Lab 1

Developing and Using Methods

Objectives

Upon completion of this lab, you should be able to:

- Complete review questions
- Create methods and use the methods in a Java program
- Use method overloading in a Java program

Lab Overview

In this lab, you complete the review questions and two exercises.

- In the first exercise, you create methods and invoke them.
- In the second exercise, you create overloaded methods and use them.

Completing Review Questions

Complete the following questions:

1. State which of the following statements are true:
   a. A class can only contain one single method declaration.
   b. The most basic form of a method is one that has a return type but no arguments.
   c. All calling methods can be worker methods and all worker methods can be calling methods.
   d. Overloaded methods have the same number of arguments.
2. The main method accepts which of the following argument types:
   a. A `String` object
   b. An `int` type
   c. An array of references to `String` objects
   d. An array of `String` objects
3. Which method corresponds to the following method call?

```
myPerson.printValues (100, 147.7F, "lavender");
```

   a.
```
public void printValues (int pantSize, float
ageInYears)
```
   b.
```
public void printValues (pantSize, float
ageInYears, favoriteColor)
```
   c.
```
public void printValues (int pantSize, float
ageInYears, String favoriteColor)
```
   d.
```
public void printValues (float ageInYears, String
favoriteColor, int pantSize)
```

Exercise 1: Using Arguments and Return Values

The objective of this exercise is to write a class with a method that invokes a worker method in another class.

In this exercise, you create a small application that instantiates three shirt objects and applies a different price to each.

Then the application instantiates a purchase receipt object that reports the total price as each shirt is added to the order.

Preparation

Ensure that the `Order.java` file and `the Shirt.java` file exists in the `SL110/exercises/08_methods/exercise1` directory. This is your working directory.

Task – Writing a Method That Uses Arguments and Return Values

In this task, you write a test class that adds multiple `Shirt` objects to an `Order` object and displays a total current amount, in dollars, for the order. Follow these steps to write your test class:

1. Go to the working directory and open an editor.
2. Open the `Shirt.java` file and examine the member variables and the method in it.
3. Open the `Order.java` file and examine its member variables and method in it.
4. Write a new class called `OrderTest` containing a `main` method.
5. In the `main` method:
   a. Create and initialize an object of type `Order` and an object of type `Shirt`. The `Order` class (`Order.java`) and the `Shirt` class (`Shirt.java`) are provided in the exercise1 directory.
   b. Declare a variable of type double, name it as `totalCost` and initialize it by 0.0.
   c. Assign the `price` for the `Shirt` object to 14.00.

d. Invoke the `addShirt` method of the `Order` class using the `Order` instance to add the `shirt` instance to the order. Store the return value of the `addShirt` method in the totalCost variable. The documentation for the `addShirt` method is as follows:

```
public double addShirt (Shirt s)
```

Adds a shirt to a list of shirts in an order

Parameters:

`s` – An object reference to a `Shirt` object

Returns:

A total current amount for the order

e. Display the return order amount. For example:

```
Total amount for the order is: 14.00
```

6. Compile and execute your program. Verify if the total order value is displayed.
7. In the main method of `OrderTest` class, create additional `Shirt` objects, assign values to the `price` variable of the new `Shirt` objects, and add the `Shirt` objects to your order by invoking the `addShirt` method.
8. Save and compile the `OrderTest.java` file.
9. Execute `OrderTest` class and verify that the total order value is displayed correctly.

Exercise 2: Using Overloaded Methods

The objective of this exercise is to implement method overloading.

In this exercise, you create overloaded methods in a class and invoke the methods and test it.

Preparation

Ensure that `CustomerTest.java` file exists in the `SL110/exercises/08_methods/exercise2` directory. This is your working directory.

Task – Developing a Class With an Overloaded Method

In this task, you write a `Customer` class with an overloaded method called `setCustomerInfo`.

Follow these steps to write your class:

1. Go to the `working` directory and open an editor.
2. Create a class called `Customer` and save the file as `Customer.java` in the working directory.
3. Within the `Customer` class, add two overloaded methods called `setCustomerInfo`.

   Depending on how the `setCustomerInfo` method is called, it does one of the following:

   - Sets the ID, name, address, and phone number for a `Customer` object. (This is the minimum information needed for a new Customer.)
   - Sets the ID, name, address, phone number, and email address for a `Customer` object.
4. Create a `display` method to display the values of all the member variables of the `Customer` class.
5. Save and close the `Customer.java` file.
6. Open the file called `CustomerTest`.java to test the overloaded methods of the `Customer` class.

7. In the `main` method of `CustomerTest` class write code to perform the following tasks:
   a. Create two object references to different `Customer` objects.
   b. Use each variation of the `setCustomerInfo` method to provide information for each `Customer` object.
   c. Display the contents of each `Customer` object.
8. Save and compile the file `CustomerTest.java` file.
9. Execute the `CustomerTest` class and view the output of the class.

Exercise Summary

Take a few minutes to identify what experiences, issues, or discoveries you had during the lab exercises.

- Experiences
- Interpretations
- Conclusions
- Applications