> q = new LinkedList<Node>();

        ArrayList<Integer> t = new ArrayList<Integer>();
        if(node != null){
            q.add(node);
            s.push(node);
        }

        while(q.size() > 0){
            Node temp = q.remove();
            //t.Add(temp.data);

            if(temp.right != null){
                q.add(temp.right);
                s.push(temp.right);
            }
            if(temp.left != null){
                q.add(temp.left);
                s.push(temp.left);
            }
        }

        while(s.size()>0){
            Node temp = s.pop();
            t.add(temp.data);
        }

        return t;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
#include <queue>
#include <stack>

using namespace std;

// Definition for a binary tree node.
struct TreeNode {
    int val;
    TreeNode* left;
    TreeNode* right;
    TreeNode(int x) : val(x), left(NULL), right(NULL) {}
};

class Tree {
public:
    vector<int> reverseLevelOrder(TreeNode* node) {
        vector<int> t;
        if (!node)
            return t;

        stack<TreeNode*> s;
        queue<TreeNode*> q;

        q.push(node);
        s.push(node);

        while (!q.empty()) {
            TreeNode* temp = q.front();
            q.pop();

            if (temp->right) {
                q.push(temp->right);
                s.push(temp->right);
            }
            if (temp->left) {
                q.push(temp->left);
                s.push(temp->left);
            }
        }

        while (!s.empty()) {
            TreeNode* temp = s.top();
            s.pop();
            t.push_back(temp->val);
        }
    }
}
```

```

        t.push_back(temp->val);
    }

    return t;
}
};

```

Python Code:

```

from collections import deque

# Definition for a binary tree node.
class TreeNode:
    def __init__(self, val=0, left=None, right=None):
        self.val = val
        self.left = left
        self.right = right

class Tree:
    def reverseLevelOrder(self, node: TreeNode) -> list[int]:
        t = []
        if not node:
            return t

        stack = []
        queue = deque()

        queue.append(node)
        stack.append(node)

        while queue:
            temp = queue.popleft()

            if temp.right:
                queue.append(temp.right)
                stack.append(temp.right)
            if temp.left:
                queue.append(temp.left)
                stack.append(temp.left)

        while stack:
            temp = stack.pop()
            t.append(temp.val)

        return t

```

```
return t
```

Zig zag Tree Traversal_HW

Solution vid:

<https://youtu.be/vWBSRdtUVsQ>

Java Code:

Maximum Level Sum_HW

Solution vid:

<https://youtu.be/fVSkSm5NLP4>

Java Code:

Size of Tree_HW

Solution vid:

<https://youtu.be/96V1RT7CvOU>

Java Code:

SMaximum Depth of Tree_HW

Solution vid:

<https://youtu.be/zFarfV2UFk0>

Java Code:



Today's agenda

↳ Introduction to Heap/PQ.

↳ K smallest element. +1

↳ median of an array. → {Leetcode hard}



AlgoPrep



11 Introduction

`insert(n)`

`getmin()`

`deletemin()`

`ArrayList`

$O(1)$

$O(n)$

$O(n)$

`LinkedList`

$O(n)$

$O(n)$

$O(n)$

`queue`

$O(1)$

$O(n)$

$O(n)$

`HashMap`

$O(1)$

$O(n)$

$O(n)$

`PQ`

$O(\log n)$

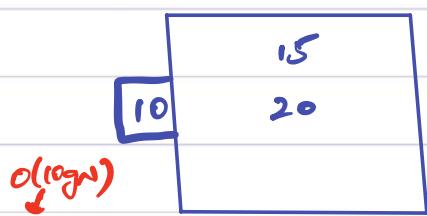
↑
No. of elements
in PQ

$O(1)$

$O(\log n)$

↳ `PriorityQueue< Integer > PQ = new`

min
`PQ`



`add(10)`

`add(20)`

`add(5)`

`add(15)`

`remove() → 5`

`min() → 10`

`PriorityQueue< Integer > PQ = new`

`PriorityQueue< T >;`
Collections.reverseOrder()

`PriorityQueue< T >;`

$\rightarrow PQ.add(n); \rightarrow O(\log n)$

$\rightarrow PQ.remove(); \rightarrow O(\log n)$

$\rightarrow PQ.peek(); \rightarrow O(1)$

$\rightarrow PQ.size(); \rightarrow O(n)$



Q) Kth Smallest Element

↳ Given N distinct elements, Point K Smallest elements.

Ex: $\text{arr}[10] = \{ \begin{smallmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 8 & 3 & 10 & 4 & 11 & 2 & 7 & 6 & 14 & 1 \end{smallmatrix} \}$

$K=4: 1 \ 2 \ 3 \ 4$

$\text{arr}[3] = \{ \begin{smallmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ -3 & 6 & 2 & 0 & 8 & 7 & 10 & 4 \end{smallmatrix} \}$

$K=3: -3 \ 0 \ 2$

1/Idea1

↳ Sort the array and return the first K elements.

T.C: $O(N \log N)$

1/Idea2

↳ Add all elements to min PQ and get the first K elements.

T.C: $O(N \log N)$



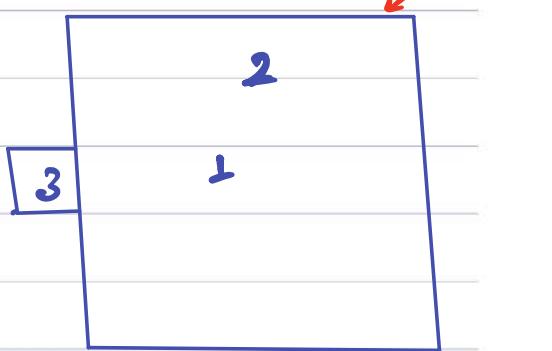
IIideas

→ K Smallest elements

$arr[10]: \{ 8 \ 3 \ 10 \ 4 \ 12 \ 2 \ 7 \ 6 \ 14 \ 1 \}$

$K=3$

4 1 2 3



IIIPseudo code

void KthSmallest (int arr[N], int K){
 maxHeap < int > mh;

Smallest element
+
weakest performer

for (int i=0; i<K; i++) {
 mh.add(arr[i]);
}

T.C: $O(N \times \log K)$

S.C: $O(K)$

for (int i=K; i<N; i++) {
 if (arr[i] < mh.peek()) {
 mh.remove();
 mh.add(arr[i]);
 }
}

else { continue; }

while (mh.size() > 0) {

S.O.P (mh.remove());

or return

mh.peek();

3

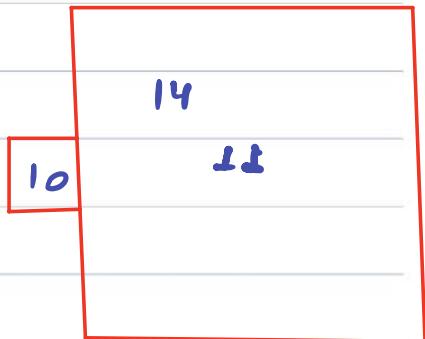


Q) K largest elements

$\text{arr}[10]: \{ 8, 3, 10, 4, 12, 2, 7, 6, 14, 1 \}$

K: 3

✓ min heap



AlgoPrep



11 median

↳ middle element of sorted array.

$$\text{arr}[3] = \{2, 5, 3\}$$

↳ {2, 3, 5} → 3

$$\text{arr}[5] = \{4, 3, 6, 8, 5\}$$

↳ {3, 4, 5, 6, 8} → 5

$$\text{arr}[6] = \{4, 3, 9, 5, 12, 2\}$$

↳ {2, 3, 4, 5, 9, 12} → \frac{4+5}{2} = 4.5

$$\text{arr}[4] = \{4, 6, 10, 14\}$$

↳ \frac{6+10}{2} = \frac{16}{2} = 8

Break till 9:28 PM

LeetCode 295



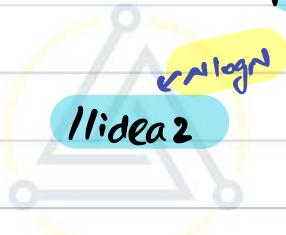
Q) Point median after each insertion.

arr[5]: 9 6 3 10 4
↓ ↑ ↓ ↑ ↓
9 7.5 6 7.5 6

Idea 1

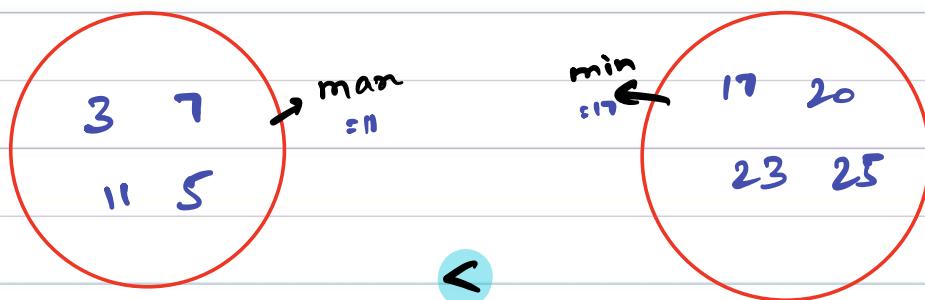
↳ After each insertion, sort the array & return the middle one.

T.C: $N \times N \log N \approx O(N^2 \log N)$



Idea 2

3 5 11 23 20 25 17 7



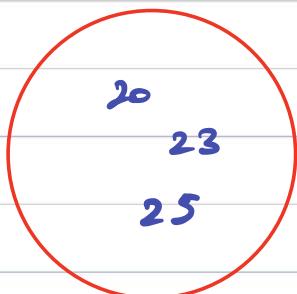
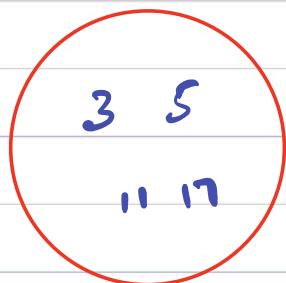
* if total elements are even:

↳ median = $\frac{\text{man of left bucket} + \text{min of right bucket}}{2}$

odd

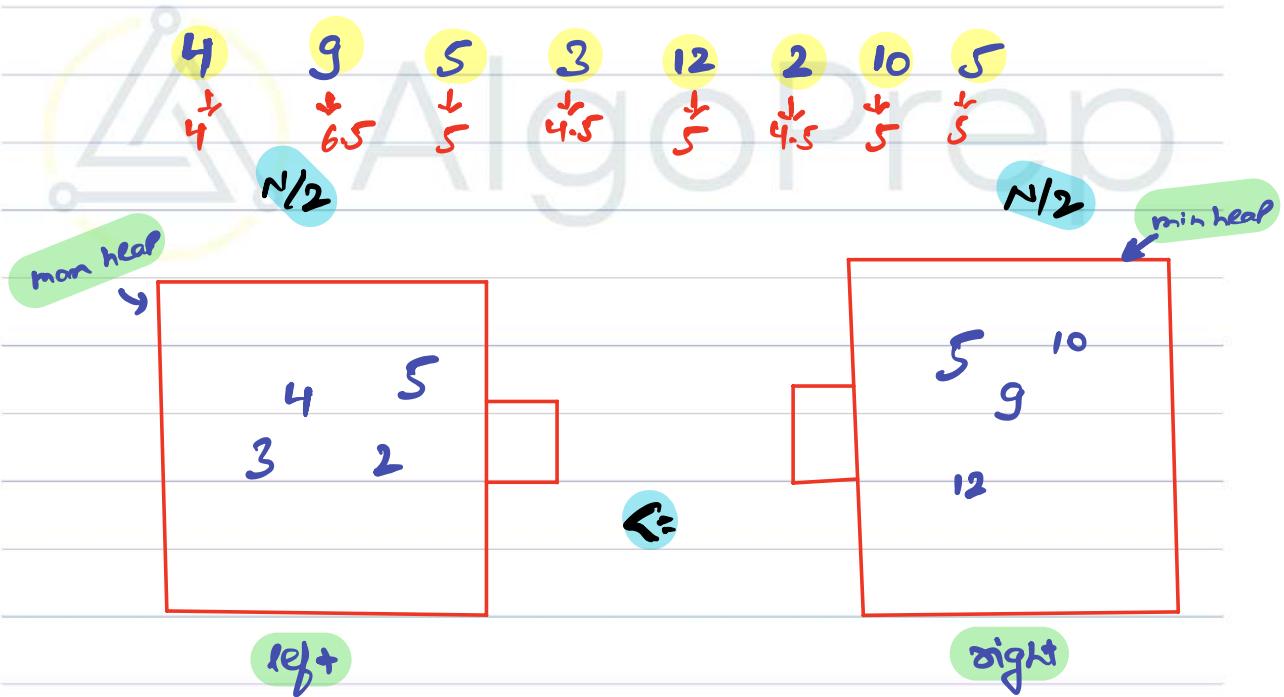


3 5 11 23 20 25 17



if total elements are odd:

↳ median = man of left bucket





if (`left.size() == right.size()`)

↳ ultimately new element should be added to `left` PQ but to maintain inequality add it via `right` PQ.

if (`left.size() != right.size()`)

↳ ultimately new element should be added to `right` PQ but to maintain inequality add it via `left` PQ.

IP Sudo Code

Class `MedianFinder` {

 maxHeap < Integer > left;

 minHeap < Integer > right;

 Public `medianFinder()` {

 3

ToC: $O(n * \log n)$

S.C: $O(n)$

 Public void `addNum(int num)` {

 if (`left.size() == right.size()`) {

 right.add(num);

 left.add(right.remove());

 3

 else {

 left.add(num);

 right.add(left.remove());



```
public double findMedian() {  
    if (left.size() == right.size()) {  
        double ans = (left.peek() + right.peek()) / 2.0;  
        return ans;  
    } else {  
        return left.peek() * 1.0;  
    }  
}
```



AlgoPrep

K Smallest Element

Java Code:

```
Java
class Solution {
    public int findKthLargest(int[] nums, int k) {
        PriorityQueue<Integer> pq = new PriorityQueue<>();

        for(int i=0;i<k;i++){
            pq.add(nums[i]);
        }

        for(int i=k;i<nums.length;i++){
            if(pq.peek()< nums[i]){
                pq.remove();
                pq.add(nums[i]);
            }
        }

        return pq.peek();
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
#include <queue>

using namespace std;

class Solution {
public:
    int findKthLargest(vector<int>& nums, int k) {
        priority_queue<int, vector<int>, greater<int> pq;

        for (int i = 0; i < k; i++) {
```

```

        pq.push(nums[i]);
    }

    for (int i = k; i < nums.size(); i++) {
        if (pq.top() < nums[i]) {
            pq.pop();
            pq.push(nums[i]);
        }
    }

    return pq.top();
}
};

```

Python Code:

```

import heapq

class Solution:
    def findKthLargest(self, nums, k):
        min_heap = nums[:k]
        heapq.heapify(min_heap)

        for i in range(k, len(nums)):
            if nums[i] > min_heap[0]:
                heapq.heappop(min_heap)
                heapq.heappush(min_heap, nums[i])

        return min_heap[0]

```

K Largest Element

Java Code:

```

class Solution{
    public static int kthSmallest(int[] arr, int l, int r, int k)
    {
        PriorityQueue<Integer> pq = new PriorityQueue<>(Collections.reverseOrder());
        for(int i=0;i<k;i++){

```

```

        pq.add(arr[i]);
    }

    for(int i=k;i<arr.length;i++){
        if(pq.peek() > arr[i]){
            pq.remove();
            pq.add(arr[i]);
        }
    }

    return pq.peek();
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>
#include <functional>

using namespace std;

class Solution {
public:
    int kthSmallest(vector<int>& arr, int l, int r, int k) {
        priority_queue<int, vector<int>, greater<int>> pq;

        for (int i = 0; i < k; i++) {
            pq.push(arr[i]);
        }

        for (int i = k; i < arr.size(); i++) {
            if (pq.top() < arr[i]) {
                pq.pop();
                pq.push(arr[i]);
            }
        }

        return pq.top();
    }
};

```

Python Code:

```
import heapq

class Solution:
    def kthSmallest(self, arr, l, r, k):
        max_heap = arr[:k]
        heapq.heapify(max_heap)

        for i in range(k, len(arr)):
            if max_heap[0] < arr[i]:
                heapq.heappop(max_heap)
                heapq.heappush(max_heap, arr[i])

        return max_heap[0]
```

Find Median from a data Stream

Java Code:

```
Java
class MedianFinder {
    PriorityQueue<Integer> small;
    PriorityQueue<Integer> large;
    public MedianFinder() {
        small = new PriorityQueue<>(Collections.reverseOrder());
        large = new PriorityQueue<>();
    }

    public void addNum(int num) {
        if(small.size() == large.size()){
            large.add(num);
            small.add(large.remove());
        }else{
            small.add(num);
            large.add(small.remove());
        }
    }
}
```

