Announcements

Thursday Extras: CS Commons on Thursdays @ 4:00 pm

- None this week (that I know about)

Usual office hours this week

- Monday, Wednesday: 1:00-3:00 (pm)
- Tuesday, Thursday, Friday: 2:30-4:00 (pm)

Friday's test covers through today's material

CSC 161 Schedule

- Today: Floating point representation (due Wed., Oct 30)
- Wednesday:
  - Data representation and its consequences: TO BE DONE
  - Machine-level ops: optional

Floating point numbers

- questions
- clicker questions

Consider the code for [non-rounded] Approach 1 for Supplemental Problem 2.

According to algebra, the formula for the payment should be exact, so the loan should stop exactly on time.

However, for a $26,000 loan at 8.75 annual rate for 300 months, the program goes for 301 months with a final payment of $0.00. What is going on?

A. The final payment $0.00 does not count, so ignore it.
B. Computers can be difficult, and this is yet another example.
C. Real numbers (e.g., floats, double) are not stored exactly, and the loop condition does not reflect numerical error inherent in all floating-point computations.
D. balance + interest is a fraction of a cent more than the payment after 299 months, so the loop goes to month 300, bringing the last month to 301.
E. Both C and D

In a single-precision (`float`) representation, what is the binary representation of the <u>exponent</u> for

$0.75_{10} = 1.1 \times 2^{-1}$

A. 10000001
B. 11111110
C. 11111111
D. 01111110
E. None of the above

In a single-precision (`float`) representation, what is the binary representation of the mantissa for

$6.875_{10} = 110.111_2$

23 bits

A. `11011100...00`
B. `10111000...00`
C. `00...00110111`
D. `00...00010111`
E. None of the above

Math != Computer Engineering

$2.00000 \times 10^{0} + 6.00000 \times 10^{-5} + 6.00000 \times 10^{-5} = 2.000012 \times 10^{0}$

$(2.00000 \times 10^{0} + 6.00000 \times 10^{-5}) + 6.00000 \times 10^{-5}$

$2.000006 \qquad + 6.00000 \times 10^{-5}$

round | truncate

2.00001×10^{0} | 2.00000×10^{0}

round | truncate

$2.00001 \times 10^{0} + 6.00000 \times 10^{-5} = 2.000016 \times 10^{0}$

round

$= 2.00002 \times 10^{0}$

$2.00000 \times 10^{0} + 6.00000 \times 10^{-5} = 2.000006 \times 10^{0}$

truncate

$= 2.00000 \times 10^{0}$

Why precision matters

Between 1981 and 1983, the Vancouver Stock Exchange Index was computed with each trade of stock. The computation used four decimal places in the computation, but the result was then truncated (not rounded) to three places of accuracy. This computation was then repeated with each of the approximately 3000 trades that occurred daily. As a result of the truncation of the calculation to three decimal places, the index lost about 1 point a day, or about 20 points per month. Thus, to correct this consistent numerical error, the index was recomputed and corrected over the weekend of November 26, 1983. Specifically, when trading concluded on November 25, the Vancouver Stock Exchange Index was quoted as 524.811; the corrected figure used to open on Monday morning was 1098.892. The correction for 22 months of compounded error caused the Index to increase by by 574.081 over a weekend without any changes in stock prices. In this example, computations began with correct data, but subsequent processing introduced a significant error in the result.

Source: Walker, H. M. (1994) *The limits of computing*, pp. 100–101.