Problem 1.

Refer to Example 5.4 in the book. Write a nonrecursive LDL++ program that solves the temporal coalescing problem of Example 5.4, using user-defined aggregates. Assume that the tuples of the valid-time relations are already sorted according to the Start time of their temporal intervals.

See Example 2.5 in the TPLP 3 (1) paper assigned for reading.

Problem 2

Given the database table emp(Eno, Sal, Dno)

1. Write an LDL++ program to compute the median salary of the employees in each department. You can use aggregates and assume that
   - the emp table is already sorted by increasing salary.
   - no two employees in the same department make the same salary
   - each department has an odd number of employees.

2. Are the aggregate you just defined monotonically? Justify your answer.

We can use the aggregate count and mcount defined in the examples. Or we can define the following aggregate:

\[
\begin{align*}
\text{single}(\text{med}, X, i). \\
\text{mult}(\text{med}, X, C, C + 1)) \\
\text{ereturn}(\text{med}, X, C, (\text{Sal}, C + 1)). \\
\text{freturn}(\text{med}, X, C, (0, C + 1))
\end{align*}
\]

Thus, we return counts. Also we return the actual Salaries, except for the final value where we set 0 Then:

\[
\begin{align*}
\text{countall}(< \text{Sal} >) & \leftarrow \text{emp}(\text{Eno}, \text{Sal}, \text{Dno}). \\
\text{median}(\text{Sal}) & \leftarrow \text{countall}((0, C)), \text{CM} = C \div 2, \text{countall}((\text{CM}, \text{Sa})).
\end{align*}
\]

The previous definition of med uses freturn: thus it is not monotonic. Indeed, the old median is lost once we introduce new facts.
Problem 4: 24 points

Consider the following versions of Datalog (listed by increasing power):

1. Datalog without recursion, negation and choice.
2. Datalog with recursion but without negation and choice
3. Datalog with stratified negation but without choice
4. Datalog with recursion, stratified negation and choice (i.e., the non deterministic choice of $\mathcal{LDL}^{++}$ returning the first answer that satisfies the choice constraints).

For all these versions, we assume that there is no function symbols, no arithmetics, no total order in the universe.

Now, consider the following queries on a directed graph $G$ stored as a binary relation:

A. Is there an Hamiltonian circuit in $G$?
B. Is node b reachable from node a in $G$?
C. Is there an even number of arcs in the graph $G$?
D. Is the graph $G$ connected?

For each query, please, state which version of Datalog (if any) is necessary and sufficient to express it, and justify your answer.

**Answer**

B $\rightarrow$ 2: Datalog with recursion but without negation and choice can solve the reachability problem.

D $\rightarrow$ 3: The problem is nonmonotonic, so it requires negation. Then a graph is connected iff there are no un-reachable nodes. This can be expressed by the previous query and two stratified negations.

C $\rightarrow$ 4: To count the arcs in the graph one need choice to enumerate the nodes one-by-one and negation to find the last node in the path.

A $\rightarrow$ ? Deciding if there is an Hamiltonian circuit is an $NP$-complete problem. All our version of Datalog compute in polynomial time, and, assuming that $P \neq NP$, they cannot express this problem.