



## ***Java SE 8 Programming Language***

# **Lab Guides**


Document Code	25e-BM/HR/HDCV/FSOFT
Version	1.1
Effective Date	20/11/2012

**RECORD OF CHANGES**

No	Effective Date	Change Description	Reason	Reviewer	Approver
1	01/Oct/2018	Add the new labs	Create new	DieuNT1	VinhNV
2	01/Jun/2019	Update template	Fsoft template	DieuNT1	VinhNV

## Contents

Unit 6: Advance OOP .....	4
Lab Guide 1: Overload.....	4
Objectives:.....	4
Problem Descriptions:.....	4
Questions to answer: .....	4
Guidelines:.....	4

	CODE:	JPL.S.L301
	TYPE:	SHORT
	LOC:	68
	DURATION:	30 MINUTES

## Unit 6: Advance OOP

### Lab Guide 1: Overload

#### Objectives: JPL-9

- Able to create Java-based applications that take advantage of Java object-oriented features, including encapsulation, inheritance, and polymorphism.

#### Problem Descriptions:

Create a new package named **fa.training.methodoverloading** in **JPL.S.L301** project.

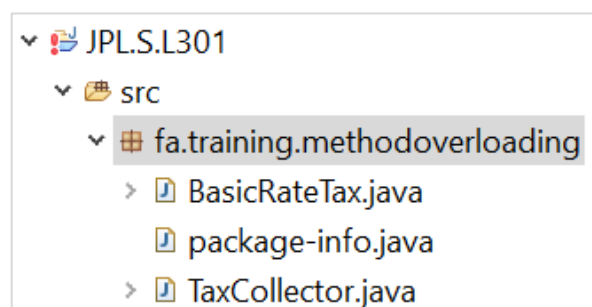
- Create a **BasicRateTax** class with a method **calcTax()** that returns 20% of a fixed base income of £1000.
- Create a java program named **TaxCollector** that creates a new **BasicRateTax** object, calls the **calcTax()** method and prints the output to the console.
  - Run the **TaxCollector** program and ensure it always prints 200.00 as calculated tax.
- Add new **calcTax()** method to **BasicRateTax** class that takes a double *grossIncome* parameter and calculates the tax as 20% of the grossIncome if it's greater than the base income of £1000
- Change the **TaxCollector** program to call the new **calcTax(double grossIncome)** method and passing the gross Income value from the command line.
  - Run the **TaxCollector** program and see if the tax is correctly calculated.
  - Re-run the program with different Gross Income values and check the output.

#### Questions to answer:

- a. How many ways to overload a method? List out invalid case of method overloading?
- b. Question 1: Explain in detail the meaning of **code line 9, 10** in **BasicRateTax** class?
- c. Question 2: Describe about piece of code **Double.parseDouble(args[0])**, what is it used for?

#### Guidelines:

Project struture:



- **BasicRateTax** class:

```
1. package fa.training.methodoverloading;
2.
3. /**
4.  *
5.  * @author DieuNT1
6.  *
7.  */
8. public class BasicRateTax {
9.     private static final double BASE_INCOME = 1000.00;
10.    private static final double BASIC_TAX_RATE = 0.20;
11.
12.    /**
13.     * This method calculates fixed base income.
14.     *
15.     * @return returns 20% of a fixed base income of £1000.
16.     */
17.    public double calcTax() {
18.        return BASE_INCOME * BASIC_TAX_RATE;
19.    }
20.
21.    /**
22.     * calculates the tax as 20% of the grossIncome.
23.     *
24.     * @param grossIncome
25.     * @return returns 20% of the grossIncome if it's greater than the base income
26.     *         of £1000.
27.     */
28.    public double calcTax(double grossIncome) {
29.        if (grossIncome < BASE_INCOME) {
30.            return calcTax();
31.        }
32.        return grossIncome * BASIC_TAX_RATE;
33.    }
34. }
35.
```