```

public double findMedian() {
    if(small.size() == large.size()){
        return (small.peek()+large.peek())/2.0;
    }else{
        return small.peek();
    }
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>

using namespace std;

class MedianFinder {
    priority_queue<int> small;
    priority_queue<int, vector<int>, greater<int>> large;

public:
    MedianFinder() {
        small = priority_queue<int>();
        large = priority_queue<int, vector<int>, greater<int>>();
    }

    void addNum(int num) {
        if (small.size() == large.size()) {
            large.push(num);
            small.push(large.top());
            large.pop();
        } else {
            small.push(num);
            large.push(small.top());
            small.pop();
        }
    }

    double findMedian() {

```

```

        if (small.size() == large.size()) {
            return (small.top() + large.top()) / 2.0;
        } else {
            return small.top();
        }
    }
};

```

Python Code:

```

import heapq

class MedianFinder:
    def __init__(self):
        self.small = [] # max-heap for smaller half
        self.large = [] # min-heap for larger half

    def addNum(self, num: int) -> None:
        if len(self.small) == len(self.large):
            # Add to the larger half (min-heap) first
            heapq.heappush(self.large, num)
            # Move the smallest element from the larger half to the smaller half
            smallest_large = heapq.heappop(self.large)
            heapq.heappush(self.small, -smallest_large)
        else:
            # Add to the smaller half (max-heap) first
            heapq.heappush(self.small, -num)
            # Move the largest element from the smaller half to the larger half
            largest_small = -heapq.heappop(self.small)
            heapq.heappush(self.large, largest_small)

    def findMedian(self) -> float:
        if len(self.small) == len(self.large):
            return (self.large[0] - self.small[0]) / 2.0
        else:
            return float(self.small[0])

```

Last Stone Weight_HW

Solution Vid:

<https://youtu.be/LOQ6g48fXsE>

Java Code:

```
class Solution {  
    public int lastStoneWeight(int[] stones) {  
        PriorityQueue<Integer> pq = new PriorityQueue<>(Collections.reverseOrder());  
        for(int ele : stones) pq.add(ele);  
  
        while(pq.size()>1){  
            int aa = pq.remove();  
            aa = aa - pq.remove();  
            if(aa != 0) pq.add(aa);  
  
        }  
        if(pq.size()==0) return 0;  
        else return pq.remove();  
    }  
}
```

Minimize the Sum_HW

Solution Vid:

<https://youtu.be/XKXFyV07ysY>

Java Code:

```
class Solution {  
    long minimizeSum(int N, int arr[]) {  
  
        PriorityQueue<Long> pq = new PriorityQueue<>();  
        for(int i = 0; i<N; i++){  
            pq.add((long)arr[i]);  
        }  
        long ans = 0;  
        while(pq.size()>1){  
            long a = pq.remove();  
            long b = pq.remove();  
            ans = ans + a + b;  
            pq.add(a+b);  
        }  
        return ans;  
    }  
}
```



Today's agenda

- ↳ Dynamic Programming Intro
- ↳ when to use DP
- ↳ steps for DP
- ↳ # of stairs
- ↳ Sqr

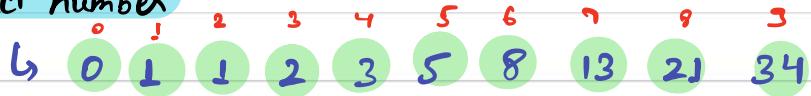


AlgoPrep

→ DRY → Don't Repeat yourself



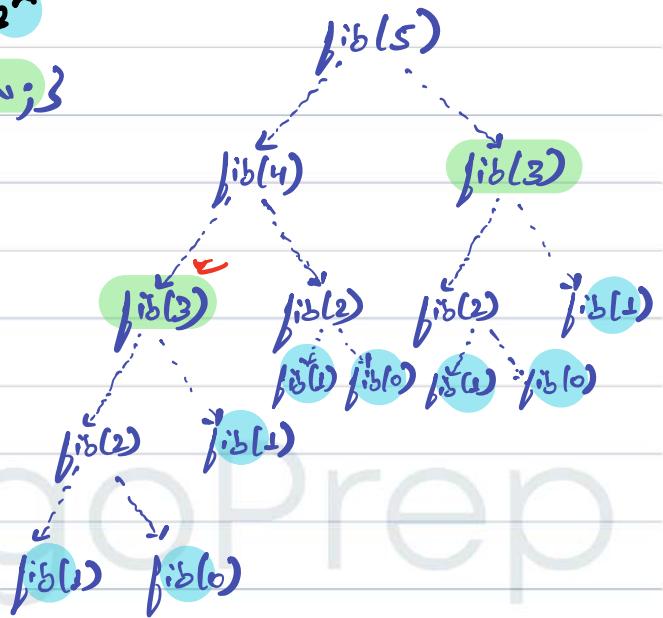
Q) Nth fibonacci number



```
int fib (int n){  
    T.C:  $2^n$   
    if (n==0 || n==1) {return n;}  
}
```

```
int a = fib (n-1);  
int b = fib (n-2);
```

return $a+b$;





int $dp[n+1] = \{ -1 \};$

dp	0	1	2	3	4	5
	-1	-1	-1	-1	-1	-1

$\rightarrow 5$

int fib (int n) {

1 if ($n == 0 || n == 1$) { return n; }

2 if ($dp[n] != -1$) { return dp[n]; }

3 int a = fib (n-1);

4 int b = fib (n-2);

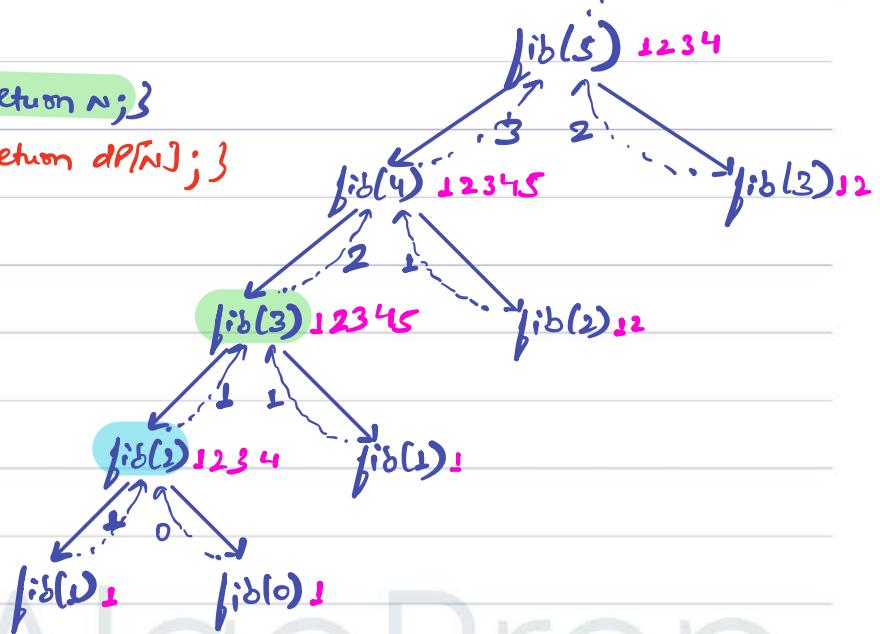
5 $dp[n] = a+b;$

6 return a+b;

}

T.C: $O(n)$

S.C: $O(n)$



\rightarrow Optimal Substructure

\hookrightarrow you can divide the problem into subproblems.

\rightarrow Overlapping Subproblems

\hookrightarrow Same Problem repeating



11 quick questions

mumbai goa delhi

$$\hookrightarrow 3 \times 2 = 6$$

mumbai goa delhi

$$\rightarrow 3 \times 2 = 6$$

$$\Rightarrow 6 + 9 = 15$$

bangalore

$$\hookrightarrow 3 \times 3 = 9$$



AlgoPrep



// N Stair

↳ Given N , how many ways we can go from 0 - N th stair.

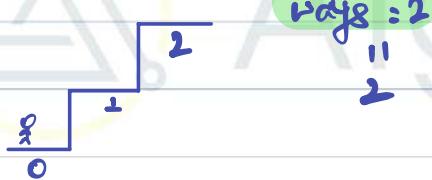
Note: you can take steps of length 1 or 2.

$N=1$



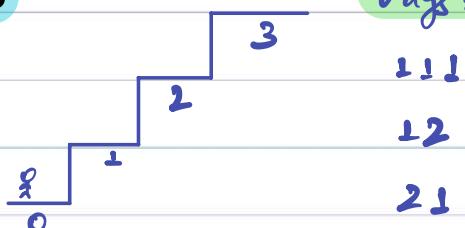
ways = 1

$N=2$



ways = 2
11
12
21

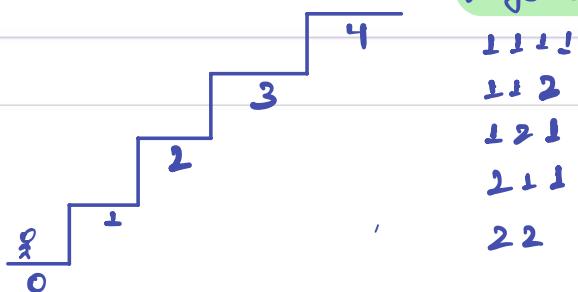
$N=3$



ways = 3

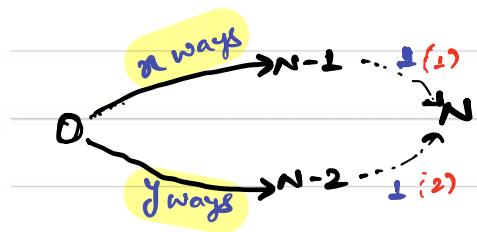
111
12
21

$N=4$



ways = 5

1111
112
121
211
22



$$n \cdot 1 + y \cdot 1 \Rightarrow x+y$$

No. of ways(n) = No. of ways($n-1$) + No. of ways($n-2$)

$$n=1 \rightarrow 1$$

$$n=2 \rightarrow 2$$

int $dp[n+1] = \{-1\}$;

int Stairs(int n) {

 1 if ($n == 1 || n == 2$) return n;

 2 if ($dp[n] != -1$) return dp[n];

 3 int a = Stairs($n-1$);

 4 int b = Stairs($n-2$);

 5 $dp[n] = a+b$;

 6 return a+b;

}

Break till 9:25 PM



11 Steps for DP

① optimal substructure

② overlapping subproblem

① DP state : what are we solving $\rightarrow f(i)$

② recurrence relation

\hookrightarrow relation betⁿ problem and subproblem.

③ DP table

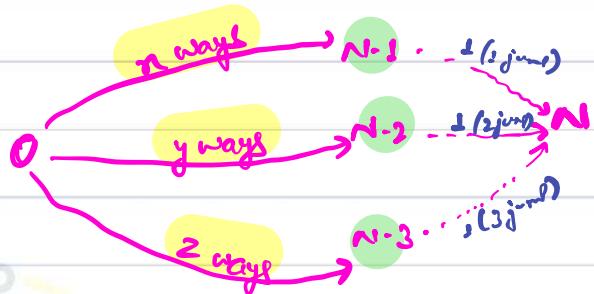
\hookrightarrow where we store the answer.



"N Stairs

↳ Given N , how many ways we can go from $0 - N^{\text{th}}$ stair.

Note: you can take steps of length 1 or 2 or 3



Golden rule of recursion:

↳ No. of Calls = No. of Choices.



Q) Find minimum number of perfect squares required to sum = N.

N=6

$$\hookrightarrow 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 = 6$$

$$1^2 + 1^2 + 1^2 = 3$$

N=10

$$\hookrightarrow 1^2 + 1^2 + \dots + 3^2 = 10$$

$$\hookrightarrow 2^2 + 2^2 + 1^2 + 1^2 = 4$$

$$\hookrightarrow 3^2 + 1^2 = 2$$

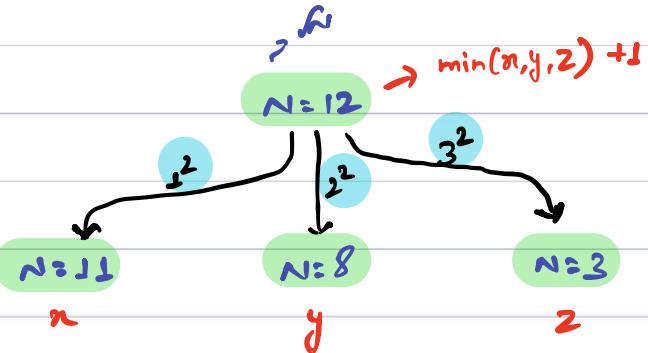
N=9

$$\hookrightarrow 3^2 = 1$$

N=12

$$\hookrightarrow 3^2 + 1^2 + 1^2 + 1^2 = 4$$

$$\hookrightarrow 2^2 + 2^2 + 2^2 = 3$$



```

int minSq (int N) {
    if (N==0 || N==1) {return N; }

    int Smallest = +∞;
    for (int i=1; i*i <= N; i++) {
        int temp = minSq (N - i*i);
        Smallest = min (smallest, temp);
    }
    return Smallest + 1;
}

```

```
int minSg ( int N ) {  
    1 if ( N == 0 || N == 1 ) { return N; }  
}
```

```
    2 int Smallest = +100;  
    for ( int i = 1; i * i < N; i++ ) {  
        int temp = minSg ( N - i * i );  
        Smallest = min ( Smallest, temp );  
    }  
    3 return Smallest + 1;  
}
```

$$\text{int } \text{dp}[N+1] = \{-1\};$$

```
int minSg ( int N , int [ ] dp ) {  
    if ( N == 0 || N == 1 ) { return N; }  
    if ( dp[N] != -1 ) { return dp[N]; }  
}
```

T.C:

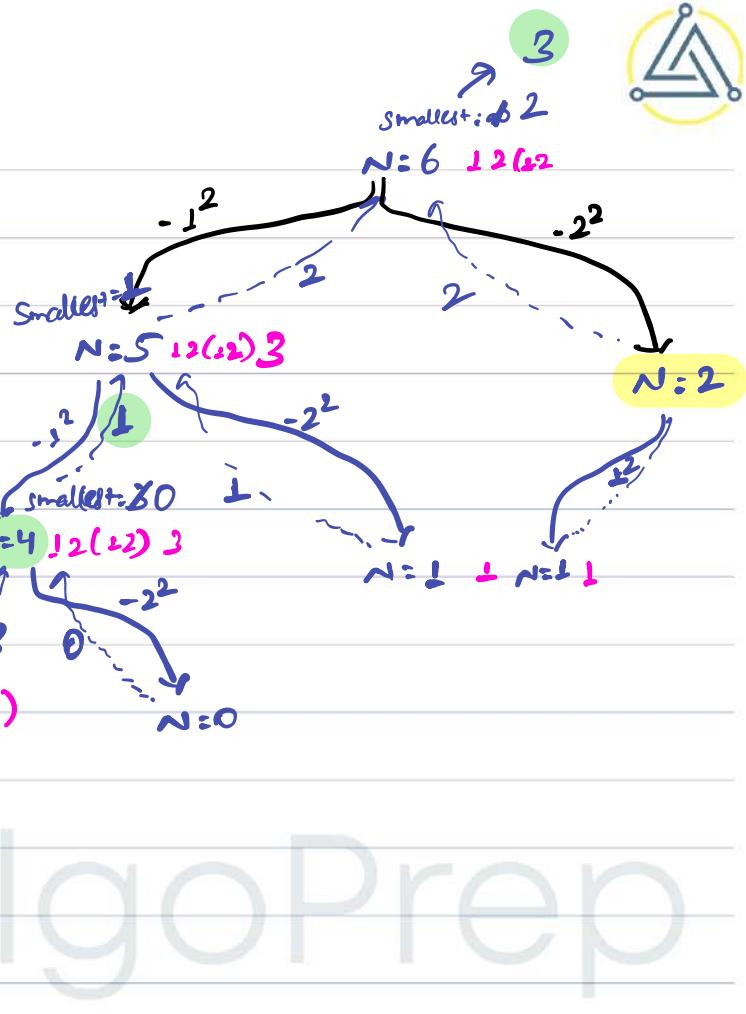
S.C:

```
    2 int Smallest = +100;  
    for ( int i = 1; i * i < N; i++ ) {  
        int temp = minSg ( N - i * i , dp );  
        Smallest = min ( Smallest, temp );  
    }  
    3
```

$$dp[N] = Smallest + 1;$$

```
return Smallest + 1;
```

3





```
int Smallest = +oo;  
for (int i=1; i*i <= N; i++) {  
    int temp = minsg(N - i*i);  
    Smallest = min(Smallest, temp);  
}
```

3



Algorithm

```
int Smallest = +oo;  
int temp1 = minsg(N - 12);  
Smallest = math.min(Smallest, temp1);  
int temp2 = minsg(N - 22);  
Smallest = math.min(Smallest, temp2);  
int temp3 = minsg(N - 32);  
Smallest = math.min(Smallest, temp3);  
return Smallest + 1;
```

Nth Fibonacci

Java Code:

