Report of Database Lab1

Name: Nayun Xu
Student ID: 515030910635

1.Design decisions

1.1 Iterator

For the Iterator part, the iterator in ***HeapPage.java*** just needs a cursor to locate the current tuple. The idea is trivial. And the iterator in ***HeapFile.java*** to iterate through the heapfile is slightly more difficult. I use a cursor for current page, and a iterator for tuples in the current page which I implement it in ***HeapPage.java***. This helps me avoid writing the same code twice. And for the ***SeqScan.java***, I simply use the iterator I implement in ***HeapFile.java***.

1.2 BufferPool.getPage()

In this part, the requirement says that there should be a lock to make this function available for multiple transactions. I use the ***ReentrantLock*** object provided by Java to implement this because I forget the associated knowledge learnt at the Computer System course.

2.Changes on API

I add a class into ***HeapPage.java*** to help implement the iterator to iterate through tuples in a page.
I add some public methods into some of the classes to help debug. These public methods are used to get access to some private members, most of which are used to debug iterators.

3.Incomplete elements

The hashcode I use is the default hashcode method provided by Java objects. And I implement none of the features that are not required in Lab1. The lock in ***BufferPool.getPage*** part is also not used in current test, and I don't know how to verify its correctness.

4.Experience

I spent about 15 to 20 hours(not exact numbers) on this lab including learning to use Junit and ant, dealing with bitbucket, and fighting against my pool Internet.
The ***HeapFile.readPage*** part was the first challenge. I learnt how to read datas from disk using ***RandomAccessFile***. And I found the encoding is different from what I had thought. For example if the first byte of the header is 11110000, it means slot 4,5,6,7 is used(I thought it would be 0,1,2,3). And I did n = n&(n-1) until n becomes 0 to count the number of 1 bits. It fails when the number is negative, so my method crashed.
The requirement about iterator ***open()*** and ***close()*** confused me. I don't know whether the iterator should be reset when it is closed or it should be restored after it being open again. And what exceptions to throw in the iterator part is also confusing. The ***Dbterator*** throws ***IllegalStateException()*** when calling a closed iterator's methods . But the

requirement in ***DbfileIterator*** doesn't contain this exception, it requires to throw a ***DbException()*****.**
I cannot fully understand the lock in BufferPool.getPage(). Maybe I will understand it in later labs.