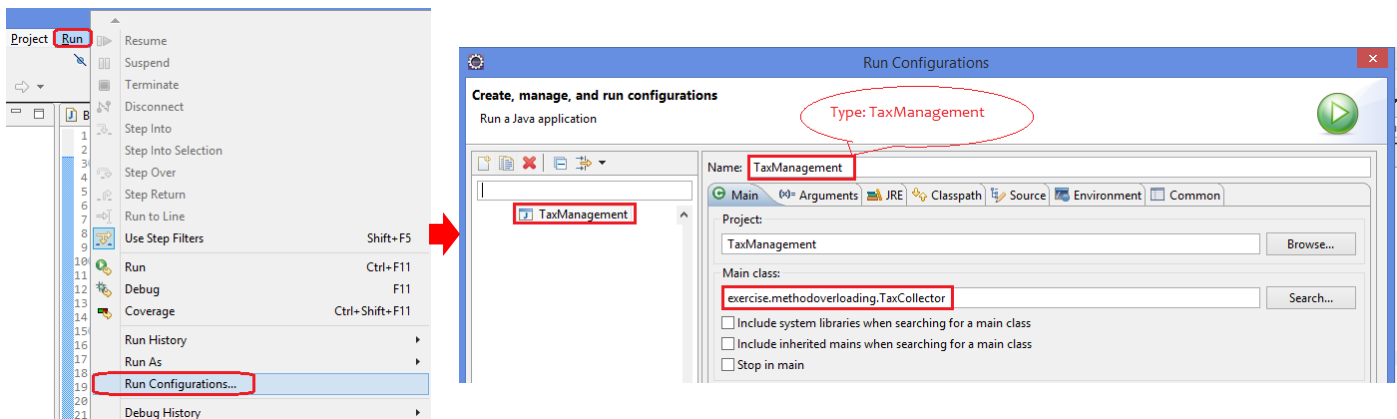
- **TaxCollector** class

```
1. package fa.training.methodoverloading;
2.
3. /**
4.  * This class contains the main method to run app.
5.  *
6.  * @author DieuNT1
7.  *
8.  */
9. public class TaxCollector {
10.    /**
11.     * The main method.
12.     *
13.     * @param args
14.     */
15.    public static void main(String[] args) {
16.        /*
17.         * The grossIncome value gets from the first argument of main.
18.         */
19.        double grossIncome = Double.parseDouble(args[0]);
20.        /*
21.         * Create a new BasicRateTax object named: taxCalculator.
```

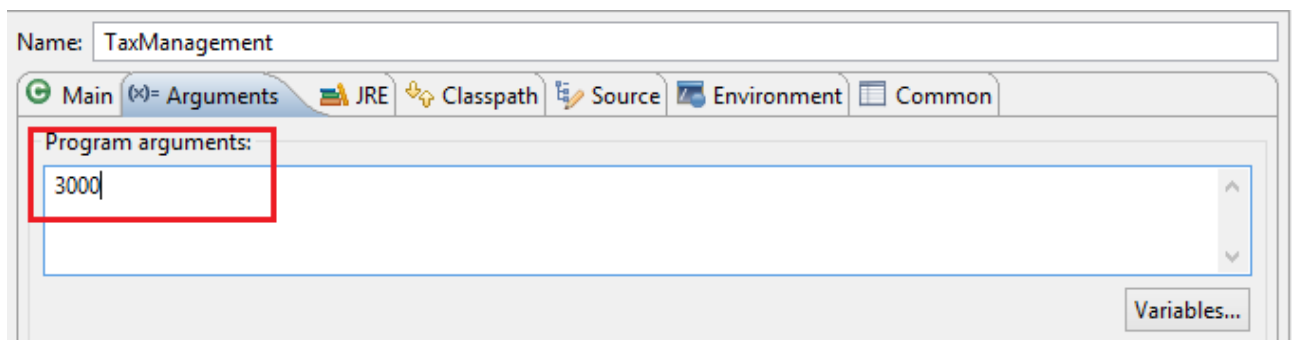
```
22.    */
23.    BasicRateTax taxCalculator = new BasicRateTax();
24.    /*
25.     * Call object's calcTax method.
26.     */
27.    double tax = taxCalculator.calcTax(grossIncome);
28.
29.    /*
30.     * Print out result.
31.     */
32.    System.out.println("Tax due is " + tax);
33. }
34. }
35.
36.
```

- How to run:

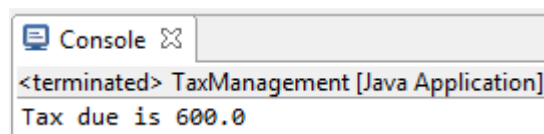
Click **Run** menu | choose **Run Configurations**:



Enter **grossIncome** value at **Program arguments** (example: 3000) | click **Run** button:



**Result:**



-- THE END --