```
Java
class Solution {
    public int fib(int n) {
        int[] dp = new int[n+1];
        Arrays.fill(dp,-1);
        int ans = fibhelper(n,dp);
        return ans;
    }

    public int fibhelper(int n,int[]dp) {
        if(n== 0 || n == 1){
            return n;
        }
        if(dp[n]!= -1){
            return dp[n];
        }
        int a = fibhelper(n-1,dp);
        int b = fibhelper(n-2,dp);

        dp[n] = a+b;
        return a+b;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>

class Solution {
public:
    int fib(int n) {
```

```

std::vector<int> dp(n + 1, -1);
int ans = fibHelper(n, dp);
return ans;
}

int fibHelper(int n, std::vector<int>& dp) {
    if (n == 0 || n == 1) {
        return n;
    }
    if (dp[n] != -1) {
        return dp[n];
    }
    int a = fibHelper(n - 1, dp);
    int b = fibHelper(n - 2, dp);

    dp[n] = a + b;
    return a + b;
}
};

int main() {
    Solution solution;
    int n;
    std::cin >> n;

    int result = solution.fib(n);
    std::cout << result << std::endl;

    return 0;
}

```

Python Code:

```

class Solution:
    def __init__(self):
        self.dp = []

    def fib(self, n: int) -> int:
        self.dp = [-1] * (n + 1)
        ans = self.fibHelper(n)
        return ans

    def fibHelper(self, n: int) -> int:
        if n == 0 or n == 1:
            return n

```

```

if self.dp[n] != -1:
    return self.dp[n]
a = self.fibHelper(n - 1)
b = self.fibHelper(n - 2)
self.dp[n] = a + b
return a + b

# Example usage:
# You can create an instance of Solution and call the fib method with the desired value of n.
solution = Solution()
n = int(input())
result = solution.fib(n)
print(result)

```

Climbing stairs

Java Code:

```

Java
class Solution {

    public int climbStairs(int n){

        int[] dp = new int[n + 1];
        Arrays.fill(dp , -1);
        return climbStairsHelper(n , dp);
    }

    public int climbStairsHelper(int n , int[] dp) {

        if(n == 1 || n == 2){
            return n;
        }

        if(dp[n] != -1){
            return dp[n];
        }
    }
}

```

```

        int step1 = climbStairsHelper(n - 1, dp);
        int step2 = climbStairsHelper(n - 2, dp);

        dp[n] = step1 + step2;

        return dp[n];
    }
}

```

C++ Code:

```

#include <iostream>
#include <vector>

class Solution {
public:
    int climbStairs(int n) {
        std::vector<int> dp(n + 1, -1);
        return climbStairsHelper(n, dp);
    }

    int climbStairsHelper(int n, std::vector<int>& dp) {
        if (n == 1 || n == 2) {
            return n;
        }

        if (dp[n] != -1) {
            return dp[n];
        }

        int step1 = climbStairsHelper(n - 1, dp);
        int step2 = climbStairsHelper(n - 2, dp);

        dp[n] = step1 + step2;

        return dp[n];
    }
};

int main() {
    Solution solution;

```

```

int n;
std::cin >> n;

int result = solution.climbStairs(n);
std::cout << result << std::endl;

return 0;
}

```

Python Code:

```

class Solution:
    def __init__(self):
        self.dp = []

    def climbStairs(self, n: int) -> int:
        self.dp = [-1] * (n + 1)
        return self.climbStairsHelper(n)

    def climbStairsHelper(self, n: int) -> int:
        if n == 1 or n == 2:
            return n

        if self.dp[n] != -1:
            return self.dp[n]

        step1 = self.climbStairsHelper(n - 1)
        step2 = self.climbStairsHelper(n - 2)

        self.dp[n] = step1 + step2

        return self.dp[n]

# Example usage:
# You can create an instance of Solution and call the climbStairs method with the desired value
# of n.
solution = Solution()
n = int(input())
result = solution.climbStairs(n)
print(result)

```

Perfect Squares

Java Code:

```
Java
class Solution {
    public int numSquares(int n) {
        int[] dp = new int[n + 1];
        Arrays.fill(dp, -1);
        return helper(n, dp);
    }

    public int helper(int n, int[] dp) {
        if (n == 0 || n == 1){
            return n;
        }

        if (dp[n] != -1)
            return dp[n];

        int smallest = Integer.MAX_VALUE;

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

            int temp = helper(n - i*i, dp);
            smallest = Math.min(smallest, temp);
        }

        dp[n] = smallest + 1;
        return smallest + 1;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
```

```

class Solution {
public:
    int numSquares(int n) {
        std::vector<int> dp(n + 1, -1);
        return helper(n, dp);
    }

    int helper(int n, std::vector<int>& dp) {
        if (n == 0 || n == 1) {
            return n;
        }

        if (dp[n] != -1) {
            return dp[n];
        }

        int smallest = INT_MAX;

        for (int i = 1; i * i <= n; i++) {
            int temp = helper(n - i * i, dp);
            smallest = std::min(smallest, temp);
        }

        dp[n] = smallest + 1;
        return smallest + 1;
    };
}

int main() {
    Solution solution;
    int n;
    std::cin >> n;

    int result = solution.numSquares(n);
    std::cout << result << std::endl;

    return 0;
}

```

Python Code:

```
class Solution:
    def numSquares(self, n: int) -> int:
        dp = [-1] * (n + 1)
        return self.helper(n, dp)

    def helper(self, n: int, dp: List[int]) -> int:
        if n == 0 or n == 1:
            return n

        if dp[n] != -1:
            return dp[n]

        smallest = float('inf')

        for i in range(1, int(n**0.5) + 1):
            temp = self.helper(n - i * i, dp)
            smallest = min(smallest, temp)

        dp[n] = smallest + 1
        return smallest + 1

# Example usage:
# You can create an instance of Solution and call the numSquares method with the desired
# value of n.
solution = Solution()
n = int(input())
result = solution.numSquares(n)
print(result)
```



Today's agenda

- ↳ unique Paths in grid +1
- ↳ minimum Path sum
- ↳ Cherry Pickup

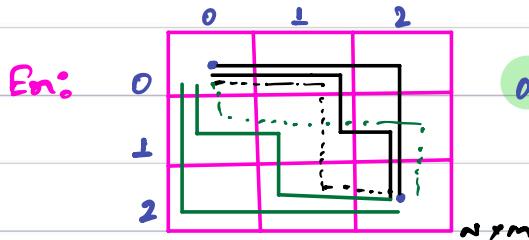


AlgoPrep



Q) Number of ways to go from $(0,0) \rightarrow (n-1, m-1)$ cell.

Movement allowed: cell $\xrightarrow[1 \text{ step}]{\text{right}}$ or $\downarrow[1 \text{ step}]{\text{bottom}}$



RRDD

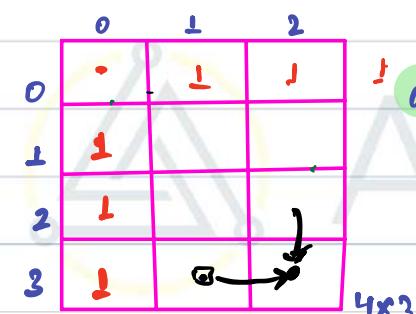
DDRR

RDRD

DRDR

RDDR

DRRD



$x+y$

PathCount $(0,0 \rightarrow 3,2)$

PathCount $(0,0 \rightarrow 2,2)$

$i-j, j$

$i-1, j$

y

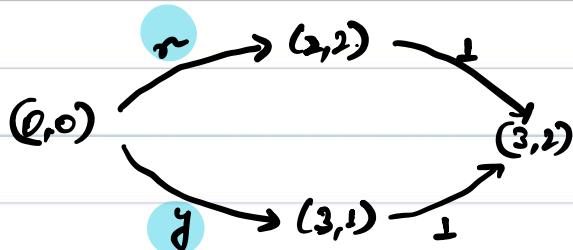
$i, j-1$

PathCount $(0,0 \rightarrow 3,1)$

PathCount $(0,0 \rightarrow 2,1)$

x

$i-1, j$



$$\text{total ways: } x+1 + y*1 \\ = x+y$$



// Pseudo code

```
int DP[n][m] = -1; // Initialize DP matrix to -1
```

```
int PathCount (int i, int j) {
    if (i == 0 || j == 0) return 1;
    if (DP[i][j] != -1) return DP[i][j];
}
```

T.C: $O(n \times m)$

S.C: $O(n \times m)$

+ StackSize

```
int x = PathCount (i-1, j);
```

```
int y = PathCount (i, j-1);
```

$DP[i][j] = x+y;$

return $x+y;$

3

$\text{if } (i < 0 \text{ || } j < 0) \{ \text{return } 0; \}$

$\text{if } (i == 0 \text{ || } j == 0) \{ \text{return } 1; \} \approx \text{if } (i == 0 \text{ && } j == 0) \{ \text{return } 1; \}$

\downarrow
 $PC(0,2) \rightarrow 1$

0	1	0	2	0
0	-	-	-	-
1	-	-	-	-
2	-	-	-	-

4x3

$PC(0,2)$

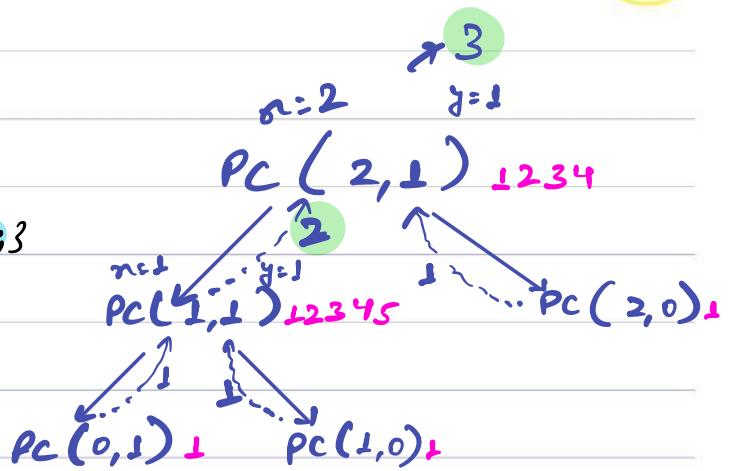
\swarrow $PC(-1,2)$ \searrow $PC(0,1)$

\swarrow $PC(-1,1)$ \searrow $PC(0,0)$



```

int dp[n][m] = {-1};
int PathCount ( int i, int j ) {
    1 if ( i == 0 || j == 0 ) { return 1; }
    2 if ( dp[i][j] != -1 ) { return dp[i][j]; }
    3 int n = PathCount ( i-1, j );
    4 int y = PathCount ( i, j-1 );
    5 dp[i][j] = n+y;
    6 return n+y;
}
  
```



dp

	0	1
0	-1	-1
1	-1	2
2	-1	13

3x2



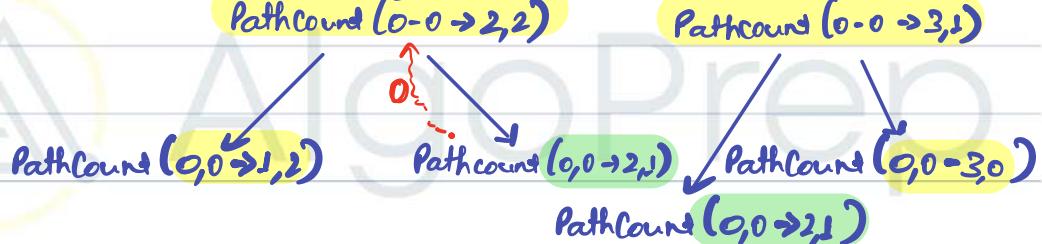
Q) Number of ways to go from $(0,0) \rightarrow (n-1, m-1)$ cell.

Movement allowed: cell $\xrightarrow[\text{right}]{\text{1 step}}$ or $\downarrow \xrightarrow{\text{bottom}} \text{1 step}$

$\hookrightarrow \text{mat}[i][j] := 0$ blocked cell, $\text{mat}[i][j] := 1$ unblocked cell
 \hookrightarrow Path Can't go via blocked cell

	0	1	2
0	1	0	1
1	1	1	1
2	1	0	1
3	1	1	1

PathCount $(0,0 \rightarrow 3,2)$



if $(i == 0 \text{ || } j == 0)$ {return 0;}
 if $(i == n \text{ & } j == m)$ {return 1;} \approx if $(i < 0 \text{ || } j < 0)$ {return 0;}
 if $(i == n \text{ & } j == m)$ {return 1;}

$PC(0,2) \rightarrow 1$

	0	1	2
0	1	0	1
1	1	1	1
2	1	1	1
3	1	1	1

$PC(0,2) \rightarrow 0$

$PC(-1,2) \rightarrow 0$

$PC(0,1) \rightarrow 0$



//Pseudo code

```
int DP[n][m] = {-1};  
int PathCount (int mat[][], int i, int j) {  
    if (i < 0 || j < 0) { return 0; }  
    if (i == 0 && j == 0) { return 1; }  
    if (mat[i][j] == 0) { return 0; }  
    if (DP[i][j] != -1) { return DP[i][j]; }  
    int x = PathCount (i-1, j);  
    int y = PathCount (i, j-1);  
}
```

DP[i][j] = x+y;
return x+y;

3



(Q) Minimum Path Sum

Given a 2d array, filled with non-negative numbers find a Path from $(0,0) \rightarrow (n-1, m-1)$ which minimizes the total cost Path.

Movement allowed: cell $\xrightarrow[right]{1\ step}$ or $\downarrow\limits_{bottom}^{1\ step}$

Ex:	0	1	2
	0	2	3
	1	1	6
	2	9	4

$\rightarrow \text{ans} = 18$

//greedy idea \rightarrow wrong idea

2	1	100
10	100	100
1	1	1

$$\rightarrow \min(x, y) + \text{mat}[3][2]$$

//idea

0	1	2
0	2	3
1	1	6
2	9	7

min Path $(0,0 \rightarrow 3,2)$

min Path $(0,0 \rightarrow 2,2)$

min Path $(0,0 \rightarrow 3,1)$

PathCount $(0,0 \rightarrow 1,2)$

PathCount $(0,0 \rightarrow 2,1)$

PathCount $(0,0 \rightarrow 2,0)$

PathCount $(0,0 \rightarrow 3,0)$



11/Pseudo code

```
int dP[N][M]:= -1;
```

```
int minCostPath (int mat[][], int i, int j) {  
    if (i < 0 || j < 0) { return Integer.MAX_VALUE; }  
    if (i == 0 & j == 0) { return mat[0][0]; }  
    if (dP[i][j] != -1) { return dP[i][j]; }
```

T.C: $O(n*m)$

S.C: $O(n*m)$

* Stack
Space

