# MAT360 Exam 1 (Spring 2017)

**Name:**

**Directions**: Show your work! Answers without justification will likely result in few points. Your written work also allows me the option of giving you partial credit in the event of an incorrect final answer (but good reasoning). Indicate clearly your answer to each problem (e.g., put a box around it). You should skip 20 points (of your choice). Write "skip" **clearly** on those parts you skip. **Good luck!**

**Problem 1**. (20 pts) Compute the following, originally done in six-digit arithmetic, but using both three-digit round-to-even and three-digit truncation. **Assume that each constant and the result of each arithmetic operation must be converted to a machine number** using the appropriate float ($fl$) conversion.

$$\frac{17.4500 - 2.23606}{3.14159 - 2.71828} \ (= 35.9404)$$

Compute differences first, then quotient, then errors (and relative errors) and put them in the following table (please show your work below the table):

| Method | Numerator | Denominator | Approximation | Error | Relative Error |
|---|---|---|---|---|---|
| 3-digit truncation | | | | | |
| 3-digit rounding | | | | | |

- (10 pts) truncation:

- (10 pts) rounding:

**Problem 2**. (20 pts) Avoidable computational errors

a. (10 pts) We all know that the roots of a quadratic are given as $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$. However we shouldn't necessarily compute them that way. Compute the roots of the quadratic $x^2+500x+1$ using four-digit arithmetic. Do it in such a way that you obtain the best two roots (or approximations to the roots) that you can.

| | |
|---|---|
| Root 1 | |
| Root 2 | |

b. (10 pts) For which values of $x$ do tiny relative errors in $x$ result in large relative errors in $\cos(x)$?

**Problem 3**. (20 pts) Given the toy computer

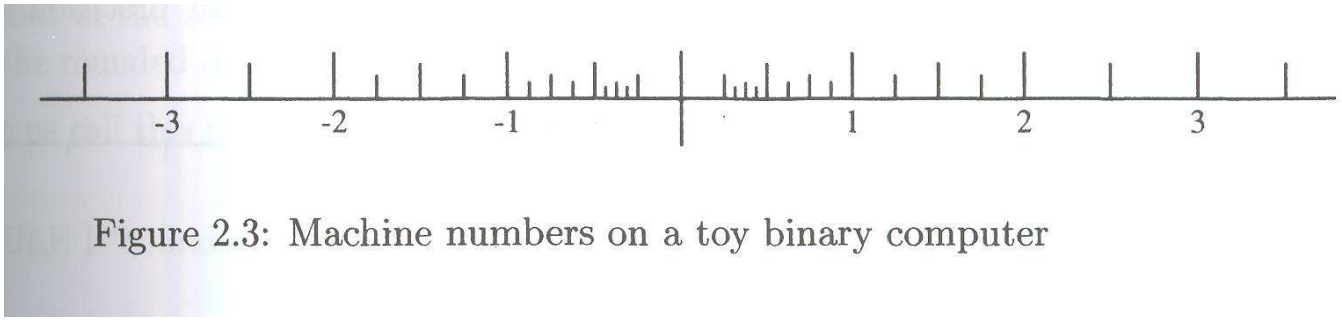



Figure 2.3: Machine numbers on a toy binary computer

a. (5 pts) Give the base, precision, and the exponent range (permissible values of $e$, the exponent) of the computer (you may simply describe it the same way that the authors did).

b. (5 pts) Assume round-to-even. What machine number will represent 1.2? 2.1? Compute $fl(fl(2.1) + fl(1.2))$. Illustrate this calculation on the diagram of the machine above.

| $x$ | Machine Number |
|---|---|
| 1.2 | |
| 2.1 | |
| $fl(fl(2.1) + fl(1.2))$ | |

c. (10 pts) Compute the product $fl(fl(1.2) \cdot fl(0.43))$ using this toy computer, assuming rounding-to-even arithmetic.

| $x$ | Machine Number |
|---|---|
| $fl(fl(1.2) \cdot fl(0.43))$ | |

**Problem 4**. (20 pts) Continuing with the toy computer

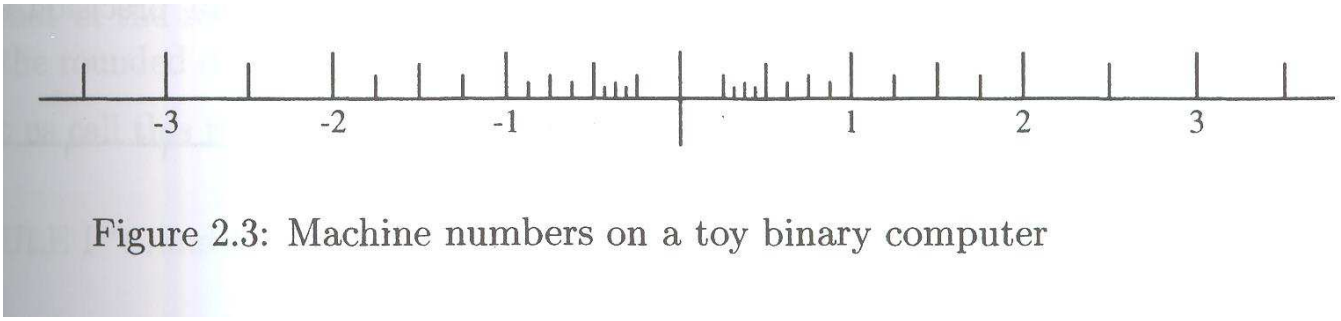



Figure 2.3: Machine numbers on a toy binary computer

a. (5 pts) Suppose we extend the possible exponents, without making any other changes to the machine. What machine numbers (answers given in base 10) would be added to the machine at the $e = 8$ level?

