Exercise 1: Part 3

Loops and functions

Python Programming Bootcamp by Dr Rohitash Chandra
UNSW, 2021

Description

1. The following is a function that determines whether or not a given a number *n* is prime. (20 points)

```
isPrime = TRUE;

FOR primeFactor = 2 to n-1 DO
        IF n is divisible by primeFactor THEN
                isPrime = FALSE;

IF isPrime THEN
        Output “number n is prime”
        Return TRUE
ELSE
        Output “number n is not prime”
        Return FALSE
```

2. Write a function for finding the first *N* prime numbers. We already know how to test whether or not a number is prime. We will use this algorithm for finding the first *N* prime numbers as follows: (20 points)

```
Start with n=2;
Assume the number n is prime;
  WHILE we have not yet found N prime numbers DO
  {
        Test divisibility of n by all potential factors between 2 and
          The square root of n itself;
       IF n is divisible by any of these numbers THEN
                n it is not prime
          ELSE
        {
                n is prime;
                count it;
          }
         check next number n;
  }
```

3. Write a function that finds all prime numbers between a lower and upper limit. (20 points)

4. By the fundamental theorem of arithmetic, every positive integer greater than 1 has a unique prime factorization. However, the fundamental theorem of arithmetic gives no insight into how to obtain an integer's prime factorization; it only guarantees its existence.

Examples:

46 = 2 * 23 48 = 2 * 2 * 2 * 2 * 3 51 = 3 * 17
142 = 2 * 71 144 = 2 * 2 * 2 * 2 * 3 * 3

Write a function that computes the prime factorization of a given composite number entered by the user. (40 points)

Resources:

1. https://en.wikipedia.org/wiki/Prime_number
2. https://en.wikipedia.org/wiki/Fundamental_theorem_of_arithmetic

Acknowledgment

The assignment is adapted from exercise designed by Prof. Christian Omlin