Exercise 1.1
Given the following C module:

```
#define NLIMIT 128
int A[NLIMIT];
int ntop = 0; // largest tabulated A entry so far
extern void errmsg( void );

int f( int n )
{
    int i;
    if (n<1 || n>=NLIMIT) {
        errmsg();
        return 0;
    }
    if (n>ntop) {
        for (i=ntop+1; i<=n; i++)
        ntop = n;
    }
    return A[n];
}
```

1. Give the symbol table of the program and the abstract syntax tree of function \( f \).
   Use the notation of the slides and complement with your own notation where necessary.

2. Give a HIR representation of function \( f \).
   (Hint: You can use short-circuit evaluation of the || operator).

3. Give a MIR representation of function \( f \).

4. Give a LIR representation of function \( f \).

5. Identify the basic block leaders in the MIR representation of \( f \) and draw the basic block graph and the flow graph of \( f \). Assume here for simplicity that function calls do not delimit a basic block.

6. Compute the dominators and immediate dominators for all nodes in the flow graph of \( f \), using the iterative algorithm. Draw the dominator tree of function \( f \).
Exercise 1.2
Given the following basic block in MIR code:

\[
\begin{align*}
    c & \leftarrow a + 1 \\
    b & \leftarrow 2 \times a \\
    b & \leftarrow b + c \\
    d & \leftarrow c \\
    a & \leftarrow b / d \\
    d & \leftarrow b / c \\
    c & \leftarrow a - b \\
    a & \leftarrow 2 \times a
\end{align*}
\]

Perform local common subexpression elimination to obtain a DAG representation of the code.

Exercise 1.3
We visualized the dominance relation on the slides with the light flow analogy. What would be the corresponding setup for the postdominance relation?
Recall that the postdominance relation is defined as follows:
\[ p \text{ postdominates } b \text{ (} p \text{ pdom } b \text{)} \text{ if every possible execution path } b \rightarrow^* \text{exit} \text{ includes } p. \]

Exercise 1.4
Given the following MIR code

\[
\begin{align*}
    i & \leftarrow \text{From} \\
    s & \leftarrow 0 \\
    t_0 & \leftarrow ( \text{From} \leq \text{To}) \\
    \text{if not } t_0 & \text{ goto } L0 \\
L1: & \quad t_1 = i \times \text{Int}_\text{Size} \\
    & \quad t_2 \leftarrow A + t_1 \\
    & \quad t_3 \leftarrow [t_2] \text{ (memory read)} \\
    & \quad t_4 \leftarrow (a < 0) \\
    & \quad \text{if } t_4 \text{ goto } L2 \\
    & \quad t_5 \leftarrow a \times 0.8 \\
    & \quad t_6 \leftarrow s + t_5 \\
    & \quad s \leftarrow t_6 \\
    & \quad \text{goto } L3 \\
L2: & \quad t_7 \leftarrow a \times 0.8 \\
    & \quad t_8 \leftarrow s - t_7 \\
    & \quad s \leftarrow t_8 \\
L3: & \quad t_9 \leftarrow i \times \text{Int}_\text{Size} \\
    & \quad t_{10} \leftarrow E + t_7 \\
    & \quad [t_{10}] \leftarrow s \text{ (memory write)} \\
    & \quad t_{11} \leftarrow i + 1 \\
    & \quad i \leftarrow t_{11} \\
    & \quad t_{12} \leftarrow (i \leq \text{To}) \\
    & \quad \text{if } t_{12} \text{ goto } L1 \\
L0: & \quad \text{return } s
\end{align*}
\]

1. Draw the control flow graph, find the basic block leaders, and draw the basic block graph and the flow graph.
2. Compute the dominators of all nodes in the flow graph using the iterative algorithm.
Compute the immediate dominators and draw the dominator tree.

3. Do a DFS traversal of the flow graph, starting with the entry node, and label the edges with T, F, B and C.

4. Identify strongly connected components and natural loops in the code. Is the flow graph reducible? If yes, draw the region hierarchy graph, following the structural analysis transformations suggested in the lecture on control flow analysis.

   Give names to the variable definitions and expressions that occur in the code [Example: \( d_1 \) denotes the first assignment to \( i \), and \( e_1 \) denotes the comparison (From \( \leq \) To)].

5. Formulate the flow functions and the dataflow equations for the Available Expressions problem for each basic block.

6. Give a trivial, but correct solution to the Available Expressions problem.

7. Determine a fixed-point solution of the Available Expressions problem with the worklist algorithm. Explain whether your solution is a maximal safe solution or not.

8. Formulate the flow functions and the dataflow equations for the Reaching Definitions (May-Reach) problem for each basic block.


10. Determine a fixed-point solution of the Reaching Definitions problem with the worklist algorithm. Explain whether your solution is a maximal (in the information-theoretic, not set-theoretic sense) safe solution or not.

Please prepare exercises 1.1–1.4.4 for the first lesson, exercises 1.4.4–1.4.10 for the second lesson.