The goal of this project is to implement and experiment with an I/O-efficient heap and with external heap-sort. The project should be done in the same groups as project 1. As project 1, it should be programmed in C or C++. The project is intentionally left somewhat open, and an important part of the project is to write a report that describes your implementation (including your implementation/design choices) and experimental work; The report (including a pointer to the implementation) should be handed in by Tuesday April 14, 2009. The evaluation of the implementation, experimentation and the report will be part of the final grade.

As in project 1, you should use the Heap, Heapsort and Quicksort code from the Sedgewick book or from STL.

Tasks:

1. Implement the I/O-efficient heap described in the paper by Fadel et al. [1]. You do not have to follow their description exactly, as long as the I/O-complexity of the insert and deletemax operations remain $O(\frac{1}{B} \log_{M/B} \frac{N}{M/B})$ amortized. Try to make your implementation as practically efficient as possible.

   Implement the I/O-specific part of the data structure (particularly partial merge) in at least three different ways (similar to project 1):

   (a) Using read and write (with or without read/write buffer blocks).
   (b) Using fread and fwrite.
   (c) Using mmap and munmap.

   In each implementation, make sure that you can easily experiment with different tree fan-outs, and node and insert buffer sizes, as well as with different (logical) block sizes $B$ (in the implementations where you explicitly use $B$).

2. Implement an External-heap-sort algorithm based on your I/O-efficient heap for sorting 32-bit integers.

3. Experiment with each of your heap implementations in the External-heap-sort, that is, e.g. try sorting with different fan-outs, node sizes, insert buffer sizes, and block sizes. Make sure to include a discussion of the results of your experiments in the report.

4. Compare the best of your External-heap-sort algorithms with (normal internal memory) Heap-sort and Quicksort, as well as with the best of your Merge-sort algorithms from project 1. Again discuss the results in the report.

References