CSC 1300-001 Programming Assignment 2: Standard Math Program

Objective

The major objective of this programming assignment is to demonstrate mastery of the following topics:

- Commenting in C++
- Formatting code in a readable way
- Basic Input and Output
- Math expressions
- Variables and data types
- Branching
- Loops (with continue and break statements)
- Functions

Description

The program will ask the user to input a positive integer, and then it will perform a set of mathematical tasks. Given the positive integer input (e.g., n), the program will (1) compute the sum of 1 to n, (2) compute the factorial value of n, (3) display the times table of n in a certain way, (4) check whether n is even or odd, (5) check whether or not n is a prime number, and (6) check whether or not n is an Armstrong number.

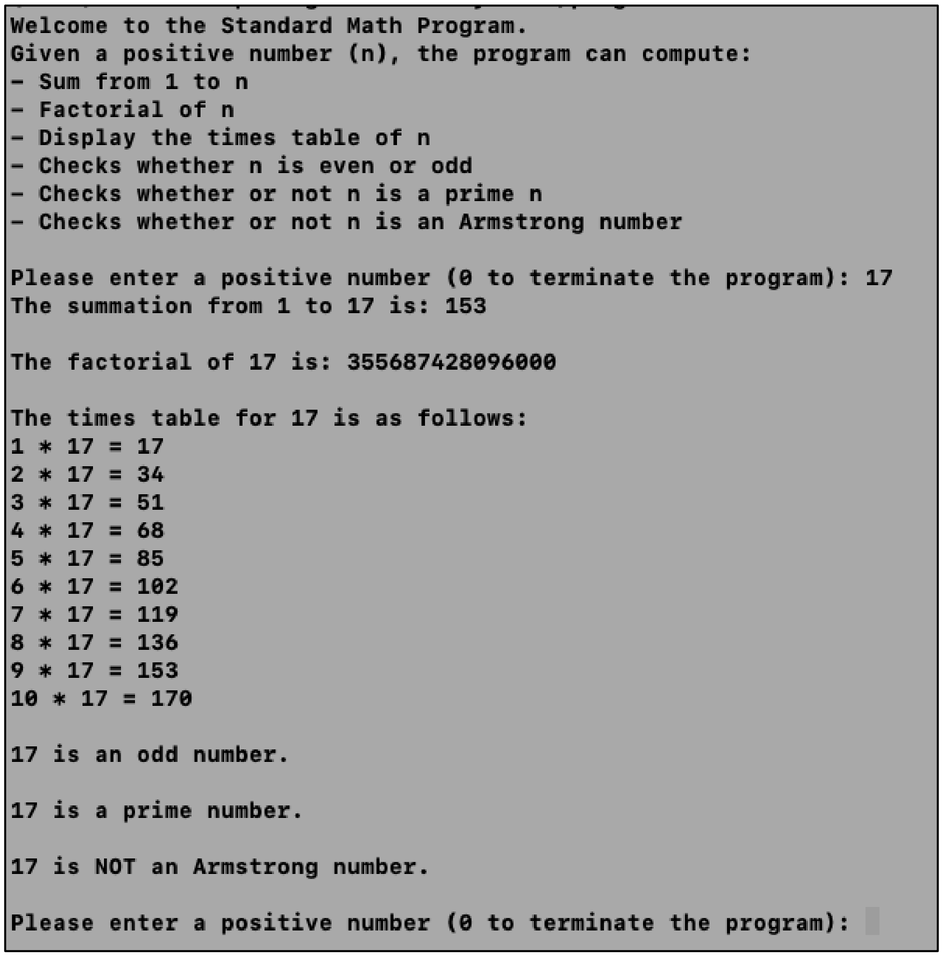
It is expected the program will continue the aforementioned operations for every positive integer; however, if and only if the user inputs 0, then the program should terminate itself. If the user inputs a negative number, then your program should not perform the aforementioned operations and ask the user to input a positive number instead to perform the tasks.