```
int x = minCostPath (mat, i-1, j);
```

```
int y = minCostPath (mat, i, j-1);
```

$dP[i][j] = \min(x, y) + mat[i][j],$

return $\min(x, y) + mat[i][j],$

}



int $dP[i][j] = \min\{dP[i-1][j], dP[i][j-1]\} + mat[i][j]$

int minCostPath (int mat[10][10], int i, int j) {

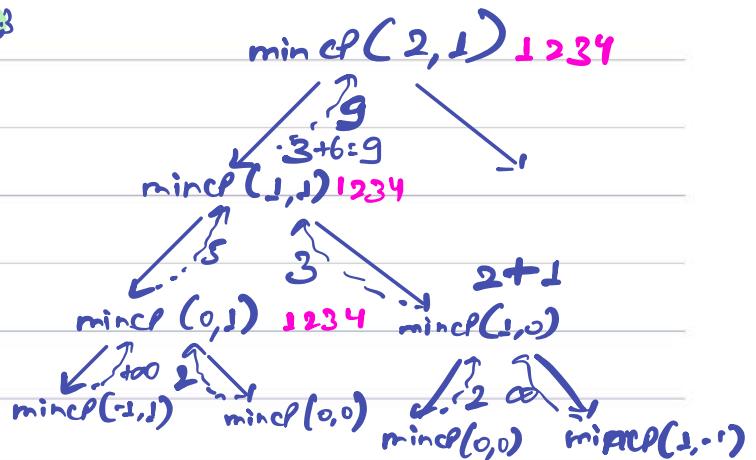
- 1 if ($i < 0$) || ($j < 0$) return Integer.MAX_VALUE;
- 2 if ($i == 0$) & ($j == 0$) return mat[0][0];
- 3 if ($dP[i][j] != -1$) return $dP[i][j]$;

4 int x = mincostPath (mat, i-1, j);

5 int y = mincostPath (mat, i, j-1);

6 $dP[i][j] = \min\{x, y\} + mat[i][j]$;

7 return $\min\{x, y\} + mat[i][j]$;



$$\begin{array}{r} \text{mat} \\ \begin{array}{r} 0 \\ 0 \\ 2 \\ \dots \\ 1 \\ 2 \\ \hline 1 \\ 0 \\ \end{array} \\ \begin{array}{r} 0 \\ 2 \\ 3 \\ \dots \\ 6 \\ \hline 2 \\ 9 \\ \end{array} \end{array} \quad \begin{array}{r} \text{dp} \\ \begin{array}{r} 0 \\ 0 \\ -1 \\ \dots \\ 3 \\ \hline 1 \\ 5 \\ \end{array} \\ \begin{array}{r} 0 \\ -1 \\ -1 \\ \dots \\ 1 \\ 9 \\ \hline -1 \\ 1 \end{array} \end{array}$$

Break till 9:50 PM



Q) Cherry Pickup

Given 2D array, each cell contains 0, 1, -1.

→ 0 means cell is empty

→ 1 means cell is having a cherry

→ -1 means blockade

→ Return the maximum number of cherries you can collect

by moving right or down and doing a full cycle.

↳ while coming back up or left

(0,0 → n-1, m-1)

8

(n-1, m-1 → 0, 0)

Ex: 0 0 1 -1

1 1 0 -1

2 1 1 1 → Ans=5

→ incorrect idea

II Max Path Sum

1 2 3 4 5 6

Ex: 0 1 1 1 1 0 0

1 0 0 0 1 0 1

2 1 0 0 1 0 0

3 0 0 0 1 0 0

4 0 0 0 1 1 1



II Correct idea

$(0,0)$	1	2	3	4	5	6	
En:	0	1	1	1	1	0	0
	1	0	0	0	1	0	1
	2	1	0	0	1	0	0
	3	0	0	0	1	0	0
	4	0	0	0	1	1	1
							$(n-1, m-1)$
							\rightarrow

\uparrow

\rightarrow

\downarrow

$P_1 \xrightarrow{\text{row1}} \text{col1}$

$P_2 \xrightarrow{\text{row2}} \text{col2}$

R	$\rightarrow x$
R	$\rightarrow y$
D	$\rightarrow z$
D	$\rightarrow A$

$$\text{man}(x, y, z, A) + \text{mat}[i][j]$$

$$\text{row1} + \text{col1} = \text{row2} + \text{col2}$$

$$\text{row1} + \text{col1} - \text{row2} = \text{col2}$$



11 Pseudo code

$\text{int } [] [] [] dP_0$

int cherryPickup(int s1, int e1, int s2, int e2, int s3, int e3) {

$$C_{12} = \sin \frac{1}{2} + C_{01} - \sin 2$$

if ($\text{row1} \geq \text{mat.length} \text{ || } \text{row2} \geq \text{mat.length} \text{ || } \text{col1} \geq \text{mat.length}$)

|| Col 2 >= max.length) { return Integer.MIN_VALUE; }

if (mat[0][0] == -1 || mat[0][1] == -1)

~~return Integer.MIN_VALUE;~~

if (row1 == mat.length - 1 && col1 == mat[0].length - 1)

`let row2 := mat.length.182 col2 := mat[0].length.182`

return mat[rows][cols];

if $(dp[\text{down1}][\text{cold1}][\text{down2}] \neq -1)$ action

$$dP[\delta\omega_1][c_0\ell_1][\delta\omega_2][c_{012}]\}$$

int temp1 = CherryPickup(mat, 0DW1, Col1D1, 0DW2, Col2D1); ~~Col2D1~~ → RR

int temp2 = CherryPickup(mat, 0DW1, colJ+1, 0DW2+1, Col2); → RD

int tempP3 = CherryPickup(mat, 0DW1+1, Col1, 0DW2, Col2+!); → DR

int temp4 = CherryPickUp(mat, 2DW1+1, Col1, 2DW2+1, Col2); → PD

```
int ans = man(temp1, temp2, temp3, temp4);
```

int Contro := 0

if ($\text{row1} == \text{row2}$ & $\text{col1} == \text{col2}$) {

Contoh : `mat[rows][cols];`

1

else {

$$\text{Contar} = \text{mat}[\text{f00w}_1][\text{c011}] + \text{mat}[\text{f00w}_2][\text{c012}];$$

1 3

$dp[\text{row1}][\text{col1}][\text{row2}] = \text{ans} + \text{Contrib}$



return $\text{ans} + \text{Contrib}$

3

You didn't come this far only to come this
far.



AlgoPrep

Number of Paths

Java Code:

```
class Solution{

    long numberOfPaths(int m, int n) {
        // Code Here
        long[][]dp = new long[m][n];

        for(int i=0;i<m;i++){
            Arrays.fill(dp[i],-1);
        }
        return pathshelper(m-1,n-1,dp);
    }

    public static long pathshelper(int sr, int sc,long[][]dp) {
        if(sr<0 || sc<0){
            return 0;
        }
        if(sr == 0 && sc == 0){
            return 1;
        }

        if(dp[sr][sc] != -1){
            return dp[sr][sc];
        }

        long path1 = pathshelper(sr - 1, sc,dp);
        long path2 = pathshelper(sr, sc - 1,dp);

        dp[sr][sc] = path1 + path2;
        return path1+path2;
    }
}
```

C++ Code:

```
#include <iostream>
#include <vector>
```

```

using namespace std;

class Solution {
public:
    long numberOfPaths(int m, int n) {
        vector<vector<long long>> dp(m, vector<long long>(n, -1));

        for (int i = 0; i < m; i++) {
            fill(dp[i].begin(), dp[i].end(), -1);
        }

        return pathsHelper(m - 1, n - 1, dp);
    }

    long pathsHelper(int sr, int sc, vector<vector<long long>>& dp) {
        if (sr < 0 || sc < 0) {
            return 0;
        }
        if (sr == 0 && sc == 0) {
            return 1;
        }

        if (dp[sr][sc] != -1) {
            return dp[sr][sc];
        }

        long path1 = pathsHelper(sr - 1, sc, dp);
        long path2 = pathsHelper(sr, sc - 1, dp);

        dp[sr][sc] = path1 + path2;
        return dp[sr][sc];
    }
};

```

Python Code:

```

class Solution:
    def numberOfPaths(self, m: int, n: int) -> int:
        dp = [[-1] * n for _ in range(m)]

        for i in range(m):
            for j in range(n):
                dp[i][j] = -1

        return self.pathsHelper(m - 1, n - 1, dp)

```

```

def pathsHelper(self, sr: int, sc: int, dp: List[List[int]]) -> int:
    if sr < 0 or sc < 0:
        return 0
    if sr == 0 and sc == 0:
        return 1

    if dp[sr][sc] != -1:
        return dp[sr][sc]

    path1 = self.pathsHelper(sr - 1, sc, dp)
    path2 = self.pathsHelper(sr, sc - 1, dp)

    dp[sr][sc] = path1 + path2
    return dp[sr][sc]

```

Number of Paths with blockade

Java Code:

```

import java.io.*;
import java.util.*;

public class Solution {

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

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

        long[][] dp = new long[m][n];

        for(int i=0;i<m;i++){
            Arrays.fill(dp[i],-1);
        }
    }
}

```

```

long ans = pathshelper(arr,m-1,n-1,dp);
System.out.println(ans);

}

public static long pathshelper(int[][]arr, int sr, int sc,long[][]dp) {
    if(sr<0 || sc<0 || arr[sr][sc] == 0){
        return 0;
    }
    if(sr == 0 && sc == 0){
        return 1;
    }

    if(dp[sr][sc] != -1){
        return dp[sr][sc];
    }

    long path1 = pathshelper(arr,sr - 1, sc,dp);
    long path2 = pathshelper(arr,sr, sc - 1,dp);

    dp[sr][sc] = path1 + path2;
    return path1+path2;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>

using namespace std;

long pathshelper(vector<vector<int>>& arr, int sr, int sc, vector<vector<long long>>& dp) {
    int m = arr.size();
    int n = arr[0].size();

    if (sr < 0 || sc < 0 || arr[sr][sc] == 0) {
        return 0;
    }

    if (sr == 0 && sc == 0) {
        return 1;
    }

    if (dp[sr][sc] != -1) {
        return dp[sr][sc];
    }

    long path1 = pathshelper(arr,sr - 1, sc,dp);
    long path2 = pathshelper(arr,sr, sc - 1,dp);

    dp[sr][sc] = path1 + path2;
    return path1+path2;
}

```

```

if (dp[sr][sc] != -1) {
    return dp[sr][sc];
}

long path1 = pathshelper(arr, sr - 1, sc, dp);
long path2 = pathshelper(arr, sr, sc - 1, dp);

dp[sr][sc] = path1 + path2;
return dp[sr][sc];
}

int main() {
    int m, n;
    cin >> m >> n;

    vector<vector<int>> arr(m, vector<int>(n));
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            cin >> arr[i][j];
        }
    }

    vector<vector<long long>> dp(m, vector<long long>(n, -1));
    long ans = pathshelper(arr, m - 1, n - 1, dp);
    cout << ans << endl;

    return 0;
}

```

Python Code:

```

def pathshelper(arr, sr, sc, dp):
    m, n = len(arr), len(arr[0])

    if sr < 0 or sc < 0 or arr[sr][sc] == 0:
        return 0

    if sr == 0 and sc == 0:
        return 1

    if dp[sr][sc] != -1:
        return dp[sr][sc]

    path1 = pathshelper(arr, sr - 1, sc, dp)
    path2 = pathshelper(arr, sr, sc - 1, dp)

```

```

dp[sr][sc] = path1 + path2
return dp[sr][sc]

def main():
    m, n = map(int, input().split())

    arr = []
    for _ in range(m):
        row = list(map(int, input().split()))
        arr.append(row)

    dp = [[-1 for _ in range(n)] for _ in range(m)]
    ans = pathhelper(arr, m - 1, n - 1, dp)
    print(ans)

if __name__ == "__main__":
    main()

```

Minimum Path Sum

Java Code:

```

Java
class Solution {
    public int minPathSum(int[][] grid) {
        int[][] dp = new int[grid.length][grid[0].length];
        for(int i=0;i<grid.length;i++){
            Arrays.fill(dp[i],-1);
        }
        return
minPathSumhelper(grid,grid.length-1,grid[0].length-1,dp);
    }

    public int minPathSumhelper(int[][] grid, int row,int col,int[][]dp) {
        if(row < 0 || col < 0){
            return Integer.MAX_VALUE;
        }
        if(dp[row][col] != -1)
            return dp[row][col];
        int down = minPathSumhelper(grid, row+1, col, dp);
        int right = minPathSumhelper(grid, row, col+1, dp);
        dp[row][col] = Math.min(down, right) + grid[row][col];
        return dp[row][col];
    }
}

```

```

    }

    if(row == 0 && col == 0){
        return grid[row][col];
    }

    if(dp[row][col] != -1){
        return dp[row][col];
    }

    int down = minPathSumhelper(grid, row-1, col, dp);
    int right = minPathSumhelper(grid, row, col-1, dp);

    int ans = Math.min(down, right) + grid[row][col];
    dp[row][col] = ans;
    return ans;
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <limits>

using namespace std;

class Solution {
public:
    int minPathSum(vector<vector<int>>& grid) {
        int m = grid.size();
        int n = grid[0].size();

        vector<vector<int>> dp(m, vector<int>(n, -1));

        for (int i = 0; i < m; i++) {
            fill(dp[i].begin(), dp[i].end(), -1);
        }

        return minPathSumHelper(grid, m - 1, n - 1, dp);
    }
}

```

```

}

int minPathSumHelper(vector<vector<int>>& grid, int row, int col, vector<vector<int>>& dp) {
    if (row < 0 || col < 0) {
        return numeric_limits<int>::max();
    }

    if (row == 0 && col == 0) {
        return grid[row][col];
    }

    if (dp[row][col] != -1) {
        return dp[row][col];
    }

    int up = minPathSumHelper(grid, row - 1, col, dp);
    int left = minPathSumHelper(grid, row, col - 1, dp);

    int ans = min(up, left) + grid[row][col];
    dp[row][col] = ans;
    return ans;
}
};


```

Python Code:

```

class Solution:
    def minPathSum(self, grid: List[List[int]]) -> int:
        m = len(grid)
        n = len(grid[0])

        dp = [[-1 for _ in range(n)] for _ in range(m)]

        for i in range(m):
            for j in range(n):
                dp[i][j] = -1

        return self.minPathSumHelper(grid, m - 1, n - 1, dp)

    def minPathSumHelper(self, grid: List[List[int]], row: int, col: int, dp: List[List[int]]) -> int:
        if row < 0 or col < 0:
            return float('inf')

        if row == 0 and col == 0:
            return grid[row][col]

```

```

if dp[row][col] != -1:
    return dp[row][col]

up = self.minPathSumHelper(grid, row - 1, col, dp)
left = self.minPathSumHelper(grid, row, col - 1, dp)

ans = min(up, left) + grid[row][col]
dp[row][col] = ans
return ans

```

Cherry PickUp

Java Code:

```

Java
class Solution {
    public int cherryPickup(int[][][] grid) {
        int[][][]dp = new
int[grid.length][grid[0].length][grid.length];
        int ans = cherryPickuphelper(grid,0,0,0,dp);

        if(ans == Integer.MIN_VALUE){
            return 0;
        }else{
            return ans;
        }
    }

    //row1 + col1 == row2 + col2 ==> col2 = row1 + col1 - row2
    public int cherryPickuphelper(int[][][] grid,int row1,int
col1,int row2,int[][][]dp){
        int col2 = row1 + col1 - row2;
        if(row1>=grid.length || row2>= grid.length || col1 >=
grid[0].length|| col2 >= grid[0].length || grid[row1][col1] ==
-1||grid[row2][col2] == -1){
            return Integer.MIN_VALUE;
        }
    }
}

```

```

    }

    if(dp[row1][col1][row2] != 0){
        return dp[row1][col1][row2];
    }

    if(row1 == grid.length - 1 && col1 == grid[0].length - 1 &&
row2 == grid.length - 1 && col2 == grid[0].length - 1){
        return grid[row1][col1];
    }

    int temp1 = cherryPickuphelper(grid, row1, col1+1, row2, dp);
    int temp2 =
cherryPickuphelper(grid, row1+1, col1, row2+1, dp);
    int temp3 =
cherryPickuphelper(grid, row1, col1+1, row2+1, dp);
    int temp4 = cherryPickuphelper(grid, row1+1, col1, row2, dp);

    int max =
Math.max(Math.max(temp1, temp2), Math.max(temp3, temp4));

    int contri = 0;
    if(row1 == row2 && col1 == col2){
        contri = grid[row1][col1];
    }else{
        contri = grid[row1][col1]+grid[row2][col2];
    }

    if(max == Integer.MIN_VALUE){
        dp[row1][col1][row2] = Integer.MIN_VALUE;
        return Integer.MIN_VALUE;
    }else{
        dp[row1][col1][row2] = max+contri;
        return max+contri;
    }
}
}

```

C++ Code:

```
#include <iostream>
#include <vector>
#include <cstring>

using namespace std;

class Solution {
public:
    int cherryPickup(vector<vector<int>>& grid) {
        int m = grid.size();
        int n = grid[0].size();
        vector<vector<vector<int>>> dp(m, vector<vector<int>>(n, vector<int>(m, -1)));

        int ans = cherryPickupHelper(grid, 0, 0, 0, dp);

        if (ans == INT_MIN) {
            return 0;
        } else {
            return ans;
        }
    }

    int cherryPickupHelper(vector<vector<int>>& grid, int row1, int col1, int row2,
    vector<vector<vector<int>>& dp) {
        int col2 = row1 + col1 - row2;

        if (row1 >= grid.size() || row2 >= grid.size() || col1 >= grid[0].size() || col2 >= grid[0].size() || grid[row1][col1] == -1 || grid[row2][col2] == -1) {
            return INT_MIN;
        }

        if (dp[row1][col1][row2] != -1) {
            return dp[row1][col1][row2];
        }

        if (row1 == grid.size() - 1 && col1 == grid[0].size() - 1 && row2 == grid.size() - 1 && col2 == grid[0].size() - 1) {
            return grid[row1][col1];
        }

        int temp1 = cherryPickupHelper(grid, row1, col1 + 1, row2, dp);
        int temp2 = cherryPickupHelper(grid, row1 + 1, col1, row2 + 1, dp);
        int temp3 = cherryPickupHelper(grid, row1, col1 + 1, row2 + 1, dp);
        int temp4 = cherryPickupHelper(grid, row1 + 1, col1, row2, dp);

        dp[row1][col1][row2] = max({temp1, temp2, temp3, temp4}) + grid[row1][col1] + (grid[row2][col2] == 1 ? 1 : 0);
        return dp[row1][col1][row2];
    }
}
```

```

int maxTemp = max(max(temp1, temp2), max(temp3, temp4));

int contribution = (row1 == row2 && col1 == col2) ? grid[row1][col1] : grid[row1][col1] +
grid[row2][col2];

if (maxTemp == INT_MIN) {
    dp[row1][col1][row2] = INT_MIN;
} else {
    dp[row1][col1][row2] = maxTemp + contribution;
}

return dp[row1][col1][row2];
}
};


```

Python Code:

