① . Given N, factorial of N.

N = 5

5! = 5 * 4 * 3 * 2 * 1

N! = N * (N-1) * (N-2) 1

```
int fact = 0;
for (int i = 1; i <= N; i++) {
    fact = fact * i;
}
sop (fact);
```

fact = 0 * 1 * 2 * 3

```
int fact = 1;
for (int i = 0; i <= N; i++) {
    fact = fact * i;
}
sop (fact);
```

N = 3

fact = 1 * 0 * 1 * 2 * 3

```
int fact = 1
for (int i = 1; i <= N; i++) {
    fact = fact * N;
}
sop (fact);
```

N^N

✓

```
int fact = 1
for (int i = 1; i <= N; i++) {
    fact = fact * i;
}
sop (fact);
```

N = 4

fact	i	
1	1	
1	2	
2	3	
6	4	
24	5	break

(2). Given n and r, find $^{n}C_{r}$.

$$^{n}C_{r} = \frac{n!}{(n-r)! * r!}$$

```
int fact1 = 1 ;  // n!
for (int i = 1; i <= n; i++) {
        fact1 = fact1 * i;
}

int fact2 = 1 ;  // (n - r)!
for (int i = 1; i <= (n-r); i++) {
        fact2 = fact2 * i;
}

int fact3 = 1 ;  // r!
for (int i = 1; i <= r ; i++) {
        fact3 = fact3 * i;
}

int ans = (fact1) / (fact2 * fact3);
```

Problems:

(i) Repeatation of code

(ii) Error prone

(iii) difficult to manage

↓

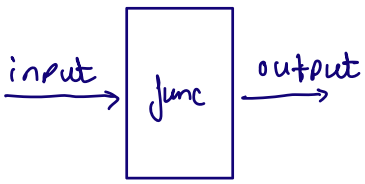
functions

ans ↵ $^{n}C_{r}$

Functions :

$f(x) = y$ output

f → function, x → input



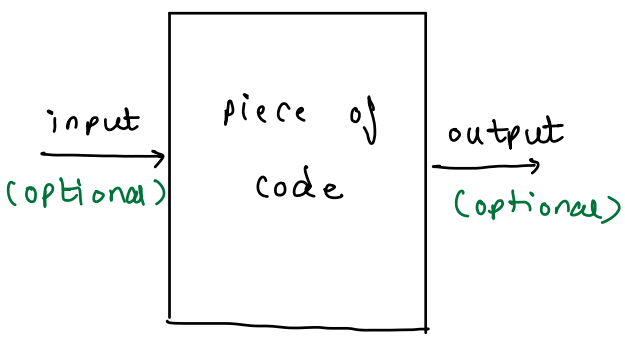
$sq(x) = x * x$

$sq(5) = 25$

$sum(a,b) = a+b$

$sum(5,10) = 15$

CS functions

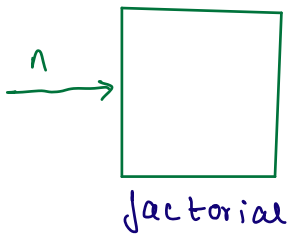



```
return_type function_name(input) {
    // code
    ____
    ____
    ____

    return _ ?
}
```

Structure of function

Given N, write a function that prints N!



return_type name input

```
void factorial (int n) {
        int fact = 1;
        for(int i=1; i<=n; i++) {
                fact = fact * i;
        }
        SOP(fact);
}
```

Given N, write a function that returns N!

```
int factorial (int N) {
        int fact = 1,
        for(int i=1; i<=n; i++) {
                fact = fact * i;
        }
        return fact;
}
```

```
public class Main {

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

    static int factorial(int n) {
        int fact = 1;
        for(int i=1; i<=n; i++) {
            fact = fact * i;
        }
        return fact;
    }
}
```

execution starts from main function.

$$^{n}C_{r} = \frac{n!}{(n-r)!\,r!}$$

$n = 4,\ r = 3$

$$^{4}C_{3} = \frac{4!}{1! * 3!} = \frac{24}{6} = 4$$

$n = 4,\ r = 2$

$$^{4}C_{2} = \frac{4!}{2! * 2!} = \frac{24}{2 \times 2} = 6.$$

3. Given two numbers, write a function that returns the sum of two no.

quiz

```
void main () {
    int a = 15;
    int b = 5;
    sum (a, b);
}
int sum (int a, int b) {
    return a+b;
}
```

```
sop (sum (a, b));

int ans = sum (a,b);
sop (ans);
```

quiz

```
void main () {
    int a = 15;
    int b = 5;
    sop(sum (a, 10));
}
int sum (int x, int y) {
    return x+y;
}
```

x = 15 y = 10

```
void main ( ) {
    int a = 15;
    int b = 5;
    SOP(sum (x, y ));
}

int sum (int x, int y ) {
    return x+y;
}
```

Compilation Error in main

quiz

```
static int square (int x) {
    return x * x;
}
static int add (int x, int y) {
    return x+y;
}
static void main ( ) {
    int n1 = square(3);
    int n2 = square (5);
    SOP( add (n1, square (9));
}
```

n1 = 9, square(9) = 81

9+81 = 90

Q. Given a N, check whether it is even or odd.

(i) Print "Even" or "Odd"

```
void EvenORodd (int N) {
    if (N%2 == 0) {
        SOP ("Even");
    }
    else {
        SOP ("odd");
    }
}
```

(ii) return true if no. is even else return false

```
boolean EvenORodd (int N) {
    if (N%2 == 0) {
        return true;
    }
    return false;
}
```

A functions stops execution when :

→ there is return statement.

→ closing bracket of that functions.

```
boolean fun ( ) {
    1. return true;
    2. SOP ("hello guys");
}
```

compilation error:

line 2 is unreachable

Doubts

a = 8 b = 12

min = minimum(8,12)

min = 8

```
for( int i = 8; i >= 1; i--) {
    if ( a % i == 0 && b % i == 0) {
        return i;
    }
}
```

a = 8

b = 12

i

8

7

6

5

4

3 x

2

1

9, 10, 11, 12

gcd(a,b) → 1

gcd(a,b) → min(a,b)

1.

0	0	0	0	5	0	0	0	0
0	0	0	4	8	12	0	0	0
0	0	3	6	9	12	15	0	0
0	2	4	6	8	10	12	14	0
1	2	3	4	5	6	7	8	9

i	jz	cnt no	dz	stv
1	4	1	4	5
2	3	3	3	4
3	2	5	2	3
4	1	7	1	2
5	0	9	0	1

(n-i) (n-i) (n-i+1)

N = 5

0	0	0	0	1	0	0	0	0
0	0	0	2	3	2	0	0	0
0	0	3	4	5	4	3	0	0
0	4	5	6	7	6	5	4	0
5	6	7	8	9	8	7	6	5

left table

i	zeros (n-i)	cnt (i)	sv
1	4	1	1
2	3	2	2
3	2	3	3
4	1	4	4
5	0	5	5

right table

i	count (i-1)	sv	dz
1	0	0	4
2	1	2	3
3	2	4	2
4	3	6	1
5	4	8	0