

Basic Programming Practicum Job Sheet 11 Meeting 15



From:

AL AZHAR RIZQI RIFA'I FIRDAUS

Class:

1 I

Absence:

01

Major:

Information Technology

Study Program:

Informatic Engineering

Experiment 1

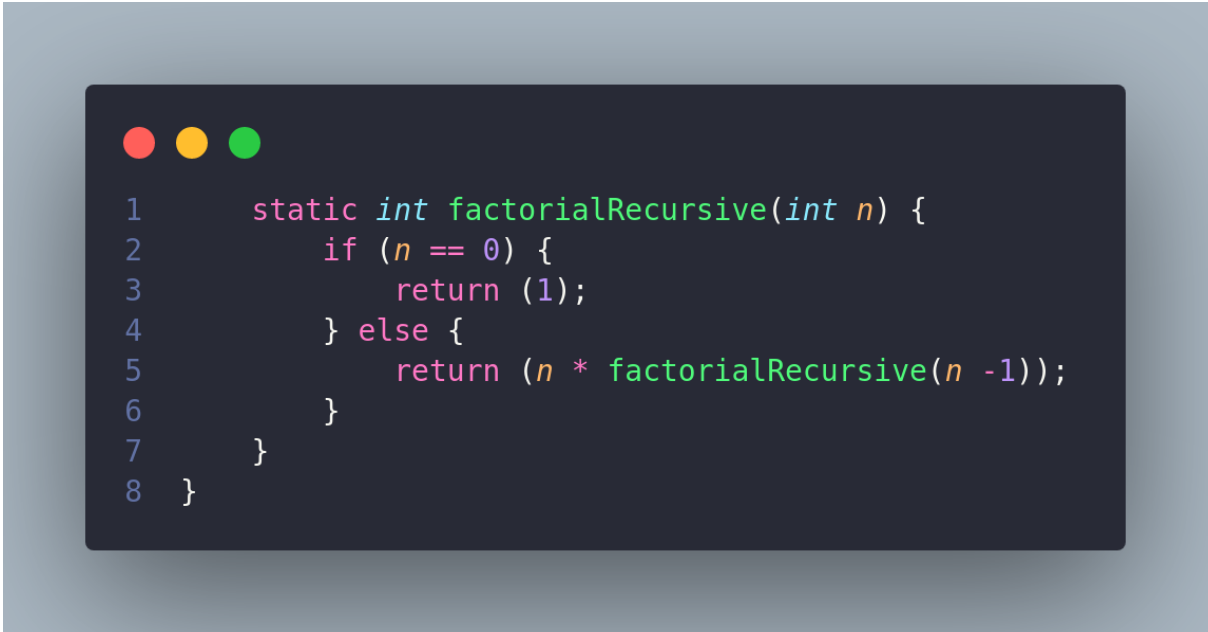
In this experiment, a program will be created to calculate the factorial value of a number using a recursive function. In addition, a function for calculating factorial values will be made using an iterative algorithm as a comparison.

1. Create a new project
2. Create a new class, name it Experiment1




```
1 public class Experiment1 {  
2  
3 }
```

3. Create a static function with the name factorialRecursive(), with return data type that is int and has 1 parameter with int data type in the form of a number calculated for its factorial value.



```
1 static int factorialRecursive(int n) {  
2     if (n == 0) {  
3         return (1);  
4     } else {  
5         return (n * factorialRecursive(n - 1));  
6     }  
7 }  
8 }
```

4. Create another static function with the name factorialIterative(), with return data type that is int and has 1 parameter with int data type in the form of a number calculated for its factorial value.




```

1  static int factorialIterative(int n) {
2      int factor = 1;
3      for (int i = n; i >= 1 ; i--) {
4          factor = factor * i;
5      }
6      return factor;
7  }

