JOBSHEET 14

Recursive Function

Name
Sherly Lutfi Azkiah Sulistyawati

NIM
2341720241

Class
1I

Department
Information Technology

Study Program
D4 Informatics Engineering

Labs Activity

Question! (Experiment 1)

1. After observing the experiment of recursive function above, what is the definition of recursive function?
2. How the recursive function works?
3. From the experiment above, do factorialRecursive() and factorialIterative() have the similar result? Then, what are the differences between recursive and iterative if both are having the same result?

Answer!

1. A recursive function is a function that calls itself in its own definition. In other words, it's a function that solves a problem by solving smaller instances of the same problem. Recursive functions typically have a base case to terminate the recursion and prevent an infinite loop.
2. When a recursive function is called, it breaks down the problem into smaller sub-problems, each closer to a base case. The function calls itself with these smaller sub-problems until it reaches the base case, at which point the recursion stops, and the results are combined to solve the original problem.
3. Yes, factorialRecursive() and factorialIterative() produce the same result - calculating the factorial of a given number.
   The main differences are:
   - Recursive functions break down the problem into smaller subproblems, while iterative functions use loops to solve the problem iteratively.
   - Recursive functions place additional load on the call stack as they push new stack frames with each recursion. Iterative solutions do not use the call stack in this way.
   - Iterative solutions are generally more efficient and faster than recursive solutions.
   - Recursive solutions can be simpler to write and understand if the problem can be broken down recursively.

Question! (Experiment 2)

1. In Experiment 2, there is a recursive function call calculatePower() in the main function, then the function calculatePower() is called repeatedly. Explain how long the function calling process will run!

2. Add program code to print the power calculation series. Example: calculatePower(2,5) will print 2x2x2x2x2x1 = 32!

Answer!

1. - The calculatePower() function is a recursive function that calculates the power of a given base.
   - The function makes recursive calls, decrementing the power (pow) by 1 in each call until the base case is reached (pow == 0).
   - The function will continue to make recursive calls until it reaches the base case, at which point the recursion stops.
   - The duration of the function calling process depends on the value of the input power (Power). The function will make Power recursive calls before reaching the base case.

2. 

Question! (Experiment 3)

1. From the above experiment, which statements that is classified as “base case” and “recursion call”!
2. Explain using simulation or trace the expansion phase and substitution phase of

Answer!

1. In the calculateProfit function:
   - Base Case: if (period == 0), where the function returns the current balance when the investment period becomes zero.

- Recursion Call: return 1.11 * calculateProfit(balance, period - 1), where the function calls itself with a decremented investment period.

2. Let's trace the expansion and substitution phases for calculateProfit(100000, 3):

   - Expansion Phase:

   calculateProfit(100000, 3)

   Returns 1.11 * calculateProfit(100000, 2)

   Returns 1.11 * calculateProfit(100000, 1)

   Returns 1.11 * calculateProfit(100000, 0)

   Returns 100000 (base case)

   - Substitution Phase:

   Substitute the results back into the original expression:

   calculateProfit(100000, 3) = 1.11 * calculateProfit(100000, 2)

   Substitute calculateProfit(100000, 2):

   calculateProfit(100000, 3) = 1.11 * (1.11 * calculateProfit(100000, 1))

   Substitute calculateProfit(100000, 1):

   calculateProfit(100000, 3) = 1.11 * (1.11 * (1.11 * calculateProfit(100000, 0)))

   Substitute calculateProfit(100000, 0):

   calculateProfit(100000, 3) = 1.11 * (1.11 * (1.11 * 100000))

   - Calculation:

   Calculate the final result:

   calculateProfit(100000, 3) ≈ 136763.31

Assignment

1. Write a program to display numbers n to 0 using recursive functions and iterative functions. (DescendingSequenceRecursive).
2. Create a program to sum the numbers using recursive function. For example n = 8, then it will result 1+2+3+4+5+6+7+8 = 36 (SummationRecursive).

3. Create a program that contains a recursive function to check whether a number n is a prime number or not. A number will not be classified as a prime number if it is divisible by a number less than n. (PrimeCheckingRecursive).