The following is a walkthrough of what I expect your program to do after execution –

1. Display the following message on screen at the beginning of your program

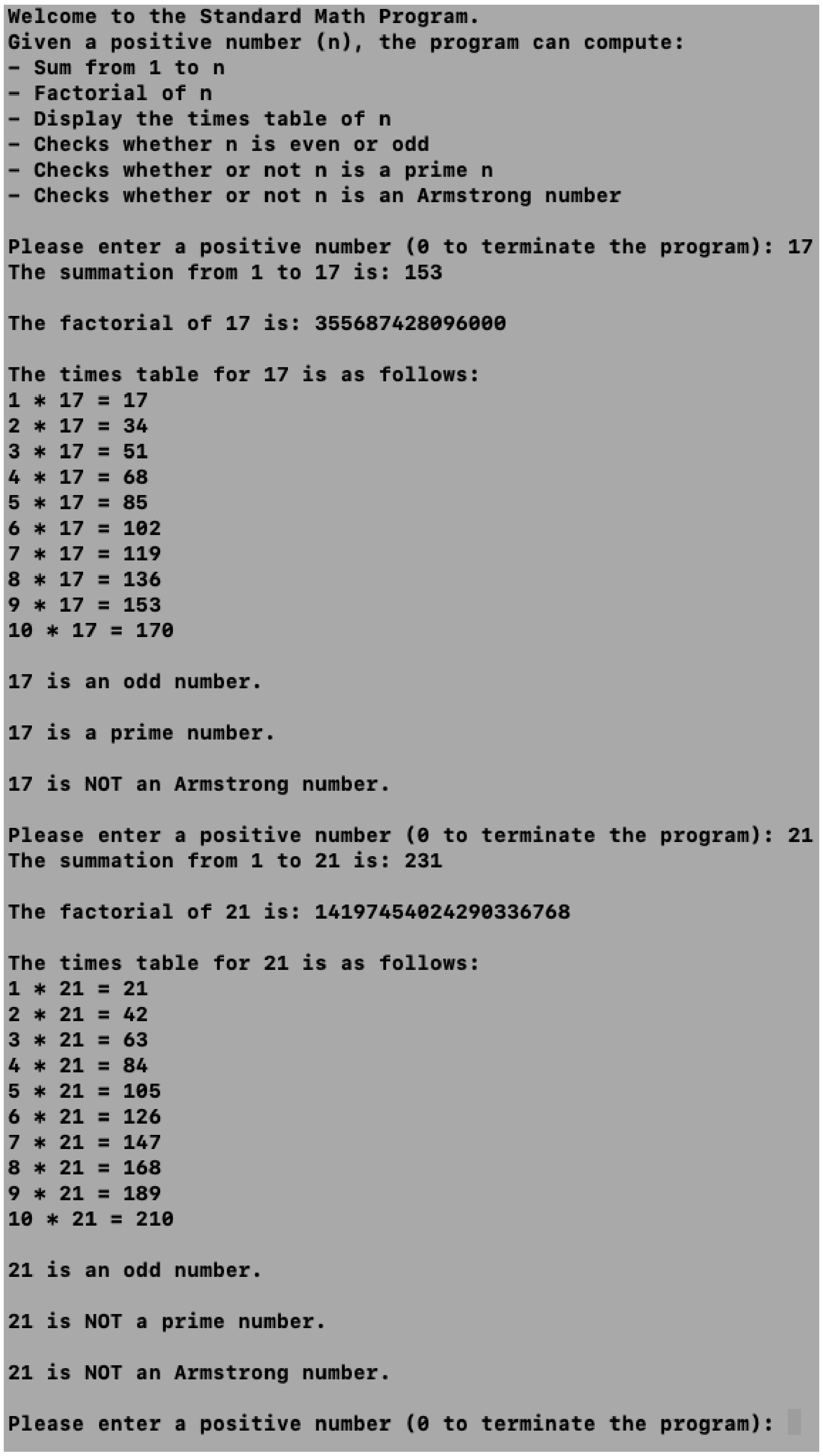
```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number


Please enter a positive number (0 to terminate the program):
```

2. Then, the program should ask the user to input a positive integer. For every positive number input through the keyboard (e.g., 17), the program should compute the aforementioned tasks and display the results in the following manner –



