Lab 3: Numeric Errors
Monday, 19 February 2018
Due: 25 February 2018 at 11:59pm

We usually say that computers are better than people at "number crunching". But as this lab demonstrates, there are several issues that programmers need to take into account when performing computer arithmetic. During this lab you will investigate the problems of numeric overflow for integer and floating point number representations and precision limitations of floating point number representation. Any explanation that is asked for in this assignment should be written in the README.md file that is included in your cloned repo. (NOTE for Mac users: Apple decided to represent long numbers with 8 bit words instead of the usual 4 bit words, so any problem dealing with long numbers may take longer to find an overflow than on other computers.)

Lab Assignment:

0. Read the **Turn In**: see instructions at the end of this document.

1. Create a new project and write a code segment that repeatedly reads n and computes the sum of the consecutive integers 1 through n: $sum = \sum_{i=1}^{n} i$

   Use **type short** for the consecutive integers and the sum.
   Report n and the sum.

2. An overflow error occurs when the result of a computation is larger than the storage for the result can handle. Experiment with several values for the upper limit of the sum (n in the formula). How can you detect an overflow in this process? Report the value of n that produces the overflow. (Hint: try values for n around 250).

3. Repeat parts 1 and 2 using **type long** for the consecutive integers and the sum. Again, test different values for the upper limit and report the value of n that causes an overflow.

4. Write a function that repeatedly reads n and computes the factorial function, n!, the product of consecutive integers:

   $$product = \prod_{i=1}^{n} i$$

   Use **type long** for the consecutive integers and **type float** for the product. Experiment with several values for n. How can you detect an overflow in this case? Report the value of n that causes an overflow.

5. Repeat part 4 using **type double** to store the result. What value of n causes an overflow?

6. Here is an example of "strange" floating point number behavior: Write a function, f(n), that computes

$$\left(\sum_{i=1}^{n}\frac{1}{n}\right) \; - \; 1.0$$

Compute the sum by computing and storing the ratio 1/n and then adding up n copies of the ratio. What is the expected value of this function if the computations are exact? Check your expectation with the lab assistant.
Try the computation with different values of n and experiment with **float and double as the data types** to store the ratio (1/n) and the result. Explain the results.

7. Here is a puzzle. Execute the following piece of code, and explain where and why the numeric error occurs:

```
for (float i = 3.4; i < 4.4; i += 0.2)
{
     cout << "i = " << i << endl;
}
```

8. Now change the variable i in part 7 to type double. Explain the effect.

Turn In:

Include your answers to the above questions in the README.md of your repo.
Include your name as a comment in the code.
Use the following guideline for naming your file: lab3-<uwyo-username>.cpp (don't write the "<" and ">", those are used in our field to usually denote a test placeholder) and please include this file in the top level of your repository (same level as your README).

Grading:

L3.P1: Sum function written and the **n** values where overflow occurred for both type short and type long are reported.
L3.P2: Product function written and the **n** values where overflow occurred for both type float and type double are reported.
L3.P3: Code generates the asked for sum and explanation given for Problem 6.
L3.P4: Point of numeric error correctly identified and explanation given for both float and double data-types.

Rubric: (next page)

Point	**5pts**	**2pts**	**0pts**
L3.P1	Correct code AND values	Incorrect code OR values	Incorrect code AND values
L3.P2	Correct code AND values	Incorrect code OR values	Incorrect code AND values
L3.P3	Correct code AND explanation	Missing code OR explanation	Incorrect code AND values
L3.P4	Correct points of error AND explanation	Missing points of error OR explanation	Missing points of error AND explanation

Total Points: 20