4. A pair of newborn guinea pigs (male and female) are placed in a nursery. After two months the guinea pig pair gave birth to a pair of twin guinea pigs (male and female). Every pair of guinea pigs that is born will also give birth to a pair of guinea pigs every 2 months. How many pairs of guinea pigs were there at the end of the 12th month? Write a program using a recursive function! (Fibonacci). And the following table illustrates the calculation.

Month	Pair Number		Pair Total
	Productive	Non-Productive	
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8
7	5	8	13
8	8	13	21
9	13	21	34
10	21	34	55
11	34	55	89
12	55	89	144

Answer!

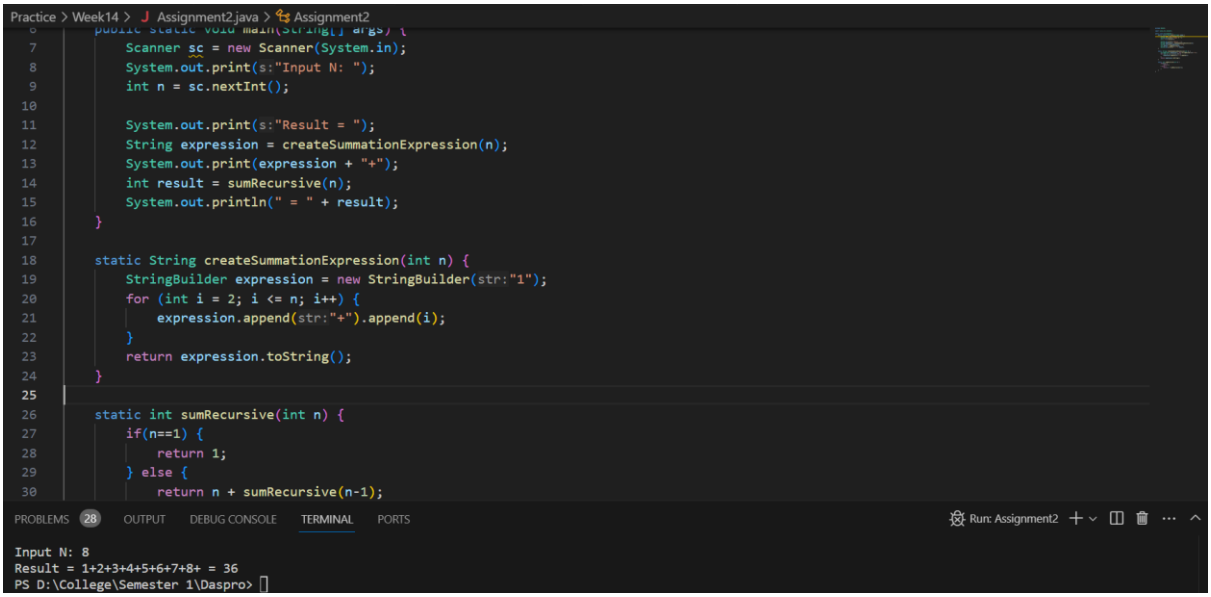
1.

Practice > Week14 > Assignment1.java > Assignment1

```java
package Week14;

public class Assignment1 {
    Run | Debug
    public static void main(String[] args) {
        descendingSeqRecursive(n:5);
    }

    static void descendingSeqRecursive(int n) {
        if(n==0) {
            System.out.print(s:"0\n");
        } else {
            System.out.print(n+" ");
            descendingSeqRecursive(n-1);
        }
    }

}
```

```
PROBLEMS 26  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS        Run: Assignment1
347c835eb98c8\redhat.java\jdt_ws\Daspro_e24281ec\bin' 'Week14.Assignment1'
5 4 3 2 1 0
PS D:\College\Semester 1\Daspro>
```