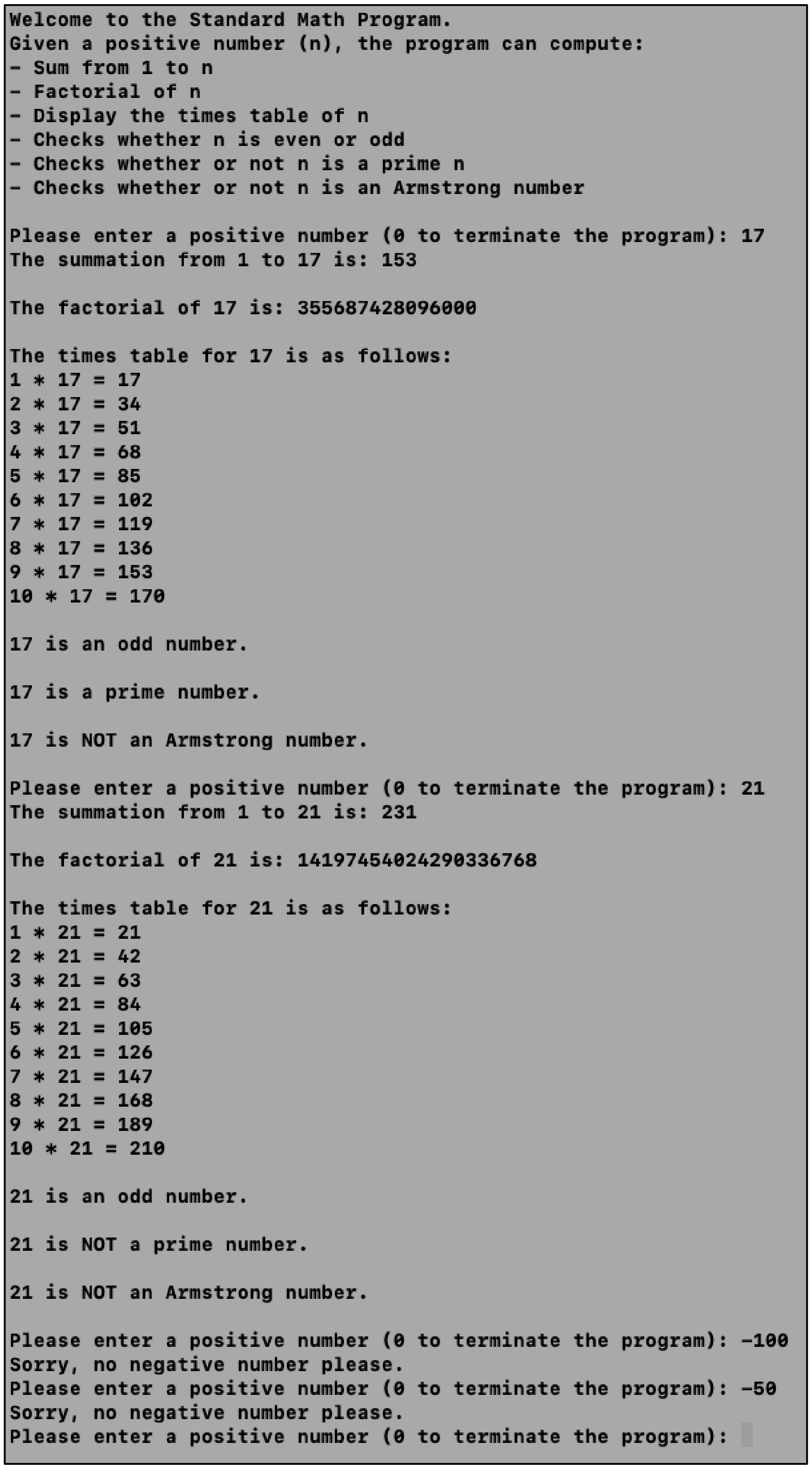
```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number

Please enter a positive number (0 to terminate the program): 17
The summation from 1 to 17 is: 153

The factorial of 17 is: 355687428096000

The times table for 17 is as follows:
1 * 17 = 17
2 * 17 = 34
3 * 17 = 51
4 * 17 = 68
5 * 17 = 85
6 * 17 = 102
7 * 17 = 119
8 * 17 = 136
9 * 17 = 153
10 * 17 = 170

17 is an odd number.

17 is a prime number.

17 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program):
```

Please take note that the program should continue asking the user for another input after performing the computations until 0 is received as the input. For example, your program should continue taking further inputs as follows:



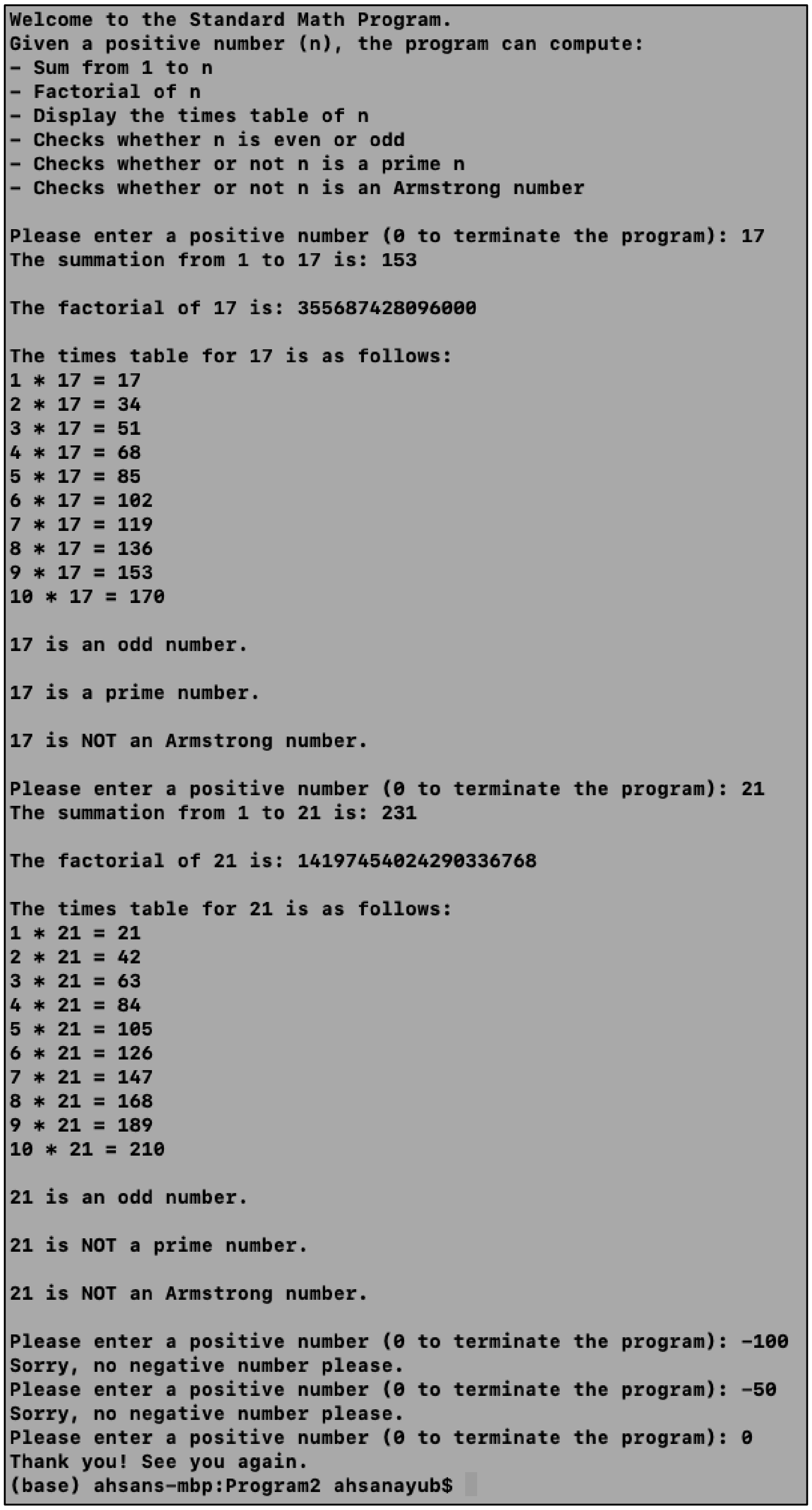
```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number

Please enter a positive number (0 to terminate the program): 17
The summation from 1 to 17 is: 153

The factorial of 17 is: 355687428096000

The times table for 17 is as follows:
1 * 17 = 17
2 * 17 = 34
3 * 17 = 51
4 * 17 = 68
5 * 17 = 85
6 * 17 = 102
7 * 17 = 119
8 * 17 = 136
9 * 17 = 153
10 * 17 = 170

17 is an odd number.

17 is a prime number.

17 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program): 21
The summation from 1 to 21 is: 231

The factorial of 21 is: 14197454024290336768

The times table for 21 is as follows:
1 * 21 = 21
2 * 21 = 42
3 * 21 = 63
4 * 21 = 84
5 * 21 = 105
6 * 21 = 126
7 * 21 = 147
8 * 21 = 168
9 * 21 = 189
10 * 21 = 210

21 is an odd number.

21 is NOT a prime number.

21 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program):
```