b. (5 pts) If we added just one bit to the machine, without making any other changes to the machine, indicate clearly on the diagram above the numbers that would be added at the $e = 0$ level? How many new numbers would be added, and what are their values or machine representations?

c. (10 pts) Clearly mark on the diagram above the machine number $(1.01)_2$ (call it $A$). Multiply $A$ by two, and indicate the resulting number. Divide $A$ by 2 and indicate that number. What do you notice about these operations and the positions of $2A$ and $A/2$ relative to the location of $A$?

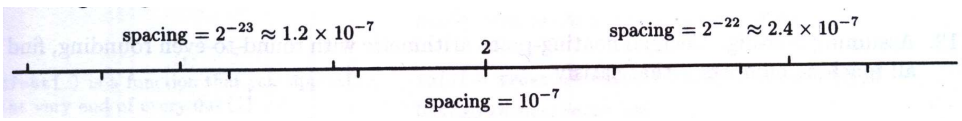
**Problem 5:** (20 pts) Short answer (5 points each):

a. Why do the authors advise using $4 \cdot \tan^{-1}(1)$ as an input to software, rather than any particular decimal approximation to $\pi$?

b. Write $935_{10}$ in base 2

c. Write $10011011_2$ in base 10.

d. Consider the Figure on page 55, representing two computers that use IEEE single precision machine numbers above and 8-digit numbers below the axis: Why does the spacing between




the base 2 numbers increase to the right of 2, but the spacing of the base 10 machine does not?

**Problem 6:** (20 pts) Recall that the Babylonian algorithm for calculating $\sqrt{c}$ can be expressed as an iterative method, where we start with an initial value $x = x_0$, then

$$y = \frac{c}{x} \text{ and } x = \frac{x+y}{2}$$

Now do it again.

a. (10 pts) Assume that $x_0 = 1$, and use three-digit round-to-even arithmetic on all real numbers. Compute an approximation to $\sqrt{\pi}$ using two iterations of the Babylonian algorithm to compute $x_1$ and $x_2$. Assume that the true value is 1.77245. How well does the algorithm do?

b. (10 pts) Now express the total error in the result $f(\pi)$ as a sum of propogation data error and computational error. Use the ideas of Figure 1.8, and consider that **two** applications of the Babylonian algorithm (starting from $x_0 = 1$) is the approximation ($\hat{f}$) to the square root function ($f$). The data is $c = \pi$, but we have approximated it using three digits.

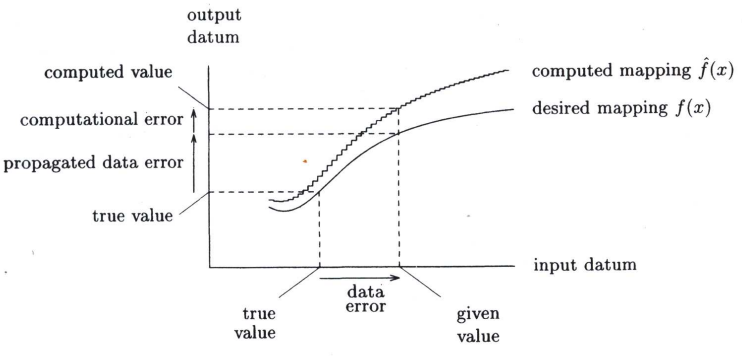



Figure 1.8: Data error vs. computational error

**Problem 7:** (20 pts) Consider Example 2.7, p. 67:

"The calculation of $x - \pi$ is typical of what must be done to compute trigonometric functions. Assume the use of four-decimal-digit numbers. If you use $x - 3.142$, you will get a completely wrong answer for $x = 3.142$ and poor accuracy for nearby values. However, $f(x) = (x - 3.141) - 0.0005927$ produces results correctly rounded to four significant digits for all $x$."

a. (10 pts) Calculate the correct values of the following, using four-decimal-digit arithmetic and round-to-even:

| $f(3.000)$ | $f(3.100)$ | $f(3.140)$ | $f(3.142)$ | $f(\pi)$ |
|---|---|---|---|---|
| | | | | |

b. (5 pts) Explain the significance (and even the cleverness) of this idea, and why it works.

c. (5 pts) The authors conclude that "...a carefully crafted expression like this can be undone by an optimizing compiler that assumes that the associative law holds for floating-point addition." What do they mean to suggest that the compiler might do?