Homework 1
Assigned: Wednesday, March 15
Due: Wednesday, March 22 at 5:00 PM

Programming Assignment #1 — Starting with C

Outcomes

After successfully completing this assignment, you should be able to:

- Write C code containing loops and if statements
- Implement functions in C
- Prompt the user for input values at the terminal

Before Starting

Review the following topics:

- Input and output to the terminal using `printf()` and `scanf()` in C
- Control statements in C
- Functions in C

Part 1: Write a C program that determines if a value is a prime number.

Your program will take an integer value from the user and determine if the value is prime number. Your output should match the example on Page 4.

Step 1: Read all instructions

Read every line of this document, including the grading rubric, before you start developing your code. Highlight key points and make sure you understand what is being asked of you.

Step 2: Read the provided pseudocode

Download the file Prime_pseudocode.txt from Canvas. Read this carefully and make notes about how you would convert this into C code.

Step 3: Transfer the provided pseudocode

Create the file `prime_number.c` in Visual Studio Code. Copy the provided pseudocode into this file as comments

Step 4: Write your code

Using the pseudocode as a guide, begin writing your algorithm in stages, starting with the first line of your pseudocode. Best programming practice is to write and test outer loops and conditional statements, before working on inner statements. This allows you to catch bugs early, debugging as you write, instead of dealing with all the errors after you have many lines of code written.

Implementation Notes

- Name your file `prime_number.c`
- Your code should contain at least 1 loop (for or while) and at least one if/else statement

Step 5: Test your code

Test your code locally before uploading it to Gradescope. Once you are confident in the results, upload to Gradescope and see if it passes the autograder. If not, read the output carefully and continue debugging your code.

Part 2: Print a list of prime numbers

Write a *C* program called `print_prime.c` that prints the first N prime numbers where N is specified by the user. You will be modifying the code from `prime_number.c` to contain a function called `is_prime()` and print the first N prime numbers.

Your code **must** contain:

- A function called called `is_prime()` that determines if a specific number is prime or not. The function should return 1 if the number is prime and 0 if otherwise.
  The function prototype for this function should be:
  ```
  int is_prime(int);
  ```
- A prompt asking the user how many numbers to print.
- A loop that prints the first N prime numbers

Step 1: Write Pseudocode

Begin by writing pseudocode that outlines the algorithm you will develop. In addition to submitting your working program, you **must** submit pseudocode outlining your algorithm.

Step 2: Implement your code

You can copy your entire program from Part 1 into a new file and call it `print_prime.c`. From there, work on modifying the code or writing it from scratch to meet the requirements listed above.

Step 3: Test your code

Test your code locally before uploading it to Gradescope. Once you are confident in the results, upload to Gradescope and see if it passes the autograder. If not, read the output carefully and continue debugging your code. Your output should match the example on Page 4.

Using your resources:

- If you consult web resources, or other students or staff when developing your program, *you must cite your source*. If you completed the entire program on your own, you should say in your write up file that this is entirely your work.
- If you borrowed some or all of an algorithm from someone else or from somewhere else, *do not copy it.* Write it out in your own words and your own coding style. Also, please explain enough about how the algorithm works that the graders can conclude that you understand it.

Deliverables:

1. Write a document called **README.txt** that
   - Summarizes the program for Part 2
   - Explains how to run `print_prime.c`, including the compile command
   - Detailing any problems that you had when developing the code
   - List anyone you worked with on the code and outside resources. If you worked alone, state that as well. If you used any outside resources for this assignment, be sure to cite your sources *and* explain in detail how the code works.
2. Write a document called **HW1_pseudocode.txt** that contains the pseudocode for Part 2 If you wrote your pseudocode on paper, you can take a picture and include it in this file.
3. `prime_number.c` and `print_prime.c`

Upload all 4 files to Canvas.

Grading Rubric:

Part 1:

- Autograder on Part 1	15 pts
- Contains a loop	10 pts
- Contains an if/else statement	10 pts

Part 2:

- Autograder on Part 2	15 pts
- Pseudocode	10 pts
- Correct implementation of `is_prime()`	10 pts

Other

- Satisfactory README file with all required parts	15 pts
- Proper indentation	15 pts

Deductions:

1. No comments on code	-10 pts
2. Poor indentation	-10 pts

Programs submitted after 5:00pm on due date (March 22nd) will be tagged as late and will be subject to the **Late Homework Policy**.

Example Output: (Values highlighted in green are user input)

Part 1:

Example 1:

```
Please enter a value:
52
This value is not a prime number.
```

Example 2:

```
Please enter a value:
47
This value is a prime number
```

Part 2:

```
How many prime numbers do you want?
10
2 3 5 7 11 13 17 19 23 29
```