```

5. Create a main function and make a call to the two previously created functions, and display the results obtained.

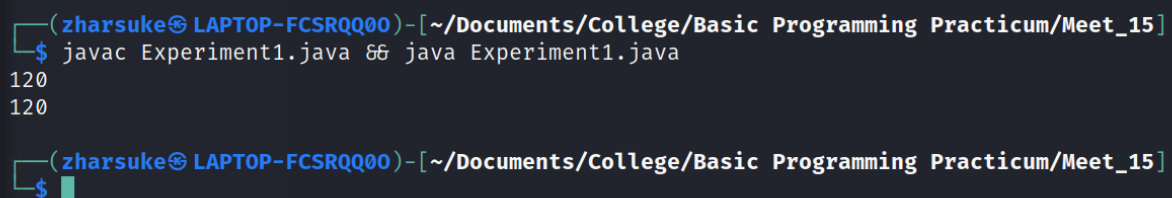


```

1  public static void main(String[] args) {
2      System.out.println(factorialRecursive(5));
3      System.out.println(factorialIterative(5));
4  }

```

6. Compile and run the program.



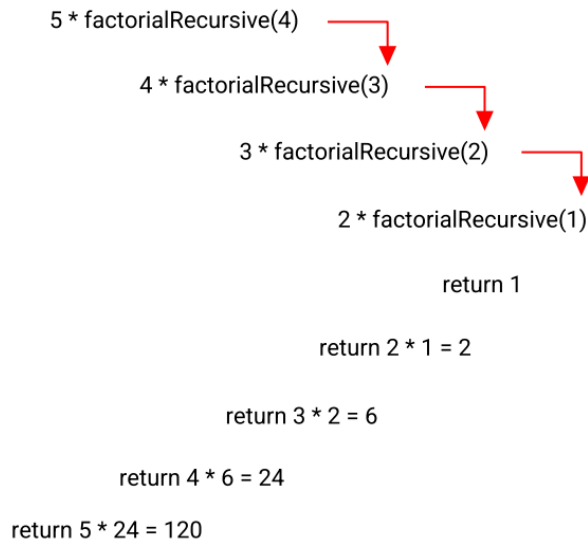
```

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac Experiment1.java && java Experiment1.java
120
120

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ █

```

7. If traced, when calling the factorialRecursive(5) function, the process that occurs can be illustrated as follows:




Experiment 2

In this experiment, a program will be created to calculate the power of a number using a recursive function.

1. Create a new class, name it Experiment2


```
1 public class Experiment2 {  
2  
3 }
```

2. Create a static function with the name `calculatePower()`, with return data type that is `int` and has 2 parameter with `int` data type in the form of numbers to be calculated and the exponents.



```
1 static int calculatePower(int x, int y) {
2     if (y == 0) {
3         return (1);
4     } else {
5         return (x * calculatePower(x, y - 1));
6     }
7 }
```

3. Create a main function and declare a Scanner with the name sc



```
1 public static void main(String[] args) {
2     Scanner sc = new Scanner(System.in);
3 }
```

4. Create two variables of type int with name number and exponent



```
1  int number, exponent;
```

5. Add the following code to accept input from the keyboard



```
1  System.out.print("Enter a number : ");  
2  number = sc.nextInt();  
3  System.out.print("Enter the exponent : ");  
4  exponent = sc.nextInt();
```

6. Call the calculatePower function that was created previously by sending two parameter values.



```
1  System.out.println(calculatePower(number, exponent));
```

7. Compile and run the program.

```
(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac Experiment2.java && java Experiment2.java
Enter a number : 2
Enter the exponent : 3
8
(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$
```

Experiment 3


In this experiment, a program will be created to calculate the amount of customer money deposited in the bank after earning interest for several years using the recursive function.