```

class Solution:
    def cherryPickup(self, grid: List[List[int]]) -> int:
        m, n = len(grid), len(grid[0])
        dp = [[[None] * m for _ in range(n)] for _ in range(m)] 

        ans = self.cherryPickupHelper(grid, 0, 0, 0, dp)

        if ans == float('-inf'):
            return 0
        else:
            return ans

    def cherryPickupHelper(self, grid: List[List[int]], row1: int, col1: int, row2: int, dp: 
List[List[List[int]]]) -> int:
        col2 = row1 + col1 - row2

        if row1 >= len(grid) or row2 >= len(grid) or col1 >= len(grid[0]) or col2 >= len(grid[0]) or
grid[row1][col1] == -1 or grid[row2][col2] == -1:
            return float('-inf')

        if dp[row1][col1][row2] is not None:
            return dp[row1][col1][row2]

        if row1 == len(grid) - 1 and col1 == len(grid[0]) - 1 and row2 == len(grid) - 1 and col2 ==
len(grid[0]) - 1:
            return grid[row1][col1]

        temp1 = self.cherryPickupHelper(grid, row1, col1 + 1, row2, dp)

```

```
temp2 = self.cherryPickupHelper(grid, row1 + 1, col1, row2 + 1, dp)
temp3 = self.cherryPickupHelper(grid, row1, col1 + 1, row2 + 1, dp)
temp4 = self.cherryPickupHelper(grid, row1 + 1, col1, row2, dp)

maxTemp = max(temp1, temp2, temp3, temp4)

contribution = grid[row1][col1] if row1 == row2 and col1 == col2 else grid[row1][col1] +
grid[row2][col2]

if maxTemp == float('-inf'):
    dp[row1][col1][row2] = float('-inf')
else:
    dp[row1][col1][row2] = maxTemp + contribution

return dp[row1][col1][row2]
```

Java



Today's agenda

↳ Intro

↳ Types of graph

↳ Storage

↳ BJS (level order) +1



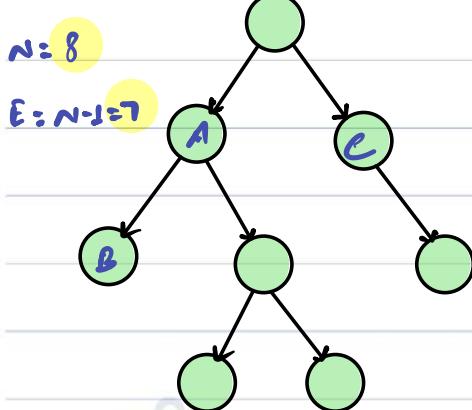
AlgoPrep



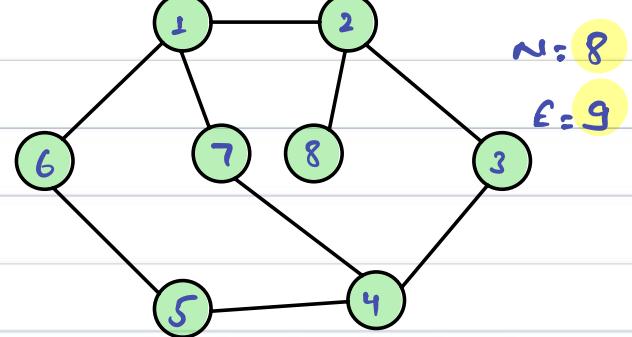
Intro

↳ Graph: Connection of nodes & edges

1. Tree



Graph



2, 6 & 7 are nodes of 1.

→ Main diff betn trees & graphs

1. Nodes in graph can have more than 1 Parent Node.

2. Graph has no hierarchy or root node.

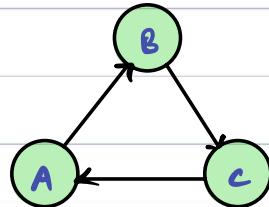
3. Graph can have cycles.

4. Any directional movement is allowed in graph.



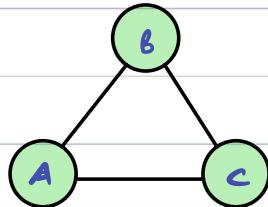
* Classification of graphs

Case I: Based on types of edges.



↳ directed graph

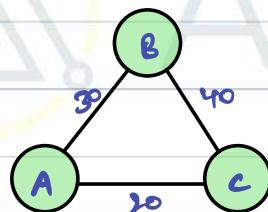
↳ insta follower



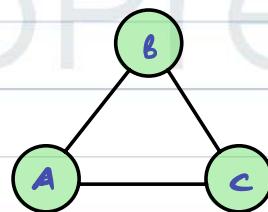
↳ undirected graph/Bidirectional

↳ facebook friend

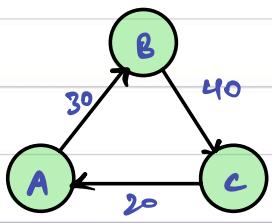
Case II: Based on weight of edge



↳ weighted graph



↳ unweighted graph



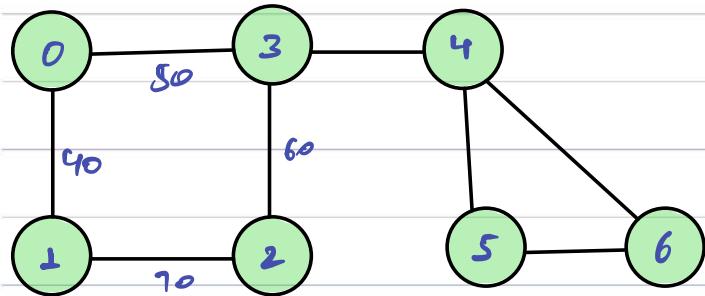
A directed weighted graph



AlgoPrep

Storing a graph

undirected graph



input



$N=7$ \rightarrow No. of nodes
 $M=8$ \rightarrow No. of edges

3	4	20
1	2	
2	3	
4	6	
0	1	
4	5	
5	6	
0	3	

① Adjacency matrix representation

Graph

		node	0	1	2	3	4	5	6
node		0	0	1	0	1	0	0	0
	1	1	0	1	0	0	0	0	0
2	0	1	0	0	1	0	0	0	0
3	1	0	0	1	0	1	0	0	0
4	0	0	0	0	1	0	1	1	0
5	0	0	0	0	1	0	0	1	0
6	0	0	0	0	1	0	1	0	0

7x7

0 → disconnected

1 → connected

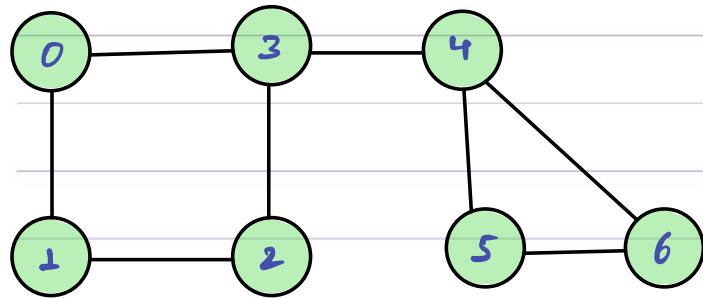
issues:

1. **Space wastage**

2. Say edge weight +ve / -ve, difficult to represent.



② adjacency list representation

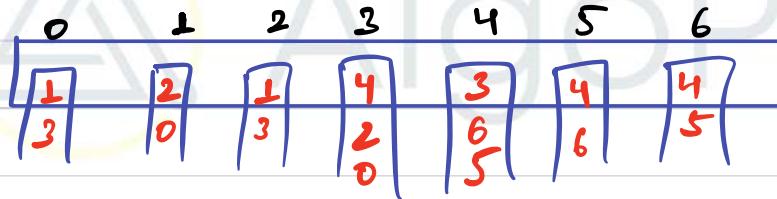


$i=0$

$N=7$ \rightarrow No. of nodes
 $m=8$ \rightarrow No. of edges

	0	1
0	3	4
1	—	2
2	2	3
3	4	6
4	0	—
5	4	5
6	5	6
7	0	3

`list<list<Integer>> graph = new ArrayList<>();`





main() {

Scanner scn = new Scanner(System.in);

int n = scn.nextInt();

int m = scn.nextInt();

int[][] edges = new int[m][2]

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

edges[i][0] = scn.nextInt();

edges[i][1] = scn.nextInt();

}

construction(n, m, edges);

}

construction(int n, int m, int[][] edges) {

List<List<Integer>> graph = new ArrayList<>();

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

graph.add(new ArrayList<>());

}

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

int u = edges[i][0]; → 3

int v = edges[i][1]; → 4

graph.get(u).add(v);

graph.get(v).add(u);

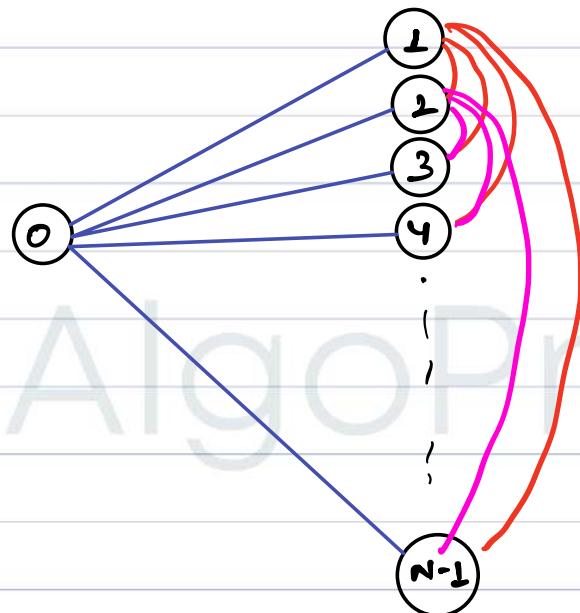
3

3



Break till 9:45 pm

Q) Find max number of edges Possible if N nodes are present in graph.



$$(N-1) + (N-2) + (N-3) + \dots + 1$$

↳ $\frac{N*(N-1)}{2}$

↳ Graph-3

→

for \rightarrow $C_{graph-2}$ $\rightarrow 8:00AM$



{ mon \rightarrow miscell.
wed \rightarrow off
fri \rightarrow off

Mon \rightarrow levelup



AlgoPrep



Today's agenda

↳ BFS +!

↳ Rotting oranges



AlgoPrep

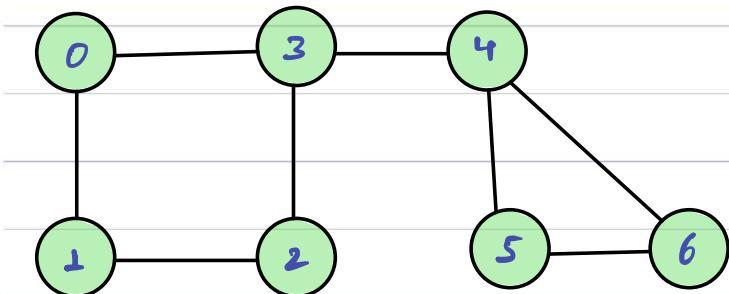
Tree → Pre, Post & in

&

level order → BFS (breadth first search)



BFS traversal



$i=0$

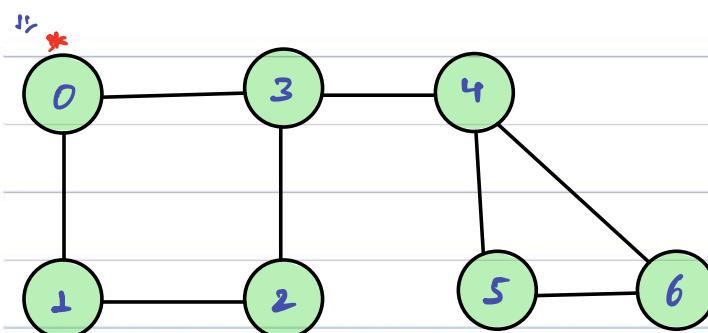
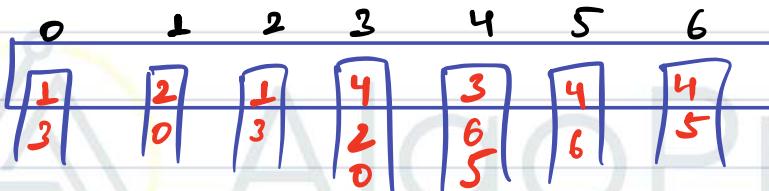
\rightarrow No. of nodes
 $N=7$

\rightarrow No. of edges
 $M=8$

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

`List<List<Integer>> graph = new ArrayList<>();`

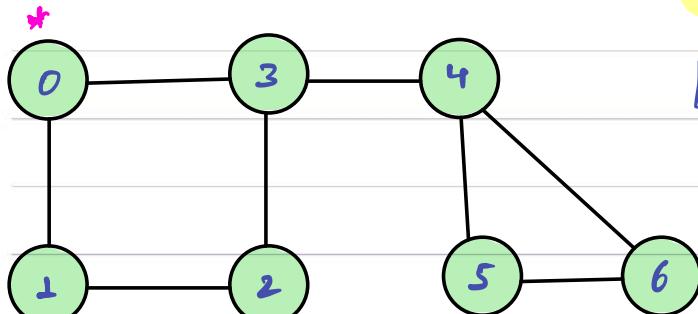
graph



\rightarrow In graph traversal,
you will have to decide the
start point.

b o (1 3) (2 4) (5 6)

BFS → level order in graph



graph

0	1	2	3	4	5	6
1	0	3	1	2	0	5
3	0	3	2	2	6	6
1	3	0	0	0	5	5
0	0	0	0	0	0	0

(0)(1 3)(2 4)(6 5)

queue: 0 1 3 2 4 6 5

vis:

T	F	F	F	F	F	F
0	1	2	3	4	5	6

dem = 5

while (q.size() > 0) {

 int dem = q.remove();
 S.O.P (dem);

// add all unvisited nodes

list<Integer> nbss = graph.get(dem);

for (int v: nbss) {
 if (vis[v] == false) {
 q.add(v);
 vis[v] = true;}}

}



// Pseudo code

```
void BFS (int n, int m, int [][] edges) {
```

```
list<list<integer>> graph = construction(n,m,edges);
```

```
Queue<Integer> q = new LinkedList<()>;  
boolean[] vis = new boolean[n];
```

```
q.add(0);  
vis[0] = true;
```

```
while (q.size() > 0) {
```

```
int elem = q.remove();  
S.O.P (elem);
```

// add all unvisited nodes

```
list<Integer> nbss = graph.get(elem);
```

```
for (int v: nbss) {
```

```
if (vis[v] == false) {  
q.add(v);  
vis[v] = true;
```

}

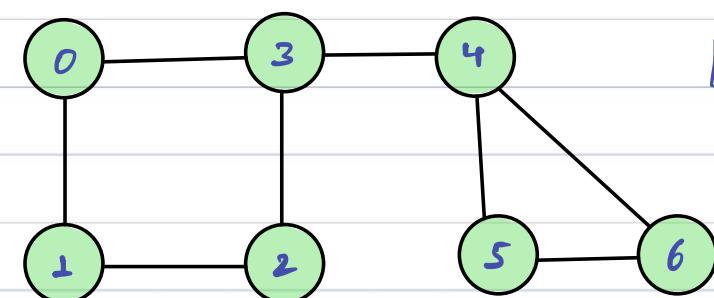
}

3



Q) Given undirected graph, source node and destination node. Check if destination node can be visited from source node or not?

SN: 0 DN: 2



graph

0	1	2	3	4	5	6
1 3	2 0	1 3	2 0	6 5	4 6	4 5

↳ 1 disconnected graph

Idea

↳ Do BFS and check if `vis[dn]` is true or not.



// Pseudo code

boolean BFS (int n, int m, int[][] edges, int sn, int dn) {

list<list<integer>> graph = Construction(n, m, edges);

Queue<Integer> q = new LinkedList<()>;
boolean[] vis = new boolean[n];

T.C: $O(N + M)$
S.C: $O(N)$
No. of nodes
No. of edges

q.add(sn);
vis[0] = true;

while (q.size() > 0) {

int elem = q.remove();
S.O.P(elem);

// add all unvisited nbss

list<Integer> nbss = graph.get(elem);

for (int v: nbss) {

if (vis[v] == false) {
q.add(v);
vis[v] = true;}

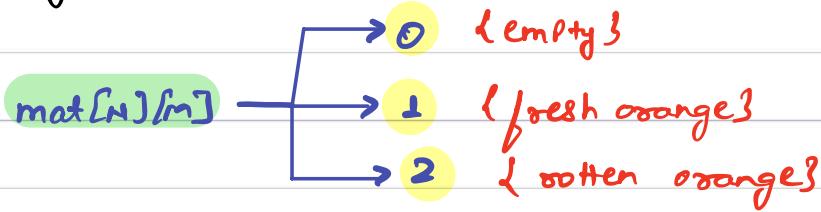
}

}
if (vis[dn] == true) { return true; }
else { return false; }

Break till 9:10 AM



Q) Rotting oranges



↳ Every minute any fresh orange adjacent to rotten orange becomes rotten. Find min time when all oranges become rotten. If not possible to rot all oranges, return -1

Ex1:

	0	1	2	3	4
0	1	2	0	1	2
1	1	2	1	2	1
2	0	2	0	1	2
3	0	1	2	1	2
4	1	2	1	2	0

$T=0$
 \downarrow
 $T=1$
 \downarrow
 $T=2$
 \downarrow
 $T=3$
 \downarrow
 $T=4$
 \downarrow
 $T=5$

Ex2:

	0	1	2	3	4
0	1	2	0	1	2
1	1	2	1	2	1
2	0	2	0	1	0
3	0	1	2	1	2
4	1	2	1	2	0

$T=0$
 \downarrow
 $T=1$
 \downarrow
 $T=2$
 \downarrow
 $T=3$



En3: $\begin{matrix} 1 & 0 & 2 \\ 0 & 0 & 2 \end{matrix} \rightarrow -1$

Multidource 633

0	1	2	3	4	
0	1	0	2	12	$T=0$
1	1	12	1	12	$T=1$
2	0	2	0	1	0
3	0	12	1	1	1
4	1	1	1	2	0

$T=2$

Diagram illustrating the state transition from $T=0$ to $T=1$ to $T=2$. The states are represented by pairs of numbers. The initial state at $T=0$ is $(0,3)$. At $T=1$, it transitions to $(0,2), (1,3), (0,4)$. At $T=2$, it transitions to $(1,2)$. The final state at $T=2$ is $(1,2)$.

q:	$\boxed{\{0,2,0\}} \quad \{2,3,0\} \quad \{4,3,0\} \quad \{0,2,1\} \quad \{1,3,1\} \quad \{0,4,1\}}$
	$\boxed{\{1,1,1\} \quad \{3,1,1\}}$

$$\text{dem} = \{2, 1, 0\}$$

Class Pair {

int $i;$ $\rightarrow \text{row}$

int $j;$ $\rightarrow \text{col}$

int $t;$

Pair(int n, int y, int z);

$i = n;$ $j = y;$ $t = z;$



// Pseudo code

