1. Here you are to time two different configurations of the buffer to perform an external merge sort. The disk will have a seek and latency time totaling 9ms and a block transfer time of 1ms. You will be sorting a file of 2,000,000 pages with a buffer of 32 pages. Determine the times for the buffer configurations below – do not include the time used in the first pass where you create 2,000,000/32 = 62,500 runs of 32 pages each.
   a) Use 3 input buffers and 1 output buffer, each of 8 pages.
   b) Use 6 input buffers of 4 pages each, and 1 output buffer of 8 pages.

2. Consider the two relations
   \textit{office} (office\#, otype, ophone\#, ofloor)
   \textit{employee} (essn, efname, elname, eaddress, eoffice\#)

   With a page size of 2k, the \textit{office} table has 8,000 records each of 100 bytes. At 20 records per page, the \textit{office} table fills 400 pages. The \textit{employee} table has 20,000 entries each of 200 bytes. That gives 2000 pages at 10 per page.

   The goal here is to determine the amount of disk i/o to compute the join of \textit{office} and \textit{employee}. You do not need to consider the disk i/o required to write the outcome of the join, just that to compute it. Use the following techniques.
   a) sort-merge join with a 52 page buffer
   b) index loop, \textit{employee} outer with an index on office\# of the \textit{office} table (it is a B+-tree of depth 3)
   c) index loop, \textit{office} outer, with an index on eoffice\# of the \textit{employee} table, also of depth 3

3. Consider the query using the schema of the previous problem
   \begin{verbatim}
   SELECT efname, elname, ophone#
   FROM office, employee
   WHERE office#=eoffice# AND floor=4;
   \end{verbatim}

   Write an optimized relational algebra query tree for this SQL query.

4. See the other side
4. Show the recovery process in the event of a crash given that the following logical log is found on disk at the time of restarting. Extend the log to show the prevLSNs and undoNextLSNs. Show the log records that are added after the crash during recovery.

```
00  checkpoint
10  update: t1 writes p1
20  update: t2 writes p1
30  update: t3 writes p1
40  update: t1 write p3
50  abort: t1
60  update: t2 writes p2
70  CLR: undo t1 LSN 40
80  update: t4 writes p3
90  commit: t3
100 CLR: undo t1 LSN 10
110 end: t3
120 abort: t4
130 update: t2 writes p1
140 end: t1
```

5. [extra credit, not very much] Write the symbol for a semi-join.