FUNCTIONS

Page No. :
Date : 08/01/23

functions / Methods

block of code - we can perform that particular thing by calling again and again.
[Reusable]

Syntax

Output Type

```
returnType name() {
  // body
  return statement;
}
```

```
import java.util.*;
public class JavaBasics {
  public static void printHelloWorld() {
    System.out.println("Hello World");
    System.out.println("Hello World");
    System.out.println("Hello World");
  }
```

③

```
  public static void ~~main~~ CalculateSum (String args[]) {
    Scanner sc = new Scanner(System.in);
    int a = sc.nextInt();
    int b = sc.nextInt();
    int sum = a+b;
    System.out.println("sum is : "+sum);
```

①

```
  public static void main(String args[]) {
    ② CalculateSum(); // call calculateSum function.
  }
}
```

Jun 26, 23

* Inverted half-Pyramid with numbers

```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

```
public static void inverted_half_pyramid_with_num(int n){
    for(int i=1; i<=n; i++){
        // inner loop
        for(int j=1; j<=n-i+1; j++){
            System.out.print(j);
        }
        System.out.println();
    }
}
```

- void → 'empty' int

return; return 1/2/3;

- Those function is written within the class, that function is called **method** of that class

Class के अंदर function → Method

Parameter → input

```
return Type name (type param1, type param2){
    //body
    return statement;
}
```

* Call by Value

Call by Reference
↓
original

- Java always call by value
  ↓
  copy of original

*

Methods

User defined

eg. { factorial, sum, product }

Inbuilt Method

eg. Math → pow, sqrt, max

Sc. nextInt

*

Function Overloading
↓
Multiple function with same name but different parameters

eg. { multiply (int a, int b)
multiply (float a, float b)
multiply (double a, double b) }

↓ using Parameters, → does not depend upon return Type.

only depend on parameters

f1: sum of 2 no.

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

f2: sum of 3 no.

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

Using data Types

f1: add 2 int val

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

f2: add 2 float val

```
float sum(float a, float b){
    return a+b;
}
```

Qus Check a given number is prime or not.

n → prime → 1, n.

for(int = 2 to int = n-1)

(n % i == 0) not prime

prime