Question 1. (0.75 points)
Discuss the four standard isolation levels and the anomalies each level prevents. Give one example for each anomaly.

Question 2. (0.75 points)
Compare optimistic concurrency control techniques and pessimistic concurrency control techniques. What are the advantages and disadvantages of each approach?

Question 3. (0.50 points)
Consider the schedule of operation of three transactions that will be executed concurrently.

\[ R_1(A), R_2(B), R_3(A), W_1(A), R_1(C), R_2(C), W_3(C), W_2(A), R_2(A), R_1(D), W_1(D) \]

(a) Is this schedule serial?
(b) Give the precedence graph of this schedule.
(c) Is this schedule conflict serializable?
(d) If the answer is ‘yes’ to (c) provide the equivalent serial schedule. If you answer ‘no’, briefly explain why.
Question 4. (1.00 point)

Consider the following schedules, where R, W, C stands for ‘Read’, ‘Write’ and ‘Commit’ respectively and subscripts refers to transactions (i.e., $R_1(X)$ means transaction $T_1$ read $X$):

**Schedule 1:** $R_1(X), W_1(X), R_2(X), W_2(X), R_1(Y), W_1(Y), C_1, R_2(Y), W_2(Y), C_2$

**Schedule 2:** $R_2(X), W_2(X), R_3(Y), W_3(Y), R_3(Z), W_3(Z), C_3, R_2(Z), W_2(Z), C_2, R_1(X), W_1(X), C_1$

**Schedule 3:** $W_1(A), W_2(B), W_3(C), R_1(X), R_2(X), R_1(Y), W_1(Y), C_1, W_2(Y), C_2, W_3(Y), C_3$

For each schedule answer the following question:

(a) Is this schedule conflict serializable? If yes, provide an equivalent serial schedule of transactions.

(b) Does the schedule avoid cascading aborts? Explain your answer.

(c) Is it possible under **rigorous 2PL**? Explain your answer. Remember that the 2PL protocol acquires an exclusive lock for writing and a shared lock for reading. In rigorous 2PL, the locks (shared and exclusive) are only released once the transaction has been committed or aborted. If rigorous 2PL leads to a deadlock, give the wait-for graph.