3. If the user inputs a negative integer, then your program should tell the user that it's an invalid entry and allow the user to provide you a positive integer. A sample execution case is as follows:



```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number

Please enter a positive number (0 to terminate the program): 17
The summation from 1 to 17 is: 153

The factorial of 17 is: 355687428096000

The times table for 17 is as follows:
1 * 17 = 17
2 * 17 = 34
3 * 17 = 51
4 * 17 = 68
5 * 17 = 85
6 * 17 = 102
7 * 17 = 119
8 * 17 = 136
9 * 17 = 153
10 * 17 = 170

17 is an odd number.

17 is a prime number.

17 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program): 21
The summation from 1 to 21 is: 231

The factorial of 21 is: 14197454024290336768

The times table for 21 is as follows:
1 * 21 = 21
2 * 21 = 42
3 * 21 = 63
4 * 21 = 84
5 * 21 = 105
6 * 21 = 126
7 * 21 = 147
8 * 21 = 168
9 * 21 = 189
10 * 21 = 210

21 is an odd number.

21 is NOT a prime number.

21 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program): -100
Sorry, no negative number please.
Please enter a positive number (0 to terminate the program): -50
Sorry, no negative number please.
Please enter a positive number (0 to terminate the program):
```

4. Your program should only terminate itself when the user inputs 0 from the keyboard.



```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number

Please enter a positive number (0 to terminate the program): 17
The summation from 1 to 17 is: 153

The factorial of 17 is: 355687428096000

The times table for 17 is as follows:
1 * 17 = 17
2 * 17 = 34
3 * 17 = 51
4 * 17 = 68
5 * 17 = 85
6 * 17 = 102
7 * 17 = 119
8 * 17 = 136
9 * 17 = 153
10 * 17 = 170

17 is an odd number.

17 is a prime number.

17 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program): 21
The summation from 1 to 21 is: 231

The factorial of 21 is: 14197454024290336768

The times table for 21 is as follows:
1 * 21 = 21
2 * 21 = 42
3 * 21 = 63
4 * 21 = 84
5 * 21 = 105
6 * 21 = 126
7 * 21 = 147
8 * 21 = 168
9 * 21 = 189
10 * 21 = 210

21 is an odd number.

21 is NOT a prime number.

21 is NOT an Armstrong number.

Please enter a positive number (0 to terminate the program): -100
Sorry, no negative number please.
Please enter a positive number (0 to terminate the program): -50
Sorry, no negative number please.
Please enter a positive number (0 to terminate the program): 0
Thank you! See you again.
(base) ahsans-mbp:Program2 ahsanayub$
```

<u>Development Instructions</u>

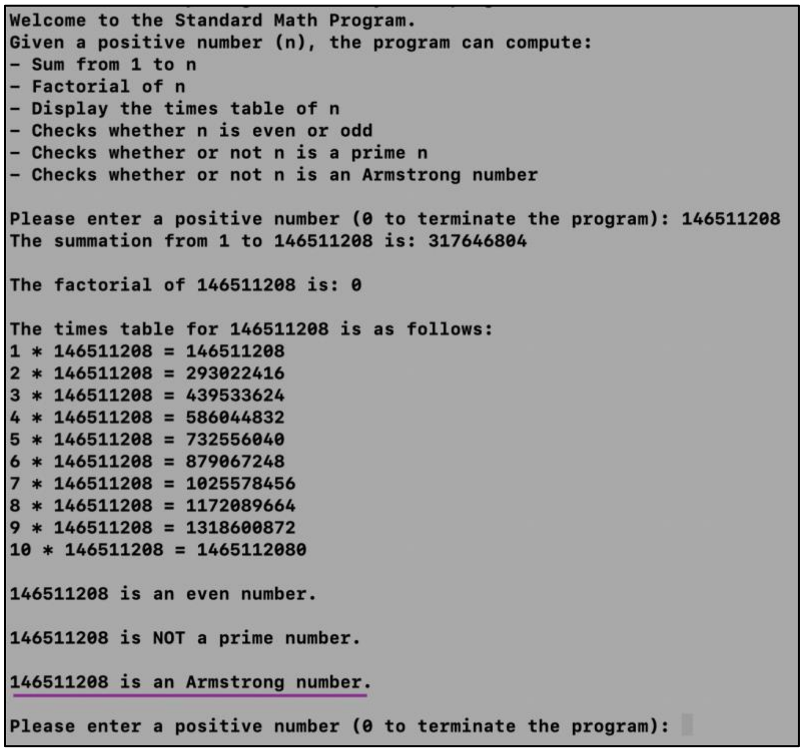
You are given three files in a compressed folder that I'd like you to utilize to perform all the tasks. The files are "main.cpp", "utils.cpp", and "utils.h". You don't need to modify any content on the header file (utils.h); however, you will see the function prototypes / declarations on there that I want you to implement in the "utils.cpp" file. That CPP file shouldn't include the main function (from where the program will start its execution) – only the definition of each function that is declared in its header file.

