

⇒ Functions

- function is a piece of code
- function declaration :- body
- function calling :- where we piece of code
- main fn always called first
- Parameterised fn :- with para.
- Non-para fn :- without para.

⇒ Return Type

↳ which a function returns as a result

```
[ public static int fun1() {  
    return 0;  
}
```

```
[ Public static void main() {  
    fun1();  
}
```

ex-

{ int
 boolean
 String
 byte
 long
 ;
 arrays
 AL
 stack
 Queue--

void

⇒ return statement

- it returns back some result (optional)
- when we encounter return statement then the function get destroyed.

Note :-

- 1) break statement destroys the loop
- 2) return statement destroys the function

ex :-

void, int, boolean, string, ...

```
public class Main {  
    public static int sum(int a, int b) {  
        int s = (a + b);  
        return s;  
        // System.out.println(s);  
    }  
}
```

```
    public static void main(String[] args) {  
        int ans = sum(5, 6);  
        System.out.println(ans);  
    }  
}
```

```
}
```

Ex:-

```
public class Main {  
    public static boolean isIt5(int a) {  
        if ( a == 5 ) {  
            return true;  
        } else {  
            return false;  
        }  
    }  
  
    public static void main(String[] args) {  
        boolean ans = isIt5(6);  
        System.out.println(ans);  
    }  
}
```

Find product of the two numbers using function.

```
public static void main(String[] args) {  
    Scanner scn = new Scanner(System.in);  
    int t = scn.nextInt();  
  
    for (int i = 0; i < t; i++) {  
        int a = scn.nextInt();  
        int b = scn.nextInt();  
  
        → int ans = findProduct(a, b);  
        System.out.println(ans);  
    }  
}
```

```
public static int findProduct(int a, int b) {  
    int prod = a * b;  
    return prod; ←  
}
```

Swap x and y

process / logic

✓
int x = 5;
int y = 6; i/p

✓
int x = 6;
int y = 5; o/p

temp
int temp = x ; 5
x = y ; 5
y = temp ; 5

x	y
5	6

6	6
---	---

6	5
---	---

Swap x and y

Code

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

    swap(x, y);
}

public static void swap(int x, int y) {
    → int c = x;
    System.out.println("c = " + c);

    → x = y;
    System.out.println("x = " + x);

    → y = c;
    System.out.println("y = " + y);

    System.out.println("x = " + x);
    System.out.println("y = " + y);
}
```


Swap x y z

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

    swap(x, y, z);
}

public static void swap(int x, int y, int z) {
    → int temp = z;
    → z = y;
    → y = x;
    → x = temp;
}

System.out.println(x);
System.out.println(y);
System.out.println(z);
}
```

gmp

	(10)	(20)	(30)
temp	x	y	z
30	10	20	30
30	10	20	20
30	10	10	20
30	30	10	20

Factorial of N

$$n! = n \times (n-1) \times (n-2) \times \dots \times 3 \times 2 \times 1$$

$$6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1$$

code

```
public static void main(String[] args) {  
    → Scanner scn = new Scanner(System.in);  
    → int n = scn.nextInt();  
  
    → int ans = findFactorial(n);  
       System.out.println(ans);  
}  
public static int findFactorial(int n) {  
    → int ans = 1;  
    [ for (int i = n; i >= 1; i--) {  
        ans = ans * i;  
    }  
    → return ans;  
}
```

dry run

n = 6

ans = 1;

i = 6, ans = 6

i = 5, ans = 6 × 5

i = 4, ans = 6 × 5 × 4

i = 3, ans = 6 × 5 × 4 × 3

i = 2, ans = 6 × 5 × 4 × 3 × 2

i = 1, ans = 6 × 5 × 4 × 3 × 2 × 1

Find nCr .



$(r < n)$

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

$${}^{10}C_7 = \frac{10!}{3! \times 7!} = \frac{10 \times 9 \times 8 \times \cancel{7!}}{\cancel{7!} \times 3 \times 2 \times 1}$$

Code

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

    if ( r > n ) {
        return 0;
    }

    int ans = solve(n, r);
    System.out.println(ans);
}

public static int solve(int n, int r) {
    int a = fact(n);
    int b = fact(n - r);
    int c = fact(r);
    int ans = (a / (b * c));
    return ans;
}

public static int fact(int a) {
    int ans = 1;
    for (int i = a; i >= 1; i--) {
        ans = ans * i;
    }
    return ans;
}
```

↳ Digit Traversal

$$\begin{array}{c} n / 10 \\ n \% 10 \end{array}$$