Name: ____

1. (1 pt.)

   - **Read all material carefully.**
   - *If in doubt whether something is allowed, ask, don't assume.*
   - You may refer to your books, papers, and notes during this test.
   - E-books may be used *subject to the restrictions* noted in class.
   - Computers are not permitted, except when used strictly as ebooks.
   - Network access of any kind (cell, voice, text, data, ...) is not permitted.
   - Write, and draw, carefully. Ambiguous or cryptic answers receive zero credit.
   - Use class and textbook conventions for notation, algorithmic options, etc.

   Write your name in the space provided above.

WAIT UNTIL INSTRUCTED TO CONTINUE TO REMAINING QUESTIONS.

Do not write on this page below this point.

Q	Full	Score
1	1	
2	9	
3	10	
4	10	
5	10	
6	10	
7	10	
total	60	

2. (9 pts.) Recall the database schema used in class for several examples:

   ```
   Students(sid, name, year)
   Courses(cid, title, ta)
   Enrolls(sid, cid, credits)
   ```

   Write a *SQL* query for the IDs and names of all students who are enrolled for at least three credits in a course with the title `Thermodynamics`.

   Explain why your query is correct.

3. (10 pts.) Write a query using the *relational or bag algebra* (as needed) that is *equivalent* to the query of Question 2.

   Explain why the two queries are equivalent.

4. (10 pts.) Write a query in *Datalog* that is *equivalent* to the query of Question 2. *Explain why the two queries are equivalent.*

5. (10 pts.) Write a *SQL* query for the total number of enrolled credits for each student in year 4. In more detail, the desired result is a list of all tuples of the form (i, n, c), where i and n are the ID and name, respectively, of a student with `year` $= 4$ and c is the sum of the credits of all enrollments (in all courses) of that student. The list should be sorted in descending order of c and (secondary sort) ascending order of n and (tertiary sort) ascending order of i.

   Explain why your query is correct.

6. (10 pts.) Write a query using the *relational or bag algebra* (as needed) that is *equivalent* to the query of Question 5.

   Explain why the two queries are equivalent.

7. (10 pts.) For the database of Question 2, we say a student (with ID) i *has TA* student (with ID) j iff i is enrolled in at least one course for which j is a TA. (Recall that the `ta` attribute of `Courses` is a student ID.)

   Further, we say i *has recursive TA* j if there is a sequence of $k \geq 2$ students $l_1, l_2, \ldots, l_k$ such that $l_1 = i$, $l_k = j$, and for all $1 \leq i < k$, l_i has TA l_{i+1}.

   Write a *Datalog* query for all pairs of student IDs (x, y) such that x has recursive TA y.

   Explain why your query is correct.

[additional space for answering the earlier question]