factorial ⇒ $5! = 5 * 4 * 3 * 2 * 1$

$0! = 1$

arrange n distinct items

$n! = n * (n-1) * (n-2) * (n-3) * \ldots * 1$

$7! = 5040$

A B C
A C B
B A C
B C A
C A B
C B A

$3! = 6$

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

$5!$

ans = 1

i=1 ans = 1 * 1 = 1

i=2 ans = 1 * 2 = 2

i=3 ans = 2 * 3 = 6

i=4 ans = 6 * 4 = 24

i=5 ans = 24 * 5 = 120

^nC_r = no of ways in which you can choose r item from n items.

↳ selection

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

n, r

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

find $n!$, $(n-r)!$, $r!$

N = 15 players ⇓ 11 players

A B C D

$^4C_2$

⇒ AB
AC CD
AD
BC
BD

$\frac{n!}{r!(n-r)!}$

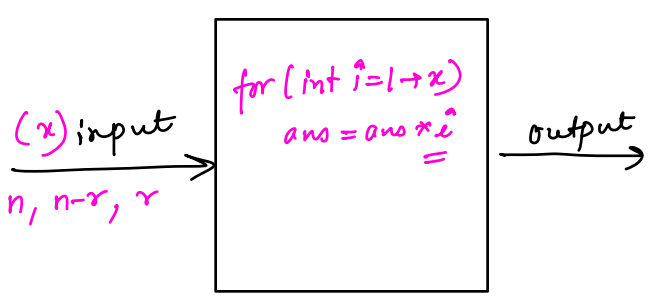
```
int fact_n = 1;          // n!

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

$x!$

```
int fact_r = 1;          // r'

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

- duplication
- error prone
- lengthy

```
int fact_nr = 1;

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

ans = fact_n / ( fact_r * fact_nr );
```



$f(x) = x^2$

$f(x) = x/2$

$f(x) = x!$

name

$f(x,y) = x^2 + xy$

functions / methods:- block of code which takes some input (optional) & gives some output & it is run again & again.

as many inputs = no >=0

```
return_type function_name ( inputs ) {
```

return_type → what kind of data is expected as output

inputs → what type of inputs

```
    ========
    ========
    ========
    ========

    return ____;
}
```

return → it returns your output

$f(x) = x^2 + 5x + 3$

parameters / arguments

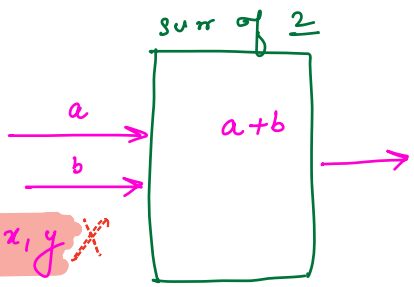
```
int fact ( int n ) {
    int ans = 1;
    for( int i = 1; i <= n; i++)
      {  ans = ans * i; }
    return ans;
}

main ( ) {
    // your code goes here
    int x = fact ( 9 );
}
```

calling of fn →

input → find its fact → =

sum of two numbers

2 variables which will store the inputs




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

```
_____ main ( ){
    - - - ,
    int a = sc.nextInt();
    int b = sc.nextInt();

    int res = sum ( a , b );
    SOP(result);
}
```

---

que 1

```
int a = 15, b = 5 ;        - unused ✓
SOP( sum(10, 6));
```

```
System.out.print ( sum( 10, 6));
```

fn

16

quiz

```
int sum( int a, int b){
    SOP( a+b);
}
```

expecting an int

→ no return statement

Error

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

print the sum

```
void sum( int a, int b)
{
    SOP(a+b);
}
```

not returning anything

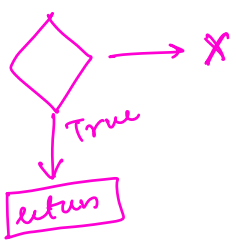
number, if it is even — please write a f^n for that.

```
bool isEven ( int n)
{
    if ( n%2==0)
    { return true; }
}
```

(17)

1) error
2) won't work for odd

→ compile → messy return statement



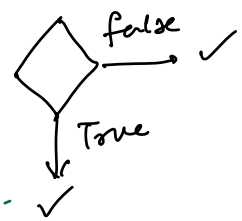
compile = run

code → run

compile → syntax ✓

```
bool isEven ( int n)
{
    if ( n%2==0)
    { return true; }
    else
    { return false; }
}
```

✓



✓

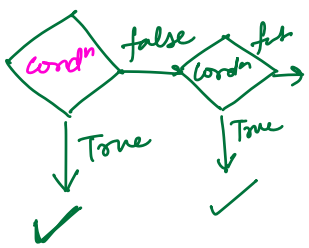
return ⇒ terminates the f^n execution

```
bool isEven ( int n)
{
    if ( n%2==0)
    { return true; }
    return false;
}
```

Q height of a tree

$h <= 6$ small → 1

$h > 6$ large → 0




```
int height ( int h)
{
    if ( h <= 6)
        return 1;
    else if ( h > 6)
        return 0;
}
```

a missing return statement

```
int height ( int h)
{
    if ( h <= 6)
        return 1;
    else
        return 0;
}
```