**Name:**

**Advanced Programming in C++**

**Lab Exercise 4/6/2021**

**Problem Solving**

In this activity, you will use your programming prowess to solve a variety of problems. You may use any programming language to solve these problems (Python, VC#.NET, Java, or C++). Your task is to complete as many of these problems as possible in the 90 minute session.

1. The sum of the primes below 10 is 2 + 3 + 5 + 7 = 17.

Find the sum of all the primes below two million.

Answer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. 215 = 32768 and the sum of its digits is 3 + 2 + 7 + 6 + 8 = 26.

What is the sum of the digits of the number 21000?

Answer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. The following iterative sequence is defined for the set of positive integers:

n = n/2 (n is even)  
n = 3n + 1 (n is odd)

Using the rule above and starting with 13, we generate the following sequence:

13 → 40 → 20 → 10 → 5 → 16 → 8 → 4 → 2 → 1

It can be seen that this sequence (starting at 13 and finishing at 1) contains 10 terms. Although it has not been proved yet (Collatz Problem), it is thought that all starting numbers finish at 1.

Which starting number, under one million, produces the longest chain?

NOTE: Once the chain starts the terms are allowed to go above one million.

Helpful hint: When developing your algorithm, only test chain lengths for smaller numbers such as numbers to 1000 instead of 1000000.

Answer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Using names.txt (found on the server), a 46K text file containing over five-thousand first names, begin by sorting it into alphabetical order. Then working out the alphabetical value for each name, multiply this value by its alphabetical position in the list to obtain a name score.

For example, when the list is sorted into alphabetical order, COLIN, which is worth 3 + 15 + 12 + 9 + 14 = 53, is the 938th name in the list. So, COLIN would obtain a score of 938 × 53 = 49714.

What is the total of all the name scores in the file?

Answer: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_