Zixuan Kang 903404772

Appendix C - Turn-in Sheet

Part 1: OUT OF RANGE ERROR - In addition to submitting your source code, enter the results below:

The largest power that can be represented with regular int type is ___11___

The largest power that can be represented with a long long type is ___22___

Part 2: ROUND-OFF ERROR - In addition to submitting your source code, enter the results below:

Using a float

x(+) = ___-3072___ %error from actual root = ___-2.4___

x(-) = ___-0.001___ %error from actual root = ___4.474974×10^{-6}___

Using a double

x(+) = ___-3000___ %error from actual root = ___9.94381×10^{-9}___

x(-) = ___-0.001___ %error from actual root = ___0___

Given the fact that both roots can easily be represented with a float, why do you get a couple percent error for one of the roots?.

when calulating square root and dividing a big number it will cause overflow.

Part 3: TRUNCATION ERROR - In addition to submitting your source code, enter the results below:

Float Results:

forward answer = ___1.41421___ forward %error = ___0.000364174___

backward answer = ___1.41421___ backward %error = ___0.000355745.___

Double Results:

forward answer = ___1.41421___ forward %error = ___0___

backward answer = ___1.41421___ backward %error = ___1.57009×10^{-14}___

Is there a difference between the two summations? Why do you think there would be a difference between the forward and backward summations?

Yes, there is. When summing forward the number are manipulated to have the same power of 2 to the last term which cause less precision where as when summing backward the terms have the same power of 2 to the first term providing higher precision.

Part 4: COLLATZ SEQUENCE - In addition to submitting your source code, enter the results below:

Assuming an input value of **n=21** what is the total number of terms and the sequence of terms:

terms: ___8___

Values of Terms:

21 → 64 → 32 → 16 → 8 → 4 → 2 → 1