Please use the "main.cpp" file to invoke all the function that you've implemented to perform the tasks. One of the ways you can compile your code is as follows –

```
$ g++ -o program2 main.cpp utils.cpp
$ ./program2
```

<u>Constraint</u>:

- To test your program, the positive integer value (that user is going to give as inputs) could be as high as 150,000,000.
  Please note that an unsigned long long variable can hold the value of 65! (Factorial of 65). Therefore, if the value of positive integer becomes more than 65, then it is OK when you'll be receiving 0 as a result for the time being. ☺



```
Welcome to the Standard Math Program.
Given a positive number (n), the program can compute:
- Sum from 1 to n
- Factorial of n
- Display the times table of n
- Checks whether n is even or odd
- Checks whether or not n is a prime n
- Checks whether or not n is an Armstrong number

Please enter a positive number (0 to terminate the program): 146511208
The summation from 1 to 146511208 is: 317646804

The factorial of 146511208 is: 0

The times table for 146511208 is as follows:
1 * 146511208 = 146511208
2 * 146511208 = 293022416
3 * 146511208 = 439533624
4 * 146511208 = 586044832
5 * 146511208 = 732556040
6 * 146511208 = 879067248
7 * 146511208 = 1025578456
8 * 146511208 = 1172089664
9 * 146511208 = 1318600872
10 * 146511208 = 1465112080

146511208 is an even number.

146511208 is NOT a prime number.

146511208 is an Armstrong number.

Please enter a positive number (0 to terminate the program):
```

- To test whether or not your program correctly works for the Armstrong number, the correct test cases within the range are 153, 370, 371, 407, 1634, 8208, 9474, 54748, 92727, 93084, 548834, 1741725, 4210818, 9800817, 9926315, 24678050, 24678051, 88593477, and 146511208.

Things to Note

- I ask you to put comments throughout your code. Try not to exceed 80 characters per line.
- Declare the ownership of your program at the top of your code.
- **Submission guideline**: Please place all the files for your program, such as "utils.cpp", "main.cpp", "utils.h", and "readme.txt" in a folder named "Program2_<your_Tech_user_name>" (e.g., "Program2_mayub42"). Then, please zip the folder and submit it on iLearn before the deadline.
- The submission deadline is on Friday, October 29, 2021 at 11:59 PM (Central Daylight Time). I will be subtracting 5-points off per day late. However, please note that no late submission will be accepted from Monday, November 1, 2021 12:00 AM (CDT).

How Your Submission Might Be Graded

Points Possible	Trait	Exceptional	Acceptable	Amateur	Unsatisfactory
25	Compiling / Syntax Errors	Compiles with no syntax errors.			Does not compile.
60	Specifications	The program runs without errors and meets all of the specifications.	The program produces the correct results and displays them as specified. It also meets most of the other specifications.	The program produces correct results but does not display them correctly. OR The program displays results perfectly but results are not all correct.	The program is producing incorrect results and has display issues.
15	Readability	The code is exceptionally well organized and very easy to follow.	The code is fairly easy to read.	The code is readable only by someone who knows what it is supposed to be doing.	The code is poorly organized and very difficult to read.
-	Delivery	The program was delivered on time.	Subtracting 5 points off per day late. Will only be allowed to be turned in up to two days late. May not be submitted via email.		