• Please give details of your calculation. A direct answer without explanation is not counted.

• Your answers must be in English.

• Please carefully read problem statements.

• During the exam you are not allowed to borrow others’ class notes.

• Try to work on easier questions first.

1. (15%) Write the IEEE binary format of 0.0859375 using single precision

2. (20%) Recall that we have the following theorem

   **Theorem 1** Using $p + 1$ digits for $x - y$ ⇒ relative rounding error $< 2\epsilon$ ($\epsilon$: machine epsilon)

   If in the proof we assume $y \geq 0$ and the rounding is by truncation, can we prove a tighter bound that the relative rounding error $\leq \epsilon$? If you think the answer is yes, show the proof. If no, explain why.

   Please note that there is an error in our slide:

   $$(x - y) - (x - \bar{y} + \delta) = \bar{y} - y + \delta$$

   should be

   $$(x - y) - (x - \bar{y} + \delta) = \bar{y} - y - \delta$$

3. (20%) A simple Internet search finds that someone implements isnanf (float version of isnan) and isninf by the following code:

   ```c
   #define isnanf(x) (((*(long *)&(x) & 0x7f800000L)==0x7f800000L) &&
                        ((*long *)&(x) & 0x007fffffL)!=0000000000L))
   #define isninf(x) (((*(long *)&(x) & 0x7f800000L)==0x7f800000L) &&
                      ((*long *)&(x) & 0x007fffffL)==0000000000L))
   ```

   Note that in C, a long int starting with ‘0x’ means a hexadecimal number.

   (a) Explain in detail why the above code can check nan and inf?
(b) If we modify the above code for double, what these hexadecimal numbers should be changed? Other modifications (e.g., change long to long long) may be needed, but you just need to give the new hexadecimal numbers.

4. (20%) Calculate the LU factorization of the following matrix.

\[
\begin{bmatrix}
4 & 2 & 2 & 4 \\
2 & 4 & 2 & 3 \\
6 & 9 & 6 & 11 \\
2 & 7 & 4 & 9 \\
\end{bmatrix}
\]

Please show details of your calculation.

5. (25%) We taught two ways to derive the Cholesky factorization. One is from the viewpoint of directly solving equations. In particular, we used

\[
A_{..j} = \sum_{k=1}^{j} L_{jk} L_{..k}.
\]

Then we obtain the \( j \)th column using the 1st to \((j - 1)\)st columns.

What if we check \( A_{i..} \)? Please derive a “row” version of Cholesky factorization. That is to obtain the \( i \)th row using the 1st to \((i - 1)\)st rows. Show your calculation and then the pseudo code.