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. (No sharing.)
   - **E-books** may be used **subject to the restrictions** noted in class. (Briefly, do only those things with an e-book that are just as easily done with a physical book.)
   - **Computers of any kind** (including tablets, phones, and similar devices) are **not permitted** except when used exclusively as e-book readers.
   - **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.
   - Questions that ask for **explanations** allocate a sizable fraction of points to those. **Answers without sufficient explanations will score very poorly.**
   - Do not add, remove, detach, or mangle pages (causes scanner problems).
   - Budget your **time**, noting that *number of points = number of minutes*.

Write your name in the space provided above.
Do not write anything else on this page.

WAIT UNTIL INSTRUCTED TO CONTINUE TO REMAINING QUESTIONS.

(Do not view any other pages.)

Do not write on this page.
(It is for use in grading only.)

Q	Full	Score
1	1	
2	2	
3	2	
4	5	
5	5	
6	10	
7	20	
8	15	
9	5	
10	5	
11	5	
12	5	
13	20	
total	100	

2. (2 pts.) Consider a Prolog environment that encodes the edges of a directed graph with a predicate `edge(a,b)` indicating a directed edge from vertex `a` to vertex `b`. [Recall also a similar example from class discussions.] Write a Prolog query to find the out-neighbors of vertex `m`. Recall that y is an out-neighbor of x iff (if and only if) there is an edge from x to y.

3. (2 pts.) In the Prolog environment of Question 2, define a unary predicate `vertex` such that `vertex(X)` is true iff there is an incoming or outgoing edge incident on `X`.

4. (5 pts.) In the Prolog environment of Question 2, define a binary predicate `twoHops` such that `twoHops(X,Y)` is true iff there is a directed path of length two (composed of two edges) from `X` to `Y`.

5. (5 pts.) In the Prolog environment of Question 2, define a unary predicate `triCycle` such that `triCycle(X)` is true iff there is a directed cycle of length three (composed of exactly three edges) that includes the vertex `X`.

6. (10 pts.) In the Prolog environment of Question 2, define a binary predicate `threePath` such that `threePath(X,Y)` is true iff there is a path of length $3k$ (composed of exactly $3k$ edges), for some nonnegative integer k, from `X` to `Y`.

   Explain briefly why your answer is correct.

7. (20 pts.) Consider the following grammar (using *yacc*/*PLY* syntax):

```
S : A B
A : a b | a A B b
B : b b B | b b | A
```

Does the sentence `abaabbaabb` belong to the language of this grammar? If it does then provide a corresponding *leftmost derivation* and a *parse tree* using class and textbook conventions; otherwise, provide a *proof* (as precise as possible) that it does not.

[additional space for earlier material]

8. (15 pts.) This question centers on the issue of *scope* as discussed in class and the textbook. What is the output of the following Python program? (If there is an error then explain the error in detail.) **Explain your answer using concepts related to scope.**

```
def main():
    x = 1
    y = 2
    z = 3

    def f(x):
        y = 20
        z = 30

        def g():
            z = 300

            def h(z):
                return x + y + z

            return h

        return g

    x = 1000
    y = 2000
    z = 3000
    print(f(3)()(5))


main()
```

[additional space for earlier material]

9. (5 pts.) For the following *Standard ML* expression, provide the response when that expression is evaluated by the `sml` REPL (read-eval-print loop). Assume that the expressions are evaluated in the order listed. In your response, *draw a* ***box*** *around the* ***type*** *and* ***`oval`*** *around the* ***value****.* (If there is an error then clearly explain the error.) *Explain* briefly why your answer is correct.

```
1     42.42 / 2.0 * 3.0;
```

10. (5 pts.) Repeat Question 9 for the following:

```
1     fun f501 (x::y::z) = y :: x :: f501(z)
2     | f501 (w) = w;
```

11. (5 pts.) Repeat Question 9 for the following:

```
1    f501([]);
```

12. (5 pts.) Repeat Question 9 for the following:

```
1    f501([10, 20, 30, 40, 50]);
```

13. (20 pts.) Provide a **complete JCoCo assembly language program** that

    (a) Reads two newline-terminated strings from *standard input* and interprets them as numbers in the usual way. (For example, the string "142" is interpreted as the integer 142.) Let these two numbers be a and b (in input order).

    (b) Writes, to *standard output*, exactly $\max(0, b - a)$ newline-terminated lines, where the k-th line is the value of the sum of the squares of the first k integers starting with a. (For example, if $a = 3$ and $b = 7$ then four lines of output are produced, with `9`, `25`, `50`, and `86`, respectively (since $3^2 = 9$, $3^2 + 4^2 = 25$, etc.).

    Explain why your program is correct.

[additional space for earlier material]