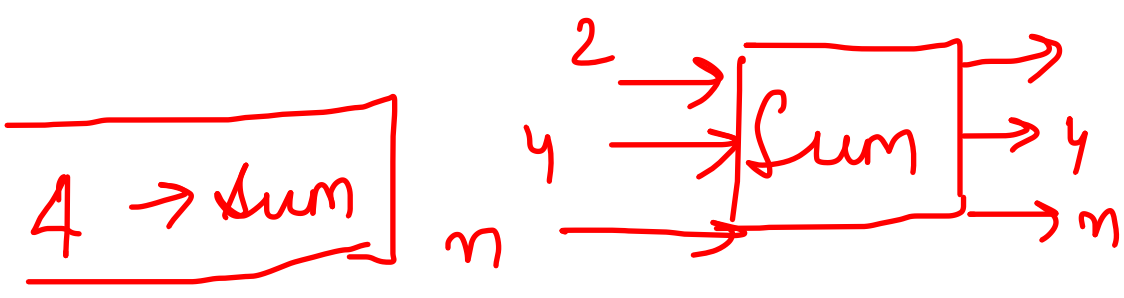
Functions → its a piece of code which can be re-used.

Why functions ? → Repetation → Readibility



=

sum

|

=

sum

|

5 lines → 20 lines → wfo

5 lines →

→ char → char

→ long → long

→ Int → int

→ Double → double

→ boolean → boolean

→ short → short

→ float → float

→ String → string

→ Array → int[]

void

↓

null.

System.out.println()

Integer → Integer

Int _ - -

void → no return statement
only sysout()

others → return
↳ gives you o/p
→ destroy the function

one can have many return statements

Syntax.

→ Declaration
→ calling

Access specifier → public
→ private
→ protected

Syntax to declare.

public static return type fun-name(parameters){

// statements.

return ;

}

error → missing return type.

parameters → optional

↳ parameterized func.

↳ non-parameterized fun.

parameters → i/p of the func.

→ data type should be defined.

Call → func_name (arguments)

Calling a func. → Sum(a, b)

collecting a func. → int ans = sum(a, b)

int → int

char → char

boolean → boolean

array → int []

int [] arr

char ch

arguments → o/ps.

→ don't define data type.

Parameter

→ during func. declare

→ define data type

→ a, b

→ No. of para & argu should be same. & order should be same.

Argument

→ during func. call

→ no data type defined.

→ a, b.

→ variable that has taken i/p

x, y. sum(a, b) ✗

sum(x, y) ✓

function which is called → callee (sum)

function which is calling → caller (main)

Code always start compiling from main

func. → factorial

^nP_r | ^nC_r

✓ ✓

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

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

arr →

largest → 9 Integer.MIN_VALUE

SL → 8 Integer.MIN.VALUE

arr[i] > SL && arr[i] != larg

1 > SL && T

T

4 > 1 && 4 < 9

3 > 4 && 3 < 9

f T

f

lar = 4

9

[SL < larg.]

>

SL → 4 8

8 > 4 && 8 < 9

6 > 8 && 6 < 9

f T

9 > 8 && 9 < 9

T F f

```java
public static int findsecondLargest(int arr[]){
    int n = arr.length;

    if(n <= 1){
        return Integer.MIN_VALUE;
    }

    int largest = Integer.MIN_VALUE;
    int secondLargest = Integer.MIN_VALUE;

    for(int i =0; i <n;i++){
        if(arr[i] > largest){
            secondLargest = largest;
            largest = arr[i];
        }
        else if(arr[i] > secondLargest && arr[i] != largest){
            secondLargest = arr[i];
        }
    }
    // if all the values are same ie. no SL found
    if(secondLargest == Integer.MIN_VALUE){
        return Integer.MIN_VALUE;
    }
    return secondLargest;
}
```

What will be the output of the following code ?

```
public static void func(int a, int b){
System.out.println(a + b);
}

public static void main(String[] args) {
int a = 7;
func(a, 12);
}
```

o/p → 19

What will be the output of the following Java code?

```
public static void demo(int a, int b){
   System.out.println(a + " " + b);
}

public static void main(String[] args) {
   int a = 5;
   int b = 15;
   demo(a);
}
```

o/p → error, argu. length

What should be return type of the following function:

```
public static ____ division(int a, int b)
{
   float c = a/b;
   return c;
}
```

o/p → float

Would the following code generate an error:

```
public static double add(int a,int b)
{
   float c=a+b;
   return c;
}
public static void main (String[] args) {
   System.out.print(add(10,3));
}
```

o/p → No, 13.0

Will the given code generate any error:

```
public static void func1(int a)
{
   System.out.print("a");
}
public static void main (String[] args) {
   func1(2.5);
}
```

o/p → a

What will be the output of the following code:

```
public static void func1(int a,int b)
{
   int ans=1;
   for(int i=0;i<b;i++)
   {
      ans*=a;
   }
   System.out.print(ans);
}
public static void main (String[] args) {
   func1(2,5);
}
```

o/p → 32

What will be the output of the following code ?

```
public static int square(int a){
   int ans = a * a;
   return ans;
}

public static void main(String[] args) {
   int a = 4;
   a = square(a);
   System.out.println(a);
}
```

o/p → 16

What will be the output of the following code:

```
public static int sum(int a,int b)
{
   System.out.print("int sum ");
   return a+b;
}
public static double sum(double a,double b)
{
   System.out.print("float sum ");
   return a+b;
}
public static void main (String[] args) {
   System.out.print(sum(5,4));
   System.out.print(sum(5.0,4.0));
}
```

o/p → int sum 9float sum 9.0

What will be the output of the following code:

```
public static int sum(int a,int b)
{
   System.out.print("int sum ");
   return a+b;
}
public static long sum(long a,long b)
{
   System.out.print("long sum ");
   return a+b;
}
public static void main (String[] args) {
      int a=4;
      int b=5;
   System.out.print(sum(a,b));
}
```

o/p → int sum 9

function overloading

4.4 double

[1 - 50 - int
1 - 100 - long]

[? int]

Will following code generate any error ?

```
public class Main {
   public static void func(int a) {
      int b = 10;
      a = a + 10;
      System.out.println(a);
   }

   public static void main(String[] args) {
      int a = 10;
      func(a);
      System.out.println(a);
   }
}
```

o/p → No