1. Create a new class, name it Experiment3




```
1 public class Experiment3 {
2
3 }
```

2. Create a static function with the name calculateInterest(), with return data type that is double and has 2 parameter with int data type int in the form of customer balance and time spent saving. In this case, it is assumed that the interest set by the bank is 11% annually. Because the interest calculation is interest * balance, so to calculate the amount of money after adding interest is balance + interest * balance. In this case, the interest rate is 0.11 * balance, and the balance is considered 1 * balance, so 1*balance + 0.11 * balance can be summarized into 1.11 * balance for calculating the balance after adding interest (in a year).




```
1 static double calculateInterest(double balance, int year) {  
2     if (year == 0) {  
3         return (balance);  
4     } else {  
5         return (1.11 * calculateInterest(balance, year - 1));  
6     }  
7 }
```

3. Create a main function and declare a Scanner with the name sc




```
1 public static void main(String[] args) {  
2     Scanner sc = new Scanner(System.in);  
3 }
```

4. Create a variable of type double named openingBalance and a variable of type int named year




```
1  double openingBalance;  
2  int year;
```

5. Add the following code to accept input from the keyboard



```
1  System.out.print("Enter the opening balance : ");  
2  openingBalance = sc.nextDouble();  
3  System.out.print("Enter the duration of saving (years) : ");  
4  year = sc.nextInt();
```

6. Call the calculateInterest function that was created previously by sending two parameter values.



```
1  System.out.print("Amount of money after " + year + " years : ");  
2  System.out.println((int) calculateInterest(openingBalance, year));
```

7. Compile and run the program.

```
(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac Experiment3.java && java Experiment3.java
Enter the opening balance : 10000
Enter the duration of saving (years) : 20
Amount of money after 20 years : 80623

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ █
```

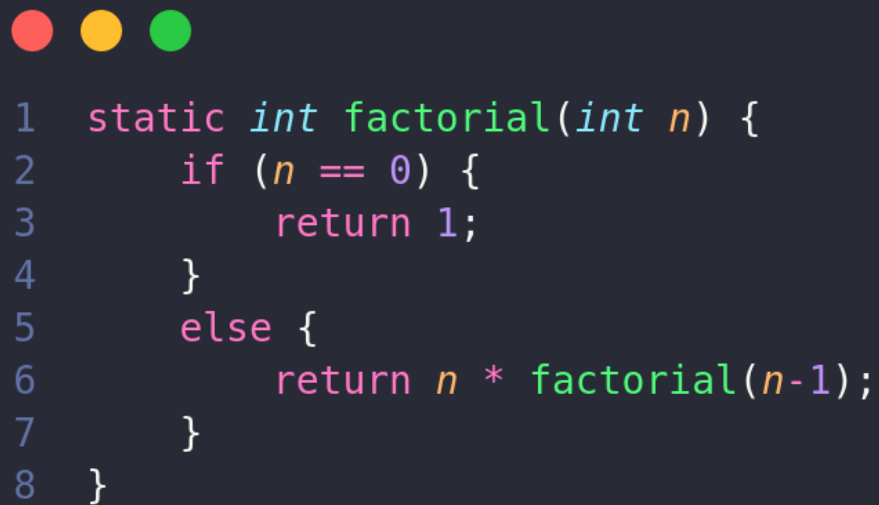
Questions!

1. What is a recursive function?
2. What are the examples of recursive functions?
3. In Experiment 1, do the factorialRecursive() function and factorialIterative() function give the same result? Explain the difference in the flow of the program in the use of recursive functions and iterative functions!
4. In Experiment 2, there is a recursive function call calculatePower(number, exponent) in the main function, then the calculatePower() function calls are repeated. Explain how long the function calling process will be executed!
5. In Experiment 3, state which program code block is the "base case" and "recursion call"!

