Data-Intensive Computing Systems

- This exercise will not be graded. Solutions will be posted once you get a chance to work on this exercise.

Question 1

Suppose a database system reboots after a crash and finds that both A and B on disk have the value 10. The log is found to be:

```
<START T1>
<START T2>
<T1, A, 5>
<COMMIT T1>
<T2, B, 5>
<T2, A, 15>
[CRASH]
```

1. If the system is using UNDO logging, then give the initial state of the database before T1 and T2 began executing (i.e., what were the initial values of A and B on the disk?).
2. If the system is using UNDO logging, then what will be the final state of the database after recovery (i.e., what will be the values of A and B on the disk after the recovery process has finished?).
3. If the system is using REDO logging, then give the initial state of the database before T1 and T2 began executing (i.e., what were the initial values of A and B on the disk?).
4. If the system is using REDO logging, then what will be the final state of the database after recovery (i.e., what will be the values of A and B on the disk after the recovery process has finished?).

Question 2

Assume that a database system using UNDO/REDO logging and nonquiescent checkpointing crashes with the log records on disk given below. Record $< T, X, v, w >$ means that transaction T changed the value of database element X; its former value was v, and its new value is w.

```
<START T1>
<T1, X, 14, 28>
<T1, Y, 15, 5>
<START T2>
<T2, Z, 20, 10>
<COMMIT T1>
<START CHKPT (T2)>
<T2, W, 4, 7>
<START T3>
<END CHKPT>
<T3, X, 28, 17>
<COMMIT T2>
```

1. What are all of the possible values on disk for each of the database elements W, X, Y and Z?
2. Which, if any, transactions will need to be redone in the recovery process?
3. How would your answers to parts (1) and (2) change if <END CHKPT> were not present in the log?

Question 3

Consider the following transaction log from the start of the run of a database system that is capable of doing UNDO/REDO logging with checkpointing:

```
1) <START T1>
2) <T1, A, 45, 10>
3) <START T2>
4) <T2, B, 5, 15>
5) <T2, C, 35, 10>
6) <T1, D, 15, 5>
7) <COMMIT T1>
8) <START T3>
9) <T3, A, 10, 15>
10) <START CHKPT (T2, T3)>
11) <T2, D, 5, 20>
12) <COMMIT T2>
13) <END CHKPT>
14) <START T4>
15) <T4, D, 20, 30>
16) <T3, C, 10, 15>
17) <COMMIT T3>
18) <COMMIT T4>
```

Assume the log entries are in the format <Tid, Variable, Old value, New value>. What are the values of the data items A, B, C, and D on disk after recovery:

1. If the system crashes just before line 6 is written to disk?
2. If the system crashes just before line 10 is written to disk?
3. If the system crashes just before line 12 is written to disk?
4. If the system crashes just before line 13 is written to disk?
5. If the system crashes just before line 16 is written to disk?
6. If the system crashes just before line 18 is written to disk?

Question 4

Schedule S1 is said to be *conflict-equivalent* to schedule S2 if S2 can be derived from S1 by a sequence of swaps of non-conflicting actions. For example, the schedule S1 = r1(A), r2(A), w2(A), w1(A), r2(B), w2(B) is conflict-equivalent to the schedule S2 = r2(A), r1(A), w2(A), r2(B), w1(A), w2(B), since S2 can be derived from S1 as shown below:

```
S1 = r1(A), r2(A), w2(A), w1(A), r2(B), w2(B); swap(r1(A),r2(A))
   = r2(A), r1(A), w2(A), w1(A), r2(B), w2(B); swap(w1(A), r2(B))
S2 = r2(A), r1(A), w2(A), r2(B), w1(A), w2(B)
```

Prove or disprove each of the following statements.

1. If two schedules are conflict equivalent, then their precedence graphs are identical.
2. If two schedules involve the same set of transactions, and have identical precedence graphs, than they are conflict equivalent.