ACMS 20210 Assignment 2

Due: 11:59 PM on Friday, September 14, 2018

Submit the C++ programs below using Sakai. If you are compiling from the command line on the CRC or another Linux system, I suggest using the command

```
g++ your_program_name.cpp -std=c++1y -o your_executable_name
```

This tells the compiler (called g++) to compile your .cpp source file into an executable file. To run this executable file from the command line, type

```
./your_executable_name
```

If you do not yet know how to use the CRC and do not have a compiler on your local machine, I suggest using the website *cpp.sh* to test your code (be sure to save a copy on your local machine, though!). You must submit your code in separate .cpp files.

Use the following naming scheme for submitting your solutions on Sakai: For exercise n on homework m, the name of your submission should be hw_m_ex_n.cpp. As an example, for homework 1 exercise 3, the name of your submission should be: hw_1_ex_3.cpp.

1. Write a program that prompts the user to enter three integers. The program then reports the values of the three integers, ordered from least to greatest. (Use conditionals to determine the order of the integers. There are more sophisticated ways to sort larger collections of data that we might discuss toward the end of the semester if there is time.)

2. Revise your third program from Homework 1 to account for some additional details:

   (a) Use correct grammar for when the user enters one of an item (i.e. do not have your program report "1 apples" but instead "1 apple" in this case).

   (b) Display money to two decimal places. There are several ways of doing this: the easiest is probably to use setprecision from the iomanip library. Another possibility is to use modular arithmetic with integer variables.

   (c) If the user enters a negative value for the number of an item, prompt them to enter the value again until they enter a non-negative number of items. (Requires iteration.)

3. Write a program that prompts the user to enter a positive integer N. The program should then compute the sum of the cubes of the integers from 1 to N inclusive using a loop. The program should also compute the sums of the cubes of the integers using the formula:

$$\sum_{k=1}^{N} k^3 = \left(\frac{(N)(N+1)}{2} \right)^2,$$

   and report this value to the user as well.

4. Write a program that prompts the user to enter a positive integer N. The program should then compute the sum of all the integers from 1 to N inclusive that are divisible by either 2 or by 3, but not by both. Report the sum of the appropriate integers to the user. Use a for loop and conditionals to do this problem, even if you can think of a more efficient way that uses neither.

5. Write a program to approximate the sum of the following alternating series:

$$\sum_{k=0}^{\infty} (-1)^k \frac{1}{k + 1 + \ln(k^2 + 4)}$$

   so that the error in the sum is smaller than a given tolerance. Your program should request the user to enter a non-negative error tolerance, and then add terms in order until the error between the sum of the series and the partial sum is less than the tolerance (assuming the arithmetic were exact). Use the error bound from the alternating series test to determine when to stop adding terms. Report the sum of the series and the number of terms computed to the user. (You may wish to use the log function from the cmath library in the computation of the terms.)

The following two problems are not due for submission, but I recommend completing them using truth tables.

- Show that for mathematical statements P, Q, R, we have the equivalences:

  (a) $\sim\sim P \equiv P$

  (b) $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$

  (c) $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$

  (d) $\sim (P \wedge Q) \equiv (\sim P) \vee (\sim Q)$

  (e) $\sim (P \vee Q) \equiv (\sim P) \wedge (\sim Q)$

- Show that for mathematical statements P, Q, we have the equivalence:

$$\sim ((P \wedge \sim Q) \vee (Q \wedge \sim P)) \equiv (P \wedge Q) \vee ((\sim P) \wedge (\sim Q))$$