CS 143
Lab 1 Report:

Aanish Sikora 804028077
Tushar Shrimali 804070047

For all of the provided code, no argument types/numbers and return values were changed. The only change to the API was the inclusion of private iterator classes within: HeapFile, HeapPage and Tuple. This is because the default iterator for most data structures was not applicable and additional functionality was needed.

ITERATORS:

HEAP FILE: The HeapFileIterator implements DbFileIterator which has open, hasnext, next, rewind and close methods. The iterator must go through all tuples stored on the file. Therefore, the iterator must use database's global getBufferPool method which is used to get the page through the getPage method which is casted to a HeapPage. It is required that this class contains the member variables of a Heapfile, integers to track the number of pages on this file and the number of current pages, a HeapPage and an tuple iterator. The hasNext and next methods utilize the tuple iterators defaults. The open method gets the page from the bufferpool and intializes the iterator.

HEAP PAGE: The HeapPageIterator implements the tuple iterator. The iterator goes through all the tuples on the HeapPage but doesn't return tuples in empty slots. Therefore, the member variables are a List of tuples and a list iterator. This data structure was utilized as it allows easy iteration through all the tuples in the page and has a built in iterator class.

TUPLE: The FieldIterator implements field iterator and goes through all the fields of a particular tuple. Therefore, it contains member variables corresponding to the Tuple being iterated through and integers representing the number of fields and the current field. As a result, the hasnext method checks if the current field is less than the number of fields. Also, the next method simply increments the current field.

GENERAL DATA STRUCTURES USED:

TUPLE: The member variables needed are the TupleDesc corresponding to the Tuple, the RecordId to identify the tuple and an array of fields filled with data.

TUPLEDESC: The member variables are an array of type and strings which correspond to the types and names of each field.

BUFFERPOOL: The member variable is a hashmap keeping pageId's and Pages. This allows for easy page retrieval given a page ID.

CATALOG: There is a private member class called Table which contains a name and private key. ConcurrentHashMaps are used to store Table and Files and FileID and Tables. This allows for easy access to Tables and Files given a fileID.

HEAPPAGE: Private variables include pid (to store the heap page id), td (current tuple descriptor), header array to store the head for the page, tuple array of type tuples and numSlots (total number of slots on the page). All functions except isSlotUsed are very straightforward and simply return values. isSlotUsed calculates the byte in the header (a bitmap) that contains the bit for the tuple. This is &ed with the 0s containing 1 at the bit poisition. If the resulting value is 1, the slot is used.

HEAPFILE: HeapFile uses a private file readpagefile of Java type RandomAccessFile opened with read permissions only. In readPage, readpagefile navigates to the correct offset with seek() and reads into the byte buffer. the class also implements a private class HeapFileIterator as described perviously.

SEQSCAN: Most of the methods of seqscan are implemented through the heapfile iterator (dbit). dbit is of type DbFileIterator and is called through heapfile.iterator (calls hasnext, next, rewind, etc.). Other private variables include tablealias, _tableid, file (of the DbFile) and heapfile (of type HeapFile).