JPL :: Modular Programming - 1

TalentSprint

Licensed To Skill

Version 1.0.4

Learning Objectives

By the end of this presentation, you are able to:

- Write programs to problems by decomposing functionality into methods and using the methods
- Write computationally efficient programs
- Create meaningful functional decomposition systematically
- Develop and test your programs progressively

Modular Programming - 1

Modular Programming - 1

```
public static void main(String[] args) {

    Reading responsibility ============>

    Arithmetic responsibility ==========>

    Perfect Number responsibility ======>

    Prime responsibility ===============>

    Factorial responsibility ===========>
}
```

Why?

Modular Programming - 1

Finding Factorial of a given number...

```
... void main(String[] args) {
    int i, n = 5;
    int factorial = 1;
    for (i = 1; i <= n; i++) {
        factorial = factorial * i;
    }
    System.out.print(factorial);
}
```

Before entering loop n = 5, factorial = 1

factorial	i	condition
1	1	1 <= 5
1	2	2 <= 5
2	3	3 <= 5
6	4	4 <= 5
24	5	5 <= 5
120	6	6 <= 5

Modular Programming - 1

Let us now learn more about methods in Java...

- A method is a set of instructions implemented in order to perform a specific task.
- All methods in Java must be defined inside some class. Main method is also part of a class.
- Methods are identified by their signature.
- Signature contains method name and the parameters it takes. Each method in Java should have four parts.
  - Return-type
  - Method-name
  - Parameters
  - Method body

Modular Programming - 1

Writing Methods

Syntax

```
<return-type> <method-name>(<parameters>) {
    statement(s);
}
```

Example

```
int factorial(int num) {
    return fact;
}
```

Modular Programming - 1

Method Name

- Name of the method can be any name.
- Name of the method should indicate what it does.
- As per the naming conventions, method name should be started with lower-case letter. If method name contains more than one word, then first letter of each word must be a upper-case from second word.
- Method name must be started with an alphabet.
- **Examples:** get(), getSquare()

Modular Programming - 1

Method Body

Body of method contains a logical sequence of instructions intended to perform some task.

```
int add(int firstNum, int secondNum) {
    – – – – –
    return firstNum + secondNum;
}
```

Modular Programming - 1

Parameters

- Parameters are like variable declarations and are local variables for the method.
- Parameters are inputs for the method to use for processing.

100, 200

```
add(100, 200);
```

```
int add(int firstNum, int secondNum) {
    – – – – –
    return firstNum + secondNum;
}
```

Modular Programming - 1

Parameters

A method can have any number of parameters including zero.

```
int add(int n1, int n2) {
    // 2 parameters
}
```

```
int add(float n1, float n2)
    {
    // 2 parameters
}
```

```
int add(int n1, int n2, int n3) {
    // 3 parameters
}
```

```
int add() {
    // No parameters
}
```

Modular Programming - 1

Return Type

It denotes type of value the method returns.

```
int add(int num1, int num2) {
    return value;
}

double sqrt(double num1) {
    return value;
}

boolean isPrime(int num1) {
    return value;
}
```