# OODP Workshop 7

Modularizing (or breaking down) of code into methods is used so that code becomes reusable, easier to read and also easier to debug.

1. Find definitions for the following terms as they relate to Java programming:
   1. Parameter

Parameters are variables declared in a method header that receive values from arguments when the method is called. They define the input data required for the method to execute its logic. For example, in addValues(int x, int y), x and y are parameters.

* 1. Method

A method is a reusable block of code that performs a specific task. It consists of a method header (including modifiers, return type, name, and parameters) and a body containing executable statements. Methods enable code modularization and abstraction.

* 1. Return value

The return value is the data a method sends back to the calling code after execution. It is specified by the return type in the method header (e.g., int, void) and returned using the return keyword. A method with void returns nothing.

* 1. Method signature

The method signature combines the method name and parameter list (types and order). It uniquely identifies a method for overloading. For example, addValues(int, int) is the signature of a method taking two integers.

1. Consider the following method:

public static int addValues(int x, int y){

System.out.println(x + y);

return (x + y);

}

* 1. What parameters does the method have?

The method has two parameters: int x and int y.

\_\_

* 1. What are appropriate values for the arguments when this method is called?

Valid arguments are integers (e.g., 5, -3, or integer variables like num1).\_\_\_\_\_\_\_

* 1. What is return value?

The return value is the sum of x and y as an integer.\_\_

* 1. Provide one example of how you might call this method

**int** sum = addValues(10, 20); System.out.println("Sum: " + sum);

\_

1. Consider the following method:

public static void addValues(int x, int y){

System.out.println(x + y);

}

* 1. What parameters does the method have?

The method has two parameters: int x and int y.\_\_\_

* 1. What are appropriate values for the arguments when this method is called?

Valid arguments are integers (e.g., 15, 25, or variables storing integers).

* 1. What is return value?

This method has no return value (void return type).\_\_\_\_\_\_\_

* 1. Provide one example of how you might call this method

addValues(30, 40); *// Directly prints the sum*\_\_\_\_\_\_\_\_

1. Practice modularization

**Objective of this activity:- converting given code to modularised code**

Open menu.java and do following tasks:

* Create a method to display menu that will display the menu which is given in file and ask to enter their selection and return that selection to calling method. Call this method in main method where there is a need to print menu.
* Move the switch case to a separate method which will take menuSel as input and print necessary details. In case 1, ask user to enter their name and age, in case 2, print the details of user and case 3, terminate the program.

Ans:

package lab7;

import java.util.Scanner;

public class Menu {

// Method to display menu and return user selection

public static int displayMenu() {

Scanner in = new Scanner(System.in);

System.out.println("Menu");

System.out.println("1. Enter data");

System.out.println("2. Display data");

System.out.println("3. Exit");

System.out.print("Enter selection: ");

int selection = in.nextInt();

return selection;

}

// Method to process menu selection

public static void processMenuSelection(int menuSel) {

Scanner in = new Scanner(System.in);

switch (menuSel) {

case 1:

System.out.print("Enter your name: ");

String name = in.nextLine();

System.out.print("Enter your age: ");

int age = in.nextInt();

System.out.println("Data saved successfully.");

break;

case 2:

System.out.println("Displaying user details...");

// Add logic to display stored data

break;

case 3:

System.out.println("Exiting...");

System.exit(0);

break;

default:

System.out.println("Invalid selection!");

}

}

public static void main(String[] args) {

int selection = displayMenu();

processMenuSelection(selection);

}

}

