EX1. Computing Perfect Numbers in an Interval using Kotlin

Student Amirhossein Zeinali Dehaghani (std ID: 1225496)
Professor Dr. Sabah Mohammed
Lakehead University

Algorithm Idea

The algorithm to find the perfect numbers between 1 and 10,000 as follows:

Function Definition

The core function, `isPerfectNumber(number: Int): Boolean`, checks whether a given integer number is perfect. It initializes a variable, `sumOfFactors`, to add up all of the number's appropriate divisors.

Finding Divisors using Recursion function:

Since no divisor can be greater than half of the number, the algorithm iterates through all integers from 1 up to half of the number.

During each iteration, it checks if the current integer *i* is a divisor of `number` using the modulus operator (%).

- If *i* is a divisor (i.e., `number % i == 0`), it adds *i* to `sumOfFactors`.

Determine Perfectness

After the loop concludes, the function checks if the sum of the proper divisors (stored in `sumOfFactors`) equals the original number.

- If they are equal, it returns true, indicating that `number` is a perfect number; otherwise, it returns false.

Main Execution

In the main function, the program defines the range (1 to 10,000) and initializes an empty mutable list named `perfectNumbers` to store any perfect numbers found. The program loops through each integer in this range, calls the `isPerfectNumber` function, and if a number is perfect, it adds it to the `perfectNumbers` list.

Finally, it outputs all identified perfect numbers in a comma-separated format.

Output

When the program is executed, it will print the following perfect numbers found within the range of 1 to 10,000:

Figure 1: Screen Shot of the output