Tutorial 1

1. What measure do we use when analysing performance of out-of-core algorithms? Why? What idealising assumption do we make by doing so?

2. In algorithm analysis, the performance of the algorithm is usually measured as a function of the size of input. For out-of-core algorithms, we measure the performance as a function of the size of the input, N, and the size of the available RAM, B. In what units are N and B measured? Why?

3. What should the relationship between the size of the RAM and the size of the input file be for the sorting to be done in at most 2 passes?

4. Assume that the page size is equal to the number of bytes able to hold two integers; further assume that the RAM size is equal to the number of bytes able to hold 8 integers. Given these assumptions, step through the external merge-sort algorithm applied to the input files containing the following sequences of numbers and determine how many passes are needed to sort these files:

   (i) $10, 7, 1, 8, 27, 8, 7, 9, 3, 53, 45, 6, 2, 100, 52, 12, 100, 101, 21, 15, 7, 23, 1$;

   (ii) $17, 15, 3, 2, 3, 2, 17, 19, 19, 1, 7, 21, 5, 16, 15, 11, 7, 22, 19, 3$.

5. Consider the following 3 scenarios: you have

   (a) a file whose size equals 10,000 pages and RAM whose size equals 3 pages;

   (b) a file whose size equals 20,000 pages and RAM whose size equals 5 pages;

(c) a file whose size equals 2,000,000 pages and RAM whose size equals 17 pages.

For each of the scenarios above, answer the following questions about the external merge-sort algorithm (do not worry about the exact number, perform as much of the calculation as you can without the use of a calculator):

(i) How many runs will the algorithm produce in the initial pass?

(ii) How many passes will it take to sort the entire file?

(iii) What is the total cost of sorting the file?

(iv) What is the size of the memory needed to sort the entire file in two passes?

6. Consider a relation containing information about university students with the following schema:

   ```
   students (sid:  integer, sname:  varchar(50),
   street:  varchar(50), city:  varchar(30), age:  integer)
   ```

   Assume that the students file consists of 5,000 pages.

   (i) Let B be the number of pages that fits into the available RAM. If B = 50, how many passes (including pass 0) of the external sort-merge algorithm will be required to sort the relation and how many runs will be produced by each of the passes?

   (ii) For the set-up from the previous question, how many I/Os will be required? Assume that the relation is originally on disk and that the result of the sorting operation must also be written to disk.

   (iii) Now, consider the query `SELECT sid, age FROM Students ORDER BY age`. Briefly describe how we could execute this query in a way that would result in the lowest possible cost of sorting the `Students` relation? Does your approach save disk reads, disk writes, or both?

7. Consider a simple (non-recursive), 2-pass disk-based hashing strategy described in the lecture, where in the first pass the file is partitioned and in the second pass each of the partitions is hashed.

(i) Ignoring any space overhead for building hash tables and assuming a perfect hash function, what is the minimum size of RAM needed to hash the above `students` relation in 2 passes?

(ii) For the previous question, how many I/Os will be required? Assume that the relation is originally on disk and that the result of the hashing operation must also be written to disk.