Assignment 4 (5%)
Prime Numbers

Deadline: Friday 5 November 2021 at 23:59 on Submitty

Working individually, complete the assignment below. Submit your solution to Submitty (`https://submit.scss.tcd.ie`). By submitting your solution, you are confirming that you have familiarised yourself with College's policy on plagiarism (`https://libguides.tcd.ie/plagiarism`).

Your mark will be the auto-graded mark assigned by Submitty (10 marks) plus a manually assigned mark (5 marks) for programs that demonstrate excellent presentation, helpful, concise pseudo-code comments and a well-structured approach to solving the problem.

You are allowed to submit five attempts for the assignment without penalty. Subsequent attempts will attract a 1 mark penalty each, up to a maximum penalty of 5 marks.

Submitty will allow you eight "late days" over the full semester. This means, for example, you can submit one assignment late by eight days or eight assignments late by one day each, without penalty. Once your "late days" are used up, you will receive zero marks for any late submissions.

Instructions

Design and write an ARM Assembly Language program that will count the number of prime numbers that are less than a specified integer, N. The value of N will be provided in register `R1`. Your program should store the count of the number of primes in `R0`.

Your solution should include pseudo-code comments to explain your approach, using a Java-like syntax (see lecture 4.1, slide 6).

If you use the work of others to inspire your solution, you **must** cite your sources.

Hint 1: Begin by writing a program that will determine whether or not a number is a prime number. Once you are satisfied that this is working, you can extend the program to count the number of primes that are less than the specified value N.

Hint 2: The pseudo-code below describes one approach that you might use to determine whether a given number i is a prime number.

```
//
// Test if i is a prime number
//

divisor = 2;
remainder = 1; // force the while loop to execute at least once

while (divisor < i && remainder != 0) // "&&" means logical AND
{
    quotient = i / divisor;
    remainder = i - (quotient * divisor);
    divisor = divisor + 1;
}

if (remainder != 0)
{
    // i is a prime number
}
```

ARM microprocessors do not have a built-in way to calculate the remainder from a division. Instead, we need to calculate the quotient (whole part) and use that to calculate the remainder.

$$remainder = i - (quotient \times divisor)$$

Your can use the ARM `UDIV` instruction to perform a division. For example, `UDIV R4, R5, R6` will divide `R5` by `R6`, storing the while part of the result (the quotient) in `R4`.

The `while(...)` loop above keeps dividing `i` by larger divisors until a division produces a remainder of 0, meaning that `i` is not a prime number. If we get all the way to `i - 1` without a remainder of 0, then `i` is prime.

Note that you can improve on the efficiency of the above algorithm but this is left as an exercise for you.