CMSC 424, Spring 2012: Homework 3 (Due: April 17, 2012)

1.

For the B+-Tree shown above (with at most 4 pointers per node), show the effect of following operations, in sequence:

(a) insert 80
(b) insert 250
(c) insert 280
(d) delete 550

2. Suppose that we are using extendable hashing on a file that contains records with the following search-key values:

3, 8, 11, 14, 15, 16, 17, 19, 20, 33, 43, 48

Show the extendable hash structure for this file if the hash function is $h(x) = x \mod 7$, and buckets can hold 3 records each.

3. The hash join algorithm described in the textbook computes the natural join of two relations. Write pseudocode for an iterator that implements a hash-based algorithm for computing an “anti-join” operation: $R \text{ anti-join } S$ contains all tuples in $R$ that have no matching tuples in $S$. Although anti-join is not an operation that can be specified directly in SQL, it is implemented in the query processor to support some types of queries. Your pseudocode must define the standard iterator functions $\text{open}()$, $\text{next}()$, and $\text{close}()$, and must fetch the input tuples using iterator calls (in other words, $R$ and $S$ may not be base relations, but may be outputs of other operators). Show what state information the iterator must maintain between calls. For simplicity, you can assume that $S$ is small and easily fits in memory. Here are more details on anti-join: http://en.wikipedia.org/wiki/Relational_algebra#Antijoin

4. We would like to monitor the stock closing prices of companies as they change in time. We use the relation $\text{Stock}(\text{co}, \text{date}, \text{val})$ to store the closing value ($\text{val}$) of the stock of each company ($\text{co}$) at the end of each $\text{date}$. This is a historic relation that at this time observes the following statistics:

$n(\text{Stock}) = 10,000,000 \quad \text{-- # tuples in the relation}$

$b(\text{Stock}) = 100,000 \quad \text{-- 100 tuples per page}$
(a) Estimate the I/O cost required to do a block-oriented nested-loop join with a 2-block buffer memory for the query:
   
   ```sql
   select *
   from Stock S1, Stock S2
   where S1.co = S2.co
   ```

(b) Do the same as in (a) with a 1001 block buffer.

(c) Estimate the I/O cost required to do a hash join for the query in (a).