Answer

1. A recursive function is a function that calls itself in order to solve a problem.
2. The example is factorial function. The factorial of a non-negative integer n , denoted as $n!$, is the product of all the positive integers from 1 to n . For example, the factorial of 5 is $5! = 5 * 4 * 3 * 2 * 1 = 120$.

code :

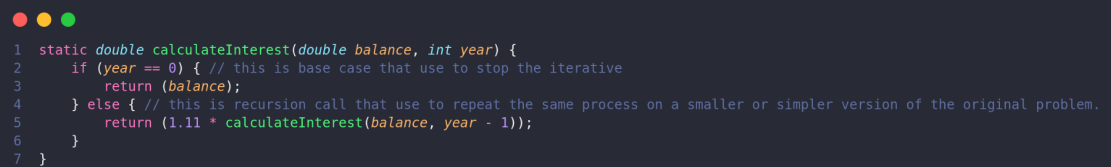


```

1  static int factorial(int n) {
2      if (n == 0) {
3          return 1;
4      }
5      else {
6          return n * factorial(n-1);
7      }
8  }

```

3. Both are different. factorialRecursive() using recursive function to iterative. Then factorialIterative() using for loop to iterative.
4. function calculatePower() will be executed y times.
5. Code :



```

1  static double calculateInterest(double balance, int year) {
2      if (year == 0) { // this is base case that use to stop the iterative
3          return (balance);
4      } else { // this is recursion call that use to repeat the same process on a smaller or simpler version of the original problem.
5          return (1.11 * calculateInterest(balance, year - 1));
6      }
7  }

```

Assignment

1. Create a program to display the numbers n through 0 using recursive and iterative functions. (RecursiveDescendingSeries).

Code :

```

1  import java.util.*;
2  public class assnum1 {
3      public static void main(String[] args) {
4          Scanner scanner = new Scanner(System.in);
5          int number;
6          System.out.print("Insert number : ");
7          number = scanner.nextInt();
8          displayNumber(number);
9          scanner.close();
10     }
11     private static int displayNumber(int number) {
12         if (number == 0) {
13             System.out.println(number + " ");
14             return 0;
15         } else {
16             System.out.println(number + " ");
17             return displayNumber(number-1);
18         }
19     }
20 }
21

```

Result :

```

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac assnum1.java && java assnum1.java
Insert number : 5
5
4
3
2
1
0
(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$

```

2. Create a program that includes a recursive function for calculating factorial numbers. For example f = 8, it will produce 1 + 2 + 3 + 4 + 5 + 6 + 7+8=36(RecursiveAddition).

Code :

```

1  import java.util.*;
2  public class assnum2 {
3      public static void main(String[] args) {
4          Scanner scanner = new Scanner(System.in);
5          int number;
6          System.out.print("Insert number : ");
7          number = scanner.nextInt();
8          System.out.printf("Factorial number of %d = %d\n", number, factorialNumber(number));
9          scanner.close();
10     }
11     private static int factorialNumber(int number) {
12         if (number == 0) {
13             return 0;
14         } else {
15             return number + factorialNumber(number-1);
16         }
17     }
18 }
19

```

Result :

```

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac assnum2.java && java assnum2.java
Insert number : 8
Factorial number of 8 = 36

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ █

```

3. Create a program that includes a recursive function to check whether a number n is a prime number or not. n is said to be not a prime number if it is evenly divided by a number less than n. (RecursivePrimaCheck).

Code :

```

1  import java.util.*;
2  public class assnum3 {
3      public static void main(String[] args) {
4          Scanner scanner = new Scanner(System.in);
5          int number, i = 2;
6          System.out.print("Insert number : ");
7          number = scanner.nextInt();
8          boolean isPrime = primeChecker(number, i);
9          if (isPrime) {
10             System.out.println(number + " is prime number!");
11         } else {
12             System.out.println(number + " is not a prime number!");
13         }
14         scanner.close();
15     }
16     private static boolean primeChecker(int number, int i) {
17         if (i == number){
18             return true;
19         } else if (number % i == 0) {
20             return false;
21         } else {
22             return primeChecker(number, i + 1);
23         }
24     }
25 }
26

```

Result :

```

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ javac assnum3.java && java assnum3.java
Insert number : 17
17 is prime number!

(zharsuke@LAPTOP-FCSRQQ00)-[~/Documents/College/Basic Programming Practicum/Meet_15]
$ █

```