```
int rottingOranges (int mat[n][m]) {
```

```
Queue < Pair > q = new LinkedList<>();
```

```
for (int i=0; i<n; i++) {  
    for (int j=0; j<m; j++) {  
        if (mat[i][j] == 2) {  
            Pair p = new Pair (i, j, 0);  
            q.add (p);  
        }  
    }  
}
```

```
int ans = -1;
```

```
while (q.size() > 0) {
```

```
    Pair rem = q.remove();
```

```
    int cRow = rem.i;
```

```
    int cCol = rem.j;
```

```
    int cTime = rem.t;
```

```
    ans = cTime;
```

```
// cRow-1, cCol
```

```
if (cRow-1 >= 0 & mat[cRow-1][cCol] == 1)
```

```
    q.add (new Pair (cRow-1, cCol, cTime+1));
```

```
    mat[cRow-1][cCol] = 2;
```

```
3
```



T.C: $O(N \times m)$

S.C: $O(1) + O(Nm)$

// C_{row}, C_{col}-1

if (C_{col}-1 >= 0 & mat[C_{row}][C_{col}-1] == 1) {
q.add(new Pair(C_{row}, C_{col}-1, C_{time}+1));
mat[C_{row}][C_{col}-1] = 2;

3

// C_{row}+1, C_{col}

if (C_{row}+1 < N & mat[C_{row}+1][C_{col}] == 1) {
q.add(new Pair(C_{row}+1, C_{col}, C_{time}+1));
mat[C_{row}+1][C_{col}] = 2;

3

// C_{row}, C_{col}+1

if (C_{col}+1 < m & mat[C_{row}][C_{col}+1] == 1) {
q.add(new Pair(C_{row}, C_{col}+1, C_{time}+1));
mat[C_{row}][C_{col}+1] = 2;

3

for (int i=0; i<N; i++) {
for (int j=0; j<m; j++) {
if (mat[i][j] == 1) {

return -1;

3 3

| 3



return ans;

}

ans = 0

	0	1	2
0	1	2	5
1	4	3	6
2	7	8	9

row = 0 col = 0

if (mat[row+1][col] < mat[row][col+1])
row++;

ans += mat[row][col];

}

else

col++;

ans += mat[row][col];

}

BFS Traversal

Java Code:

```
class Solution {  
    // Function to return Breadth First Traversal of given graph.  
    public ArrayList<Integer> bfsOfGraph(int V, ArrayList<ArrayList<Integer>> adj) {  
        // Code here  
        Queue<Integer> queue = new LinkedList<>();  
        boolean[] vis = new boolean[V];  
        ArrayList<Integer> ans = new ArrayList<>();  
  
        queue.add(0);  
        vis[0] = true;  
  
        while (queue.size() != 0) {  
            int rem = queue.remove();  
  
            ans.add(rem);  
  
            for (int v : adj.get(rem)) {  
                if (vis[v] == false) {  
                    vis[v] = true;  
                    queue.add(v);  
                }  
            }  
        }  
  
        return ans;  
    }  
}
```

C++ Code:

```
#include <iostream>  
#include <vector>  
#include <queue>  
  
class Solution {  
public:  
    std::vector<int> bfsOfGraph(int V, std::vector<std::vector<int>>& adj) {
```

```

std::queue<int> queue;
std::vector<bool> vis(V, false);
std::vector<int> ans;

queue.push(0);
vis[0] = true;

while (!queue.empty()) {
    int rem = queue.front();
    queue.pop();

    ans.push_back(rem);

    for (int v : adj[rem]) {
        if (!vis[v]) {
            vis[v] = true;
            queue.push(v);
        }
    }
}

return ans;
}

int main() {
    Solution solution;
    int V, E;
    std::cin >> V >> E; // Input the number of vertices and edges.

    std::vector<std::vector<int>> adj(V);
    for (int i = 0; i < E; i++) {
        int u, v;
        std::cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u); // Assuming an undirected graph.
    }

    std::vector<int> result = solution.bfsOfGraph(V, adj);

    // Output the BFS traversal.
    for (int node : result) {
        std::cout << node << " ";
    }
    std::cout << std::endl;

    return 0;
}

```

```
}
```

Python Code:

```
from collections import deque

class Solution:
    def bfsOfGraph(self, V: int, adj: List[List[int]]) -> List[int]:
        queue = deque()
        vis = [False] * V
        ans = []

        queue.append(0)
        vis[0] = True

        while queue:
            rem = queue.popleft()
            ans.append(rem)

            for v in adj[rem]:
                if not vis[v]:
                    vis[v] = True
                    queue.append(v)

        return ans

# Example usage:
# You can create an instance of Solution and call the bfsOfGraph method with the desired V and
# adjacency list.
solution = Solution()
V, E = map(int, input().split()) # Input the number of vertices and edges.

adj = [[] for _ in range(V)]
for _ in range(E):
    u, v = map(int, input().split())
    adj[u].append(v)
    adj[v].append(u) # Assuming an undirected graph.

result = solution.bfsOfGraph(V, adj)

# Output the BFS traversal.
for node in result:
    print(node, end=" ")
print()
```

Source to Destination Path

Java Code:

```
import java.io.*;
import java.util.*;

public class Solution {

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

        ArrayList<ArrayList<Integer>> graph = new ArrayList<>();
        for(int i=0;i<n;i++){
            graph.add(new ArrayList<>());
        }

        for(int i=0;i<m;i++){
            int u = scn.nextInt();
            int v = scn.nextInt();

            graph.get(u).add(v);
            graph.get(v).add(u);
        }
        int src = scn.nextInt();
        int dest = scn.nextInt();

        boolean[] vis = new boolean[n];
        sourceToDestinationPath(graph,n,src,vis);

        System.out.println(vis[dest]);
    }

    public static void sourceToDestinationPath(ArrayList<ArrayList<Integer>> graph,int n,int src,boolean[] vis){
        Queue<Integer> queue = new LinkedList<>();
        vis[src] = true;
        queue.add(src);

        while(!queue.isEmpty()){
            Integer curr = queue.poll();
            for(Integer neighbor : graph.get(curr)){
                if(!vis[neighbor]){
                    vis[neighbor] = true;
                    queue.add(neighbor);
                }
            }
        }
    }
}
```

```

ArrayList<Integer> ans = new ArrayList<>();

queue.add(src);
vis[src] = true;

while (queue.size() != 0) {
    int rem = queue.remove();

    for (int v : graph.get(rem)) {
        if (vis[v] == false) {
            vis[v] = true;
            queue.add(v);
        }
    }
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>

class Solution {
public:
    static void sourceToDestinationPath(std::vector<std::vector<int>>& graph, int n, int src,
    std::vector<bool>& vis) {
        std::queue<int> queue;

        queue.push(src);
        vis[src] = true;

        while (!queue.empty()) {
            int rem = queue.front();
            queue.pop();

            for (int v : graph[rem]) {
                if (!vis[v]) {
                    vis[v] = true;
                    queue.push(v);
                }
            }
        }
    }
}

```

```

int main() {
    int n, m;
    std::cin >> n >> m;

    std::vector<std::vector<int>> graph(n);
    for (int i = 0; i < n; i++) {
        graph[i] = std::vector<int>();
    }

    for (int i = 0; i < m; i++) {
        int u, v;
        std::cin >> u >> v;

        graph[u].push_back(v);
        graph[v].push_back(u);
    }

    int src, dest;
    std::cin >> src >> dest;

    std::vector<bool> vis(n, false);
    sourceToDestinationPath(graph, n, src, vis);

    std::cout << (vis[dest] ? "true" : "false") << std::endl;

    return 0;
}

int main() {
    Solution solution;
    solution.main();
    return 0;
}

```

Python Code:

```

from collections import deque

class Solution:
    @staticmethod

```

```

def sourceToDestinationPath(graph, n, src, vis):
    queue = deque()

    queue.append(src)
    vis[src] = True

    while queue:
        rem = queue.popleft()

        for v in graph[rem]:
            if not vis[v]:
                vis[v] = True
                queue.append(v)

def main(self):
    n, m = map(int, input().split())

    graph = [[] for _ in range(n)]

    for _ in range(m):
        u, v = map(int, input().split())
        graph[u].append(v)
        graph[v].append(u)

    src, dest = map(int, input().split())

    vis = [False] * n
    self.sourceToDestinationPath(graph, n, src, vis)

    if vis[dest]:
        print("true")
    else:
        print("false")

if __name__ == "__main__":
    solution = Solution()
    solution.main()

```

Rotting Oranges

Java Code:

```
Java
class Solution {

    static class Pair{
        int row;
        int col ;
        int time;
        Pair(int row, int col, int time){
            this.row = row;
            this.col = col;
            this.time = time;
        }
    }

    public int orangesRotting(int[][][] grid) {
        int m = grid.length;
        int n = grid[0].length;

        Queue<Pair> q = new LinkedList<>();

        int count1 = 0;
        int count2 = 0;
        for(int i = 0 ; i < m ; i++){
            for(int j = 0 ; j < n ; j++){
                if(grid[i][j] == 2){
                    q.add(new Pair(i, j, 0));
                    count2++;
                }else if(grid[i][j] == 1){
                    count1++;
                }
            }
        }

        if(count1 == 0){

    }}
```

```

        return 0;
    }else if(count2 == 0){
        return -1;
    }

int ans = -1;
while(!q.isEmpty()) {

    Pair removed = q.poll();
    int r = removed.row;
    int c = removed.col;
    int t = removed.time;

ans = removed.time;

    if(r + 1 < m && grid[r + 1][c] == 1 ) {
        q.add(new Pair(r + 1, c, t + 1));
        grid[r + 1][c] = 2;
    }
    if(r - 1 >= 0 && grid[r - 1][c] == 1 ) {
        q.add(new Pair(r - 1, c, t + 1));
        grid[r - 1][c] = 2;
    }
    if(c + 1 < n && grid[r][c + 1] == 1 ) {
        q.add(new Pair(r , c + 1, t + 1));
        grid[r][c + 1] = 2;
    }
    if(c - 1 >= 0 && grid[r][c - 1] == 1 ) {
        q.add(new Pair(r, c - 1 , t + 1));
        grid[r][c - 1] = 2;
    }
}

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

```

```

        if(grid[i][j] == 1) {
            return -1;
        }
    }

    return ans;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>

class Solution {
public:
    struct Pair {
        int row;
        int col;
        int time;
        Pair(int r, int c, int t) : row(r), col(c), time(t) {}
    };

    int orangesRotting(std::vector<std::vector<int>>& grid) {
        int m = grid.size();
        int n = grid[0].size();

        std::queue<Pair> q;
        int count1 = 0;
        int count2 = 0;

        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (grid[i][j] == 2) {
                    q.push(Pair(i, j, 0));
                    count2++;
                } else if (grid[i][j] == 1) {
                    count1++;
                }
            }
        }
    }
}

```

```

        }

    }

    if (count1 == 0) {
        return 0;
    } else if (count2 == 0) {
        return -1;
    }

    int ans = -1;
    while (!q.empty()) {
        Pair removed = q.front();
        q.pop();
        int r = removed.row;
        int c = removed.col;
        int t = removed.time;

        ans = removed.time;

        if (r + 1 < m && grid[r + 1][c] == 1) {
            q.push(Pair(r + 1, c, t + 1));
            grid[r + 1][c] = 2;
        }
        if (r - 1 >= 0 && grid[r - 1][c] == 1) {
            q.push(Pair(r - 1, c, t + 1));
            grid[r - 1][c] = 2;
        }
        if (c + 1 < n && grid[r][c + 1] == 1) {
            q.push(Pair(r, c + 1, t + 1));
            grid[r][c + 1] = 2;
        }
        if (c - 1 >= 0 && grid[r][c - 1] == 1) {
            q.push(Pair(r, c - 1, t + 1));
            grid[r][c - 1] = 2;
        }
    }

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            if (grid[i][j] == 1) {
                return -1;
            }
        }
    }

    return ans;
}

```

```
};

int main() {
    Solution solution;
    int m, n;
    std::cin >> m >> n;

    std::vector<std::vector<int>> grid(m, std::vector<int>(n));

    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            std::cin >> grid[i][j];
        }
    }

    int result = solution.orangesRotting(grid);
    std::cout << result << std::endl;

    return 0;
}
```

Python Code:

```
from collections import deque

class Solution:
    def orangesRotting(self, grid: List[List[int]]) -> int:
        m, n = len(grid), len(grid[0])
        directions = [(1, 0), (-1, 0), (0, 1), (0, -1)]

        q = deque()
        count1, count2 = 0, 0

        for i in range(m):
            for j in range(n):
                if grid[i][j] == 2:
                    q.append((i, j, 0))
                    count2 += 1
                elif grid[i][j] == 1:
                    count1 += 1

        if count1 == 0:
            return 0

        while q:
            i, j, t = q.popleft()
            for di, dj in directions:
                ni, nj = i + di, j + dj
                if 0 < ni < m and 0 < nj < n and grid[ni][nj] == 1:
                    grid[ni][nj] = 2
                    q.append((ni, nj, t + 1))
                    count2 += 1

        return t if count1 == count2 else -1
```

```

        return 0
    elif count2 == 0:
        return -1

    ans = -1
    while q:
        r, c, t = q.popleft()
        ans = t

        for dr, dc in directions:
            nr, nc = r + dr, c + dc
            if 0 <= nr < m and 0 <= nc < n and grid[nr][nc] == 1:
                q.append((nr, nc, t + 1))
                grid[nr][nc] = 2

    for i in range(m):
        for j in range(n):
            if grid[i][j] == 1:
                return -1

    return ans

```

```

# Example usage:
# You can create an instance of Solution and call the orangesRotting method with the desired
grid.
solution = Solution()
m, n = map(int, input().split()) # Input the number of rows and columns.

grid = []
for i in range(m):
    row = list(map(int, input().split()))
    grid.append(row)

result = solution.orangesRotting(grid)
print(result)

```



Today's agenda

↳ DFS

↳ No. of Connected Component +!

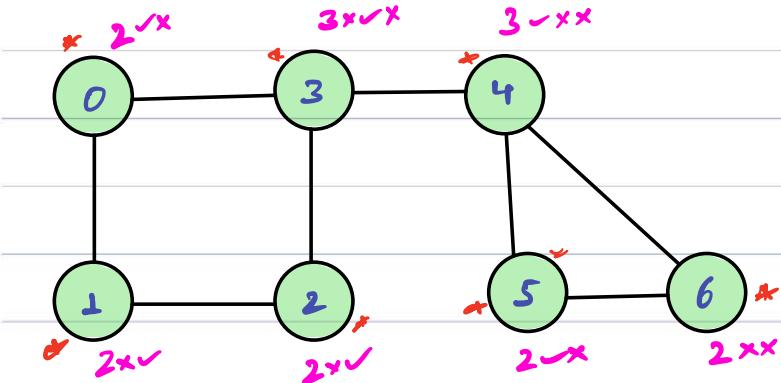
↳ Topological Sort +!



AlgoPrep



D3S → Depth first search traversal



↳ A traversal on graph in any order.

//Pseudo Code

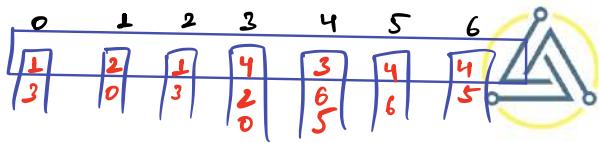
```
void dfs (list<list<integer>> graph, boolean vis[], int src){
```