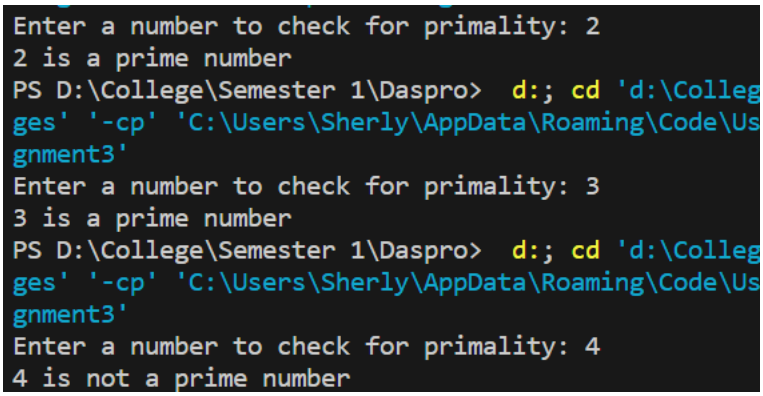
2.



```
Practice > Week14 > Assignment2.java > Assignment2
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Input N: ");
        int n = sc.nextInt();

        System.out.print("Result = ");
        String expression = createSummationExpression(n);
        System.out.print(expression + "+");
        int result = sumRecursive(n);
        System.out.println(" = " + result);
    }

    static String createSummationExpression(int n) {
        StringBuilder expression = new StringBuilder("1");
        for (int i = 2; i <= n; i++) {
            expression.append("+").append(i);
        }
        return expression.toString();
    }

    static int sumRecursive(int n) {
        if(n==1) {
            return 1;
        } else {
            return n + sumRecursive(n-1);
        }
    }

PROBLEMS 28  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS        Run: Assignment2

Input N: 8
Result = 1+2+3+4+5+6+7+8+ = 36
PS D:\College\Semester 1\Daspro>
```

3.

```
Practice > Week14 > Assignment3.java > Assignment3 > main(String[])
public class Assignment3 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number to check for primality: ");
        int number = sc.nextInt();

        if (isPrime(number, number - 1)) {
            System.out.println(number + " is a prime number");
        } else {
            System.out.println(number + " is not a prime number");
        }
    }

    static boolean isPrime(int n, int divisor) {
        if (n <= 1) {
            return false;
        }
        if (divisor == 1) {
            return true;
        }

        if (n % divisor == 0) {
            return false;
        } else {
            return isPrime(n, divisor - 1);
        }
    }
}
```



```
Enter a number to check for primality: 2
2 is a prime number
PS D:\College\Semester 1\Daspro>  d:; cd 'd:\Colleg
ges' '-cp' 'C:\Users\Sherly\AppData\Roaming\Code\Us
gnment3'
Enter a number to check for primality: 3
3 is a prime number
PS D:\College\Semester 1\Daspro>  d:; cd 'd:\Colleg
ges' '-cp' 'C:\Users\Sherly\AppData\Roaming\Code\Us
gnment3'
Enter a number to check for primality: 4
4 is not a prime number
```

4.

```java
public static void main(String[] args) {
    int months = 12;

    int[] productivePairs = new int[months];
    int[] nonProductivePairs = new int[months];
    int[] totalPairs = new int[months];

    //initial values
    productivePairs[0] = 0;
    nonProductivePairs[0] = 1;
    totalPairs[0] = 1;

    //values for the second month
    productivePairs[1] = 0;
    nonProductivePairs[1] = 1;
    totalPairs[1] = 1;

    System.out.println(x:"Month\tProductive\tNon-Productive\tPair Total");

    for (int i = 2; i < months; i++) {
        productivePairs[i] = nonProductivePairs[i - 1];
        nonProductivePairs[i] = productivePairs[i - 1] + nonProductivePairs[i - 1];
        totalPairs[i] = totalPairs[i - 1] + productivePairs[i];
    }

    for (int i = 0; i < months; i++) {
        System.out.println((i + 1) + "\t" + productivePairs[i] + "\t\t" + nonProductivePairs[i] + "\t\t" + totalPairs[i]);
    }
}
```

Month	Productive	Non-Productive	Pair Total
1	0	1	1
2	0	1	1
3	1	1	2
4	1	2	3
5	2	3	5
6	3	5	8
7	5	8	13
8	8	13	21
9	13	21	34
10	21	34	55
11	34	55	89
12	55	89	144