This assignment is a collaboration of Megan Fantes(U56999246) and Efstathios Karatsiolis(U65214507)
Lab 1
Design Document

Describe any design decisions you made. Discuss and justify any changes you made to the API. Describe any missing or incomplete elements of your code.

- For the TupleDesc class, we created a new class called TupleDescIterator and implemented it in its own file. TupleDescIterator implements the java class Iterator, so we wrote the hasNext() and next() methods for TulpeDescIterator to override the abstract methods so they are more specific to iterating over Tuple Descriptors. We implemented TupleDescIterator in its own file just to have cleaner code.

- For the Tuple class, we created a new class called FieldsIterator and implemented it in its own file. FieldsIterator implements the java class Iterator, so we wrote the hasNext() and next() methods for TulpeDescIterator to override the abstract methods so they are more specific to iterating over Tuple Descriptors. We implemented FieldsIterator in its own file just to have cleaner code.

- For the Catalog class, we implemented 3 parallel Hashmaps:
    1) Key = table name, Value = DbFile with table data
    2) Key = table name, Value = primary key of the table
    3) Key = table id, Value = table name

  How we implemented this:
  These 3 hashmaps are all initialized when Catalog is created, and as Tables are added to the Catalog, all 3 hashmaps are updated accordingly. If a Table is added and the table name already exists in the Catalog, the new DbFile replaces the old DbFile at the name entry in the <name, DbFile> hashmap, the new primary key replaces the old primary key in the <name, primary key> hashmap, and finally the old <table id, name> entry is removed from the <table id, name> hashmap and the new <table id, name> entry is added with the table id of the name table.

  Why we implemented Catalog like this:
  Hashmaps are a very efficient data structure for storing and retrieving data, because of the use of [key, value] pairs. We needed each DbFile to be searchable by its name, so we created hashmap #1. We also needed to be able to find the primary keys for each table, so we created hashmap #2. Then we needed each table name to be associated with its corresponding table id, so we created hashmap #3.

  Possible issues:
  Concurrency. It is difficult to make sure the hashmaps are in parallel with each other, and that each is always updated at the same time in the same way. We had to ensure that each update and retrieval method always accessed the hashmaps in the same way and at the same rate.

- For the BufferPool class, we only implemented the getPage() method and left the rest of the methods blank.

Describe how long you spent on the lab, and whether there was anything you found particularly difficult or confusing.

- The javadoc for the lab was not very helpful. It did not contain any information that was not already in the comments of the code. We were often left wondering what a method was supposed to accomplish, and what other classes it was supposed to interact with.
- This assignment took us about 40 hours of coding between 2 people.
- The architecture of the SimpleDb was quite confusing. We might have needed an extra lab explaining to us how we should implement core functionalities of the system.