60-141 – Introduction to Programming II Winter, 2017

Assignment 1

(Due: 11:59pm, Jan. 29, 2017)

Factorials in terms of Prime Numbers

The factorial of a number *N* (written ***N*!**) is defined as the product of all the integers from 1 to *N*. It is often defined recursively as follows:

$0! = 1$ (By definition)
$N! = N \times (N-1)!$

[NOTE: Factorial of a negative number is undefined, and is not permitted.]

Factorials grow very rapidly -- 5! = 120, 10! = 3,628,800. One way of specifying such large numbers is by specifying the number of times each prime number occurs in it. Thus 825 could be specified as (0 1 2 0 1) (or, (2,0) (3,1) (5,2) (7,0) (11,1)) meaning no twos, 1 three, 2 fives, no sevens and 1 eleven. For this assignment, we will follow the notation as: 825 = (2^0)*(3^1)*(5^2)*(7^0)*(11^1)

[NOTE: Although it is not required, students are advised to try to compute N! using simple integer multiplication in order to determine the largest N for which the computer does not produce an overflow. This optional exercise will underscore why this assignment is relevant in numeric computation.]

Your task is to:

Write a complete, well documented C program that will read in an integer number *N* (limited by $2 \leq N \leq 100$) and write out its factorial in terms of the numbers of its prime factors, using the output notation explained above.

Requirements and Hints:

1. You do not have to actually calculate the factorial of any number to solve this problem.
2. Given the first prime number 2, your program logic will:
   a. Determine how many times this prime number will occur in N!.
   b. Then the program will determine what is the next prime number, and go back to step a.
   c. Steps a. and b. will continue until all the prime numbers $\leq$ N are evaluated.
3. For example: 4! = 2X3X4, where the prime 2 occurs three times (2X4) = (2X2X2) = (2^3), and the prime 3 occurs only one time, (3^1). So 4! = (2^3)*(3^1). Likewise, 5! = (2^3)*(3^1)*(5^1).
4. Your program should implement at least the following 3 functions:
   a. **find_prime_count()**, that will count the number of a given prime in N!.
   b. **find_next_prime()**, that, given a prime number, will determine the next prime number.
   c. **is_prime()**, that will determine whether a number is a prime number or not.

Input:

You have to use input redirection technique to enter inputs from a text file.

Example: a.out < input.txt

The input file will consist of a series of lines, each line containing a single integer *N*. The file will be terminated by a line consisting of a single *0*.

Output (you must follow the following format!):

Output will consist of a series of blocks of lines, one block for each line of the input. Each block will start with the number N, right justified in a field of width 3, and the characters **`!',** space, **`='** and **2** spaces.

This will be followed by a list of pairs of numbers in parenthesis, such as **(x^y),** separated by ***,** where **x** is a prime number and **y** is the number of times the prime number **x** occurs in **N!**.
These should be right justified in variable width fields (shown below) and each line (except the last of a block, which may be shorter) should contain 9 pairs of numbers. Any lines after the first should be indented.

Follow the layout of the example shown below **exactly**.

Sample input file:

```
5
53
100
0
```

Sample output (you must follow the output format):

```
  5! = (2^3)*(3^1)*(5^1)

 53! = (2^49)*(3^23)*(5^12)*(7^8)*(11^4)*(13^4)*(17^3)*(19^2)*(23^2)
       *(29^1)*(31^1)*(37^1)*(41^1)*(43^1)*(47^1)*(53^1)

100! = (2^97)*(3^48)*(5^24)*(7^16)*(11^9)*(13^7)*(17^5)*(19^5)*(23^4)
       *(29^3)*(31^3)*(37^2)*(41^2)*(43^2)*(47^2)*(53^1)*(59^1)*(61^1)
       *(67^1)*(71^1)*(73^1)*(79^1)*(83^1)*(89^1)*(97^1)
```

REQUIREMENTS:

- Write and document a complete C program that is capable of satisfying the requirements of this assignment problem.
- UNDOCUMENTED OR IMPROPERLY DOCUMENTED code will automatically lose 50% marks.
- PLAGIARIZED work will not be graded and receive a mark of ZERO and reported according to the Senate bylaws.
- The question requires use of I/O redirection. Please review the textbook for an example on using I/O redirection from flat files.
- TO SUBMIT: No later than the submission deadline, your submission must be received. Late submissions are not accepted and will receive a mark of ZERO.
- Submit your work through Blackboard as attachments, including both the source file (assign1.c) and the script file (assign1.txt) - see below how to create the script file.

To create a script file (one that logs your compilation steps and your output in a text file):

1. script assign1.txt
2. cat assign1.c
3. cat input.txt
4. cc assign1.c
5. a.out < input.txt
6. ls -l
7. exit (DO NOT FORGET THIS STEP!!)

Submit both files using Blackboard.

NOTE: Submissions that are not received correctly by the deadline will automatically receive a ZERO mark. In the event that more than one email submission is sent, only the last one (according to the date and time stamp) will be marked.

It is your responsibility to ensure the email attachment is sent correctly (readable) and to the right mailbox by the deadline. If you omit the email subject header or fail to follow the format provided above for the attachment file, your assignment may not be graded.

Late assignment submissions are not accepted!

NOTES:

1. Your assignment must be RECEIVED by the due date and time. Late assignment submissions are NOT accepted. Keep your script file, and all your code unmodified as proof of its completion in case it is not received.

2. It is your responsibility to get an early start on the assignment, research and ask questions ahead of time from the due date.

3. Marks will be deducted for unclear code. (improper spacing and alignment, lack of proper indentation style, hard to read programs and missing outputs).

4. Make sure you turn in a complete script file that clearly shows: your code, your compilation process, a listing of the directory showing your source file(s) and the execution of a.out with the date/time stamps, and the output. (DO NOT SUBMIT a.out FILES!)

5. **PLAGIARISM:** CHEATING IS NOT TOLERATED. You must submit your own work. Students who are suspected of copying someone else's work will be reported to the department's chair and the Dean of Science and be dealt with in accordance with the University policies. You should not share your code with others. Codes that are similar to each other will BOTH be reported as potential evidence of copying. It is imperative that you write your own code.

6. Authorized/limited help on this assignment may be provided directly from your Lecture or Lab instructors and Teaching Assistants.