LAB 1
Emmanuel J Lopez 1005407
Nicholas Gandhi Peradidjaya 1005295

1. **Explain any design decisions you made.**
   Error Handling: We included comprehensive error handling mechanisms throughout the codebase to handle exceptional scenarios. This ensures that the system can handle unexpected situations robustly and provides meaningful feedback to users.

2. **Explain the non-trivial part of your code.**
   Error Handling Logic: Developing robust error handling logic involved identifying potential error scenarios and implementing appropriate exception handling mechanisms. This included checking for invalid inputs, handling file I/O errors, and responding to transaction-related exceptions such as transaction aborts or deadlocks. For example, in HeapFile.java, we catch IOException error in readPage method.

3. **Discuss and justify any changes you made to the API.**
   In HeapFile.java, we implemented DbFileIterator interface as an inner class HeapFileIterator. Implementing the iterator is essential for iterating over the tuples in the heap file. Therefore, an inner class HeapFileIterator was introduced to implement the DbFileIterator interface, providing the functionality needed to iterate over the tuples in the heap file.

4. **Describe any missing or incomplete elements of your code.**
   The code needs better logging and recovery features to maintain durability and consistency, especially when dealing with system failures. It's essential to implement logging to track transaction changes and recovery mechanisms to restore the system to a stable state after a crash, especially for production purposes.