```
    list<integer> nbcs = graph.get(src);
```

```
    for (int i=0; i<nbcs.size(); i++) {  
        int v = nbcs.get(i);  
        if (vis[v] == false) {  
            vis[v] = true;  
            dfs(graph, vis, v);  
        }  
    }
```

main C++

graph



3

```
void dfs (list<list<integer>> graph, boolean vis[], int src){
```

1 list<integer> nbrs = graph[src];

2 for (int i=0; i<nbrs.size(); i++) {

3 int v = nbrs.get(i);

4 if (vis[v] == false) {

5 vis[v] = true;

6 dfs(graph, vis, v);

src 0 1 2 3 4 5 6

1 2(0)

2 2(0)

1 2(0+2)

2 2(0)

1 2(0+1)

2 2(0+1)

1 2(0+1)

2 2(0+1)

1 2(0+1)

2 2(0+1)

1 2(0+1)

2 2(0+1)

0	1	2	3	4	5	6
FT						

3

src

nbrs

v = 0 1

0

1 3 1

3

4 2 0

v = 4 2 0

4

5 6 2

v = 5 6 2

5

4 6

v = 4 6

6

4 5

v = 4 5

2

1 3

v = 1 3

1

2 0

v = 2 0

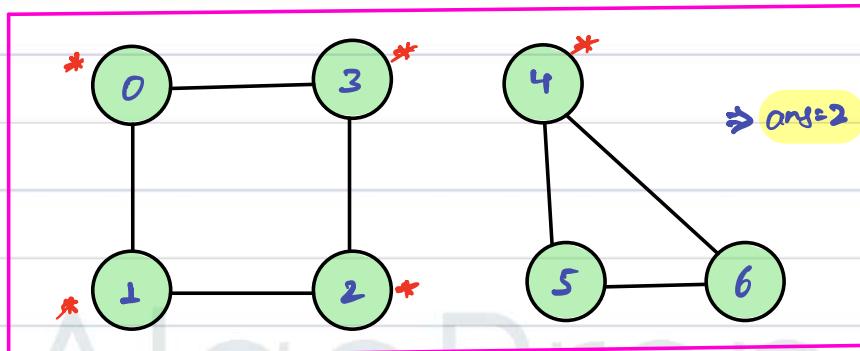


Q) Connected Components

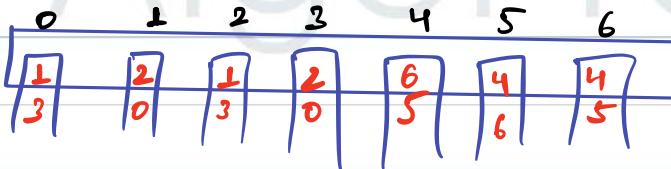
Given undirected graph, find no. of Connected Components.

Note: A component is said to be Connected, if from every node we can visit all nodes inside that component.

Ex:



graph





11 Pseudo code → multi function DFS

```
int main() {
    int n = ✓ No. of nodes m = ✓ No. of edges
    11 construction
    List<List<Integer>> graph = new ArrayList<List<Integer>>();
    int ans = 0;
    boolean[] vis = new boolean[n];
    for (int i = 0; i < n; i++) {
        if (vis[i] == false) {
            vis[i] = true;
            dfs(graph, vis, i);
            ans++;
        }
    }
    return ans;
}
```

```
void dfs (List<List<Integer>> graph, boolean vis[], int src) {
```

```
    List<Integer> nbrs = graph.get(src);
```

```
    for (int i = 0; i < nbrs.size(); i++) {
        int v = nbrs.get(i);
        if (vis[v] == false) {
            vis[v] = true;
            dfs(graph, vis, v);
        }
    }
}
```

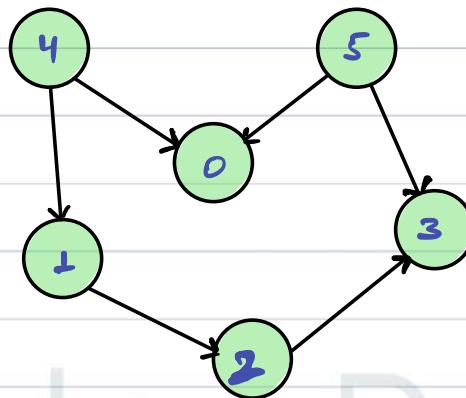


→ Kahn's algo

Q) Topological Sort

↳ Give the linear ordering of graph such that for every directed edge (u,v) , vertex u comes before v in the ordering. { DAG → Directed, Acyclic graph only }

Ex:



TS: 1 3 2 0 4 5 ~~xx~~

TS: 4 0 5 1 3 2 ~~xx~~

TS: 4 5 0 1 2 3 ~~xx~~

TS: 5 4 0 1 2 3 ~~xx~~

} multiple topological sort of same graph is possible.



// Kahn's algo

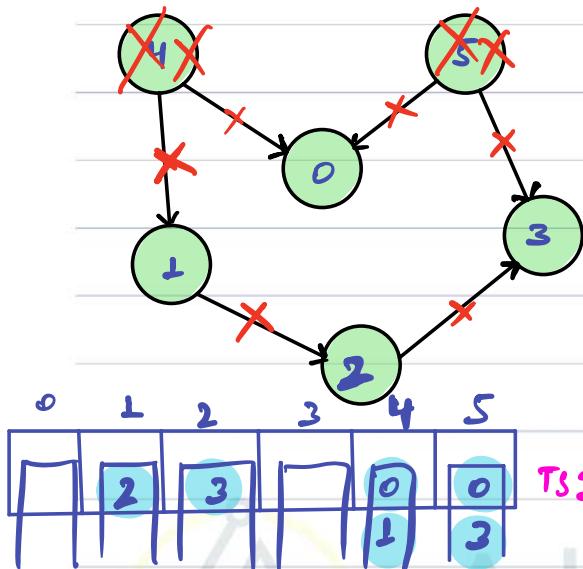
Count of incoming edges
for each node.

indegree:

0	1	2	3	4	5
2	1	1	2	0	0

q: ~~X 8 L 0 Z Z~~

rem = 3



TS: 4 5 1 0 2 3

// Pseudo Code

```
void topological_Sort (List<list<Integer>> graph, int n)
    int [] indegree = new int[n];
```

```
for (int u=0; u < n; u++) {
    List<Integer> nbrs = graph.get(u);
    for (int v: nbrs) {
        indegree[v]++;
    }
}
```



Queue<integer> q = new LinkedList<>();

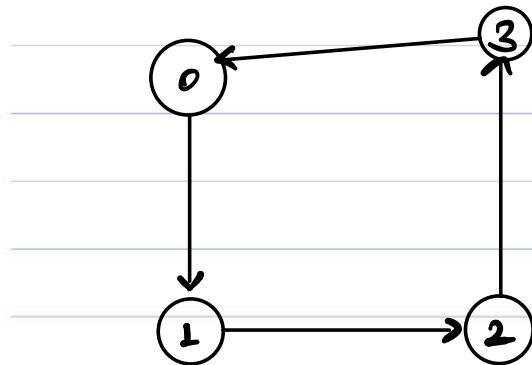
```
for (int i=0; i<m; i++) {  
    if (indegree[i] == 0) { q.add(i); }  
}
```

```
int count = 0;  
while (q.size() > 0) {  
    int rem = q.remove();  
    S.O.P(rem); → count++;
```

```
List<integer> nb8 = graph.get(rem);  
for (int v: nb8) {  
    indegree[v]--;  
    if (indegree[v] == 0) { q.add(v); }  
}  
  
if (count == n) { S.O.P("acyclic"); }  
else { S.O.P("cyclic"); }
```

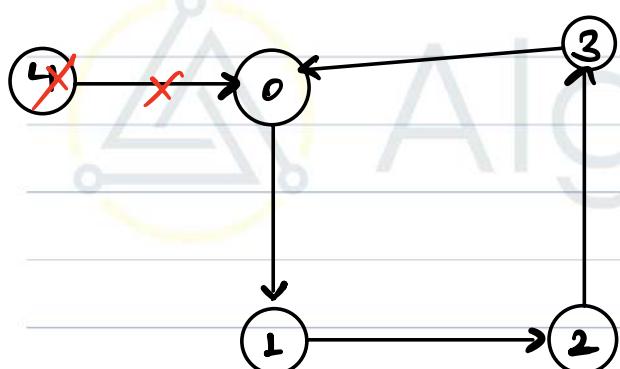


① Why not cyclic graph?



indegree:

0	1	2	3
1	1	1	1



indegree:

0	1	2	3	4
2	1	1	1	0

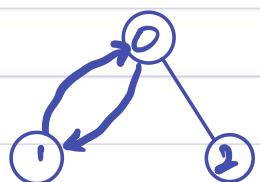
$q = 4$

dem = 4

4

In cyclic graph, dependency is also cyclic.
No node to start the topological sort.

② Why not undirected graph? → Same issue as above.



→ if "DAG" term is in the Problem Statement,
most likely first step is going to be
topological sort.



Directed

acyclic

Topological sort
is possible

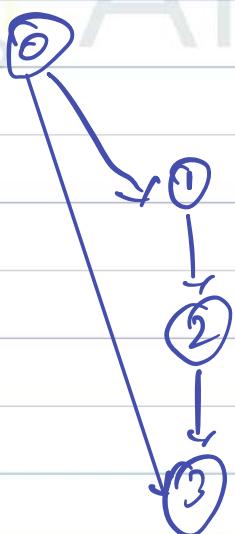
cyclic

Topological sort is
not possible

detect cycle?



0	1	2	3
0	1	1	2



2 8 1 2

ans = 1

0 1 3 2

HW



Q) Course Schedule

Given n courses that you have to take. You are also given Prerequisite in the form $[x_i, y_i]$ indicating that you must take course y_i before x_i . Come up with a valid ordering of all the courses that you are taking.

Ex: $n=5$ {1,0} {2,0} {3,1} {3,2} {3,0}



AlgoPrep

DFS of Graph

Java Code:

```
class Solution {  
    // Function to return a list containing the DFS traversal of the graph.  
    public ArrayList<Integer> dfsOfGraph(int V, ArrayList<ArrayList<Integer>> adj) {  
        // Code here  
  
        ArrayList<Integer> ans = new ArrayList<>();  
        boolean[] vis = new boolean[V];  
  
        ans.add(0);  
        vis[0] = true;  
  
        dfsOfGraphHelper(adj, 0, ans, vis);  
        return ans;  
    }  
  
    public void dfsOfGraphHelper(ArrayList<ArrayList<Integer>> adj, int src, ArrayList<Integer> ans, boolean[] vis){  
  
        ArrayList<Integer> nbrs = adj.get(src);  
  
        for(int v: nbrs){  
            if(vis[v] == false){  
                ans.add(v);  
                vis[v] = true;  
                dfsOfGraphHelper(adj, v, ans, vis);  
            }  
        }  
    }  
}
```

C++ Code:

```
#include <iostream>
```

```

#include <vector>

class Solution {
public:
    void dfsOfGraphHelper(const std::vector<std::vector<int>>& adj, int src, std::vector<int>&
ans, std::vector<bool>& vis) {
        const std::vector<int>& nbrs = adj[src];

        for (int v : nbrs) {
            if (!vis[v]) {
                ans.push_back(v);
                vis[v] = true;
                dfsOfGraphHelper(adj, v, ans, vis);
            }
        }
    }

    std::vector<int> dfsOfGraph(int V, std::vector<std::vector<int>>& adj) {
        std::vector<int> ans;
        std::vector<bool> vis(V, false);

        ans.push_back(0);
        vis[0] = true;

        dfsOfGraphHelper(adj, 0, ans, vis);

        return ans;
    }
};

int main() {
    Solution solution;
    int V, E;
    std::cin >> V >> E;

    std::vector<std::vector<int>> adj(V);

    for (int i = 0; i < E; i++) {
        int u, v;
        std::cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u);
    }

    std::vector<int> result = solution.dfsOfGraph(V, adj);

    for (int vertex : result) {

```

```

    std::cout << vertex << " ";
}

std::cout << std::endl;

return 0;
}

```

Python Code:

```

class Solution:
    def dfsOfGraphHelper(self, adj, src, ans, vis):
        nbrs = adj[src]

        for v in nbrs:
            if not vis[v]:
                ans.append(v)
                vis[v] = True
                self.dfsOfGraphHelper(adj, v, ans, vis)

    def dfsOfGraph(self, V, adj):
        ans = []
        vis = [False] * V

        ans.append(0)
        vis[0] = True

        self.dfsOfGraphHelper(adj, 0, ans, vis)

        return ans

if __name__ == "__main__":
    solution = Solution()
    V, E = map(int, input().split())

    adj = [[] for _ in range(V)]

    for _ in range(E):
        u, v = map(int, input().split())
        adj[u].append(v)
        adj[v].append(u)

    result = solution.dfsOfGraph(V, adj)

    for vertex in result:

```

```
print(vertex, end=" ")
print()
```

Connected Components(No. Of Provinces)

Java Code:

```
class Solution {
    static int numProvinces(ArrayList<ArrayList<Integer>> adj, int V) {
        //adjacency matrix to adjacency list
        ArrayList<ArrayList<Integer>> graph = new ArrayList<>();
        for(int i=0;i<V;i++){
            graph.add(new ArrayList<>());
        }

        for(int i=0;i<V;i++){
            for(int j=0;j<adj.get(i).size();j++){
                if(adj.get(i).get(j) == 1){
                    graph.get(i).add(j);
                    graph.get(j).add(i);
                }
            }
        }

        boolean[] vis = new boolean[V];
        int count = 0;

        for(int i=0;i<V;i++){
            if(vis[i] == false){
                dfsOfGraphHelper(graph,i,vis);
                count++;
            }
        }

        return count;
    }

    public static void dfsOfGraphHelper(ArrayList<ArrayList<Integer>> adj, int src, boolean[] vis){
```

```

ArrayList<Integer> nbrs = adj.get(src);

for(int v: nbrs){
    if(vis[v] == false){
        vis[v] = true;
        dfsOfGraphHelper(adj, v,vis);
    }
}

}

```

C++ Code:

```

#include <iostream>
#include <vector>

class Solution {
public:
    static void dfsOfGraphHelper(const std::vector<std::vector<int>>& adj, int src,
    std::vector<bool>& vis) {
        const std::vector<int>& nbrs = adj[src];

        for (int v : nbrs) {
            if (!vis[v]) {
                vis[v] = true;
                dfsOfGraphHelper(adj, v, vis);
            }
        }
    }

    static int numProvinces(std::vector<std::vector<int>>& adj, int V) {
        // Convert adjacency matrix to adjacency list
        std::vector<std::vector<int>> graph(V, std::vector<int>());

        for (int i = 0; i < V; i++) {
            for (int j = 0; j < adj[i].size(); j++) {
                if (adj[i][j] == 1) {
                    graph[i].push_back(j);
                    graph[j].push_back(i);
                }
            }
        }

        std::vector<bool> vis(V, false);

```

```

int count = 0;

for (int i = 0; i < V; i++) {
    if (!vis[i]) {
        dfsOfGraphHelper(graph, i, vis);
        count++;
    }
}

return count;
}
};

int main() {
    int V;
    std::cin >> V;
    std::vector<std::vector<int>> adj(V, std::vector<int>(V));

    for (int i = 0; i < V; i++) {
        for (int j = 0; j < V; j++) {
            std::cin >> adj[i][j];
        }
    }
}

int provinces = Solution::numProvinces(adj, V);
std::cout << provinces << std::endl;

return 0;
}

```

Python Code:

```

class Solution:
    @staticmethod
    def dfsOfGraphHelper(adj, src, vis):
        nbrs = adj[src]

        for v in nbrs:
            if not vis[v]:
                vis[v] = True
                Solution.dfsOfGraphHelper(adj, v, vis)

    @staticmethod
    def numProvinces(adj, V):
        # Convert adjacency matrix to adjacency list

```

```

graph = [[] for _ in range(V)]

for i in range(V):
    for j in range(len(adj[i])):
        if adj[i][j] == 1:
            graph[i].append(j)
            graph[j].append(i)

vis = [False] * V
count = 0

for i in range(V):
    if not vis[i]:
        Solution.dfsOfGraphHelper(graph, i, vis)
        count += 1

return count

if __name__ == "__main__":
    V = int(input())
    adj = []

    for _ in range(V):
        row = list(map(int, input().split()))
        adj.append(row)

provinces = Solution.numProvinces(adj, V)
print(provinces)

```

Topological Sorting

Java Code:

```

class Solution
{
    //Function to return list containing vertices in Topological order.
    static int[] topoSort(int V, ArrayList<ArrayList<Integer>> adj)
    {
        // add your code here

        int indegree[] = new int[V];
        for(int i=0;i<V;i++){
            for(int v: adj.get(i)){

```

```

        indegree[v]++;
    }
}

Queue<Integer> q = new LinkedList<>();

int topo[] = new int[V];

for(int i=0;i<V;i++){
    if(indegree[i]==0){
        q.add(i);
    }
}

int i=0;
while(!q.isEmpty()){
    int node=q.poll();
    topo[i++]=node;
    for(int v: adj.get(node)){
        indegree[v]--;
        if(indegree[v]==0){
            q.add(v);
        }
    }
}

return topo;
}
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>

class Solution {
public:
    static std::vector<int> topoSort(int V, std::vector<std::vector<int>>& adj) {
        std::vector<int> indegree(V, 0);
        for (int i = 0; i < V; i++) {
            for (int v : adj[i]) {
                indegree[v]++;
            }
        }

```

```

std::queue<int> q;
std::vector<int> topo(V);

for (int i = 0; i < V; i++) {
    if (indegree[i] == 0) {
        q.push(i);
    }
}

int i = 0;
while (!q.empty()) {
    int node = q.front();
    q.pop();
    topo[i++] = node;

    for (int v : adj[node]) {
        indegree[v]--;
        if (indegree[v] == 0) {
            q.push(v);
        }
    }
}

return topo;
}
};

int main() {
    int V, E;
    std::cin >> V >> E;
    std::vector<std::vector<int>> adj(V);

    for (int i = 0; i < E; i++) {
        int u, v;
        std::cin >> u >> v;
        adj[u].push_back(v);
    }

    std::vector<int> result = Solution::topoSort(V, adj);

    for (int vertex : result) {
        std::cout << vertex << " ";
    }

    std::cout << std::endl;

    return 0;
}

```

```
}
```

Python Code:

```
from collections import defaultdict, deque
```

```
class Solution:
```

```
    @staticmethod
```

```
    def topoSort(V, adj):
```

```
        indegree = [0] * V
```

```
        for i in range(V):
```

```
            for v in adj[i]:
```

```
                indegree[v] += 1
```

```
        q = deque()
```

```
        topo = [0] * V
```

```
        for i in range(V):
```

```
            if indegree[i] == 0:
```

```
                q.append(i)
```

```
        i = 0
```

```
        while q:
```

```
            node = q.popleft()
```

```
            topo[i] = node
```

```
            i += 1
```

```
            for v in adj[node]:
```

```
                indegree[v] -= 1
```

```
                if indegree[v] == 0:
```

```
                    q.append(v)
```

```
        return topo
```

```
if __name__ == "__main__":
```

```
    V, E = map(int, input().split())
```

```
    adj = defaultdict(list)
```

```
    for _ in range(E):
```

```
        u, v = map(int, input().split())
```

```
        adj[u].append(v)
```

```
    result = Solution.topoSort(V, adj)
```

```
    for vertex in result:
```

```
print(vertex, end=" ")
print()
```

Detect Cycle in Directed Graph

Java Code:

```
class Solution {
    // Function to detect cycle in a directed graph.
    public boolean isCyclic(int V, ArrayList<ArrayList<Integer>> adj) {
        // code here
        int indegree[] = new int[V];
        for(int i=0;i<V;i++){
            for(int v: adj.get(i)){
                indegree[v]++;
            }
        }

        Queue<Integer> q = new LinkedList<>();

        int topo[] = new int[V];

        for(int i=0;i<V;i++){
            if(indegree[i]==0){
                q.add(i);
            }
        }

        int count = 0;
        while(!q.isEmpty()){
            int node=q.poll();
            count++;

            for(int v: adj.get(node)){
                indegree[v]--;
                if(indegree[v]==0){
                    q.add(v);
                }
            }
        }
    }
}
```

```

        if(count == V){
            return false;
        }else{
            return true;
        }
    }
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <queue>

class Solution {
public:
    bool isCyclic(int V, std::vector<std::vector<int>>& adj) {
        std::vector<int> indegree(V, 0);
        for (int i = 0; i < V; i++) {
            for (int v : adj[i]) {
                indegree[v]++;
            }
        }

        std::queue<int> q;

        for (int i = 0; i < V; i++) {
            if (indegree[i] == 0) {
                q.push(i);
            }
        }

        int count = 0;
        while (!q.empty()) {
            int node = q.front();
            q.pop();
            count++;

            for (int v : adj[node]) {
                indegree[v]--;
                if (indegree[v] == 0) {
                    q.push(v);
                }
            }
        }
    }
}

```

```

        return count != V;
    }
};

int main() {
    int V, E;
    std::cin >> V >> E;
    std::vector<std::vector<int>> adj(V);

    for (int i = 0; i < E; i++) {
        int u, v;
        std::cin >> u >> v;
        adj[u].push_back(v);
    }

    Solution solution;
    bool isCyclic = solution.isCyclic(V, adj);

    if (isCyclic) {
        std::cout << "Graph contains a cycle." << std::endl;
    } else {
        std::cout << "Graph does not contain a cycle." << std::endl;
    }

    return 0;
}

```

Python Code:

```

from collections import deque

class Solution:
    def isCyclic(self, V, adj):
        indegree = [0] * V
        for i in range(V):
            for v in adj[i]:
                indegree[v] += 1

        q = deque()
        for i in range(V):
            if indegree[i] == 0:
                q.append(i)

        count = 0

```

```

while q:
    node = q.popleft()
    count += 1

    for v in adj[node]:
        indegree[v] -= 1
        if indegree[v] == 0:
            q.append(v)

return count != V

if __name__ == "__main__":
    V, E = map(int, input().split())
    adj = [[] for _ in range(V)]

    for _ in range(E):
        u, v = map(int, input().split())
        adj[u].append(v)

    solution = Solution()
    isCyclic = solution.isCyclic(V, adj)

    if isCyclic:
        print("Graph contains a cycle.")
    else:
        print("Graph does not contain a cycle.")

```

Course Schedule

Java Code:

```

Java
class Solution {
    public boolean canFinish(int numCourses, int[][] prerequisites) {

        HashMap<Integer, ArrayList<Integer>> hmap = new
        HashMap<>();

```

```

Queue<Integer> q = new LinkedList<>();
int[] inDegree = new int[numCourses];

//put the mapping in HashMap and calculate inDegree

for(int[] p : prerequisites){
    int start = p[1];
    int end = p[0];

    inDegree[end]++;
}

if(!hmap.containsKey(start)){
    hmap.put(start , new ArrayList<Integer>());
}
hmap.get(start).add(end);

}

for(int i = 0 ; i < inDegree.length ; i++){
    if(inDegree[i] == 0){
        q.add(i);
    }
}

while(!q.isEmpty()){
    int removed = q.poll();

    if(hmap.containsKey(removed)){
        for(int r : hmap.get(removed)){
            inDegree[r]--;
            if(inDegree[r] == 0){
                q.add(r);
            }
        }
    }
    numCourses--;
}

```

```

        }

        return numCourses == 0;
    }
}

```

C++ Code:

```

#include <iostream>
#include <vector>
#include <unordered_map>
#include <queue>

class Solution {
public:
    bool canFinish(int numCourses, std::vector<std::vector<int>>& prerequisites) {
        std::unordered_map<int, std::vector<int>> hmap;
        std::queue<int> q;
        std::vector<int> inDegree(numCourses, 0);

        // Put the mapping in the unordered_map and calculate inDegree
        for (auto& p : prerequisites) {
            int start = p[1];
            int end = p[0];

            inDegree[end]++;
            if (hmap.find(start) == hmap.end()) {
                hmap[start] = std::vector<int>();
            }
            hmap[start].push_back(end);
        }

        for (int i = 0; i < numCourses; i++) {
            if (inDegree[i] == 0) {
                q.push(i);
            }
        }

        while (!q.empty()) {
            int removed = q.front();
            q.pop();

```

```

        if (hmap.find(removed) != hmap.end()) {
            for (int r : hmap[removed]) {
                inDegree[r]--;
                if (inDegree[r] == 0) {
                    q.push(r);
                }
            }
            numCourses--;
        }

        return numCourses == 0;
    }
};

int main() {
    int numCourses, numPrerequisites;
    std::cin >> numCourses >> numPrerequisites;
    std::vector<std::vector<int>> prerequisites(numPrerequisites, std::vector<int>(2));

    for (int i = 0; i < numPrerequisites; i++) {
        std::cin >> prerequisites[i][0] >> prerequisites[i][1];
    }

    Solution solution;
    bool canFinish = solution.canFinish(numCourses, prerequisites);

    if (canFinish) {
        std::cout << "It is possible to finish all courses." << std::endl;
    } else {
        std::cout << "It is not possible to finish all courses." << std::endl;
    }

    return 0;
}

```

Python Code:

```

from collections import defaultdict, deque

class Solution:
    def canFinish(self, numCourses, prerequisites):
        hmap = defaultdict(list)
        q = deque()

```

```

inDegree = [0] * numCourses

# Put the mapping in the dictionary and calculate inDegree
for p in prerequisites:
    start, end = p[1], p[0]
    inDegree[end] += 1
    hmap[start].append(end)

for i in range(numCourses):
    if inDegree[i] == 0:
        q.append(i)

while q:
    removed = q.popleft()

    if removed in hmap:
        for r in hmap[removed]:
            inDegree[r] -= 1
            if inDegree[r] == 0:
                q.append(r)
    numCourses -= 1

return numCourses == 0

if __name__ == "__main__":
    numCourses = int(input())
    numPrerequisites = int(input())
    prerequisites = []

    for _ in range(numPrerequisites):
        course = list(map(int, input().split()))
        prerequisites.append(course)

    solution = Solution()
    canFinish = solution.canFinish(numCourses, prerequisites)

if canFinish:
    print("It is possible to finish all courses.")
else:
    print("It is not possible to finish all courses.")

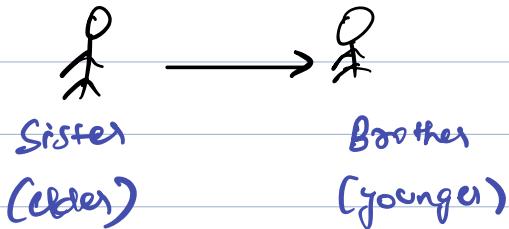
```

- ↳ **Introduction**
- ↳ **Output**
- ↳ **Operators**
- ↳ **Data types**

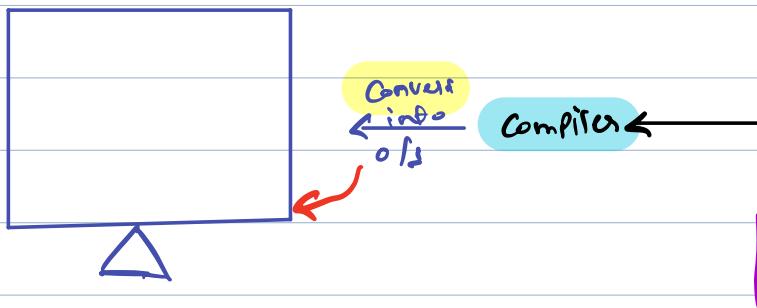
* Computer is? → Dumb

get me some

water!

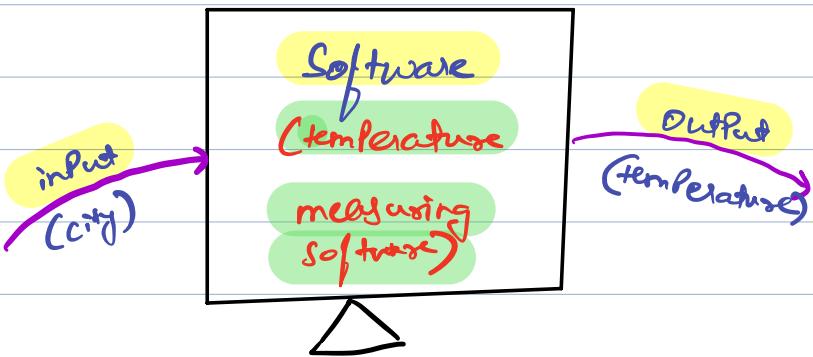


- a) get up from bed.
- b) go to kitchen
- c) get a glass
- d) fill glass with water
- e) bring the glass to me.



→ Can no we shall! → owned for english / grammar

for java as well → you need to follow rules.
↳ Syntax.



leetcode ide → google

writing code

- ① ↳ IDE: Integrated development environment?
 - ↳ Eclipse // IntelliJ // VSCode
 - install java.
- online editors

② Output Syntax/rules

↳ System.out.println(10);

↳ next line

↳ System.out.println(?);

↳ Press enter after printing

↳ System.out.print(?);

Ex:

System.out.print(7+10);
System.out.println(7*10);
System.out.println(10/5);

17 70

2
↓

→ Break till 9:15 PM

b) I want to print number as well as characters!

9 Hello?!

b) I want to do mathematical operation with characters

(10+9)
↳ 19Hello

* Operations

↳ + - * / → BODMAS

$$5 + 5 \div 5 \Rightarrow 6$$

↳ myth buster with SK Sir Part 1.

$$5 * 5 \div 5 \Rightarrow 5$$

BODMAS

Rank 1 : Bracket ()

Rank 2 : Divide / multiply *

↳ Computer will process right to left.

Rank 3 : add / subtract

$$\text{Ex-1: } 4 + \underbrace{3 * 6}_{\downarrow} - 7 / 2$$

$$4 + 18 - \underbrace{\underline{7 / 2}}_{\downarrow}$$

division in Computer will remove everything after decimal.

$$\underbrace{4 + 18 - 3}_{\downarrow}$$

$$22 - 3 = 19$$

* Data types

↳ Numbers, Char, double, boolean etc.
true false

Integer

decimal

ex: 10, 11, 100 etc ex: 5.8, 5.3, 100.2

Story:

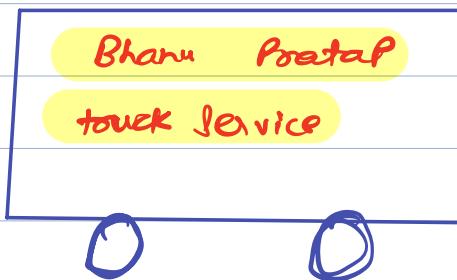
A

B

1 washing machine → Small truck

1 bhk shift → mid size truck

3 bhk shift → large "



Name of Service: Bhanu Pratap

Type of Service: Pockets & more

Quantity of Container: Small / Mid / Large

↳ Store → integer

Name of Service → integer

Type of Service → Store

Quantity → $10^1 / 10^2 / 10^3$

||



10 point xx

10 Store

↓ type ↓ name
int temp;
temp = 10; temp = 100;

Ex: int temp;

temp = 20;

→ temp = 100;

System.out.println(temp); → 100

temp
↳ 20 100

Ex: int temp;

temp = 20;

System.out.println(temp); → 20

temp = 100;

→ System.out.println(temp); → 100

temp
↳ 100

Creating integer

1st way

int temp;
→ declaration

temp = 20;
↳ initialization

2nd way

int temp = 20;

↳ both declare &

initialize in same line

```
public class Main {
    public static void main(String[] args) {

        //Rule1: Terminate your line with semicolon -> comments
        //System.out.println(7);

        //Rule2: Java is a case sensitive lang
        //System.out.println(7);

        //Rule3: Inside the parentheses, we can do maths.
        //        System.out.print(7+10);
        //        System.out.println(7*10);

        //        System.out.println(10/5);

        //Rule4 -> Use double quote to print as it is
        //System.out.println(Hello);
        // System.out.println("Hello");
        // System.out.println("10+17");
        //System.out.println("Hello SK Sir");

        // How does System.out.println(); works?
        // -> In the coming classes we will learn it.

        //Rule5 -> Concatenate different type with + while printing

        // System.out.println(10+9+"Hello");
        // System.out.println(100+9+15+"Hello");
    }
}
```

```
-- // Rule6-> Operators -> BODMAS
-- // System.out.println(6+6/6);
-- // System.out.println(4+3*6-7/2);

-- //Data Type
-- // int temp;
-- // temp = 20;
-- // System.out.println(temp);
-- // |
-- // int temp;
-- // temp = 20;
-- // System.out.println(temp);

-- //Rule -> You can't create same named variable twice.
-- //           int temp;
-- //           temp = 20;

-- //           int temp;
-- //           temp = 100;
-- //           System.out.println(temp);

-- //           int temp;
-- //           temp = 20;
-- //           // int temp;
-- //           temp = 100;
-- //           System.out.println(temp);

-- //           int temp;
-- //           // temp = 20;
-- //           // System.out.println(temp);
-- //           // temp = 100;
```