**Name:**

**Advanced Programming in C++**

**Lab Exercise 2/20/2023**

In this exercise, you will be writing some programs to implement a variety of algorithms to solve various problems. Submit you source code for these programs.

1. Write a program which asks the user to enter a positive integer 'n' (Assume that the user always enters a positive integer) and based on the following conditions, prints the appropriate results exactly as shown in the following format.

when 'n' is divisible by both 2 and 3, then your program should print

BOTH

when 'n' is divisible by only one of the numbers i.e divisible by 2 but not divisible by 3 (for example 8), or divisible by 3 but not divisible by 2 (for example 9), your program should print

ONE

when 'n' is neither divisible by 2 nor divisible by 3 (for example 25), your program should print

NEITHER

1. Write a program that asks the user to enter a positive integer n. Assuming that this integer represents the number of seconds, your program should convert the number of seconds into days, hours, minutes, and seconds and prints them exactly in the format specified below. Here are a few sample runs of what your program is supposed to do:   
     
   when user enters

369121517

your program should print:

4272 days 5 hours 45 minutes 17 seconds

when user enters

24680

your program should print:

0 days 6 hours 51 minutes 20 seconds

when user enters

129600

your program should print:

1 days 12 hours 0 minutes 0 seconds

Note that the numbers and words in the above output are separated by only one space. All the words are in lower case. Your output should exactly match the format shown above.

1. Write a function that accepts two positive integers as function parameters and **returns** their least common multiple (LCM). The LCM of two integers a and b is the smallest (non zero) positive integer that is divisible by both a and b. For example, the LCM of 4 and 6 is 12, the LCM of 10 and 5 is 10.
2. Write a function named **count\_consonants** that receives a string as parameter and returns the total count of the consonants in the string. Consonants are all the characters in the English alphabet except for 'a', 'e', 'i', 'o', 'u'. If the same consonant repeats multiple times you should count all of them. Note that capitalization and punctuation does not matter here i.e. a lower case character should be considered the same as an upper case character.