CSE 510 : Database Management Systems and Implementation

Project Phase 1 Report

By Himali Gajare (1228110730)

Group 2

Himali Hemant Gajare

Aashka Parikh

Kshitij Dhyani

Maneesha Guntur

Rigved Alankar

Abstract

The report provides a thorough analysis of the outcome of the tests, highlighting the strengths and weaknesses of the Java implementation of the Minibase database management system. The system encompasses interactive tests that gauge the performance of the code's B+ tree structure in response to a range of inputs, and non-interactive tests that assess the various subsystems of the larger Database Management System (DBMS) in the face of intended and unforeseen inputs. The results of the tests showed that the Minibase system was functioning as expected, with no significant issues detected. This report serves as a valuable resource for those who want to understand the capabilities of the Minibase system and how it can be used for database management. It provides important insights into the system's strengths and weaknesses and can be used as a reference for future modifications and updates to the system. Overall, this report offers a comprehensive evaluation of Phase 1 of the project and sets the stage for future phases of the project.

Keywords: Minibase, Database Management System, B+ Trees, Buffer Management, Heap, Disk Management.

Introduction

This report provides a comprehensive analysis of the outcome of the tests conducted on the Java implementation of the Minibase database management system. The objective of these tests is to evaluate the performance of the system's B+ tree structure in response to various inputs and to assess the various subsystems of the larger Database Management System (DBMS) in the face of intended and unforeseen inputs. The report presents a thorough examination of Phase 1 of the project. The summary of all the concepts implemented in Minibase is outlined below.

Buffer management refers to a collective set of strategies that dictate what data is held in memory and for how long. The two basic components of any buffer management policy are buffer allocation and buffer replacement [4]. Buffer allocation strategies attempt to allocate buffer frames to transactions, while buffer replacement policies attempt to identify victim frames, i.e., candidate frames for replacement. In non real-time systems the goal of buffer management is to reduce transaction response times. In real-time systems a good buffer management strategy should reduce the number of transactions missing their deadlines [4].

The Disk Space Management System is the lowest layer of Database Management System (DBMS), managing space on disk [1]. It is an integral component of the Logical Layer of a DBMS and is responsible for the allocation and deallocation of disk space as well as performing read and write operations. These functions are crucial for the addition or removal of data within a database.

A heap file is a type of file organization used to store data in a database management system (DBMS). In a heap file, data is stored in a collection of pages, and the order of the pages is unimportant. This means that there is no fixed order or structure to the data stored in a heap file, and data can be inserted, deleted, or updated at any position within the file [1].

The B+ tree is a highly utilized index in modern database systems. Each node in the tree, represented as a disk page, typically has a size of 4096 bytes. The leaves of the tree hold the database records, while index pages located above the leaf level contain only index records. The search for a single database record involves accessing exactly one node at each level of the tree. Due to its organization of clustering data in leaf pages in the order of the indexing attributes, the B+ tree is optimal for key-range searches. Additionally, the B+ tree is dynamically organized, ensuring that the height of the tree remains balanced even after record insertions or deletions [3].

The process of searching within a B+ tree consists of two fundamental steps. Firstly, the search for the correct key is conducted within a node. Secondly, the search continues recursively by following the pointer associated with the key, until a leaf node is reached. The first step offers the opportunity for parallelizing the search for the correct key, making it a crucial aspect of the search process [2].

Non Interactive Tests

Buffer Space Management System

Minibase includes three buffer manager tests as a component of its automated testing procedure.

The first test evaluates the basic operations by allocating a quantity of pages that exceeds the number of unpinned pages by one. The test then proceeds to unpin a page that was previously pinned, thus resulting in all pages in the buffer pool becoming unpinned. The test subsequently endeavors to write on all pages present in the pool, which requires pinning them prior to making any modifications and unpinning them once the changes have been implemented. Finally, the test requires pinning the pages once again to verify that the new data has been updated and then unpinning all pages in the buffer.

The second test focuses on invalid actions with the expectation of failure. The first scenario involves attempting to pin more pages than the number of frames available in the buffer, which is expected to result in failure due to insufficient space in the buffer to accommodate the pages. The subsequent scenario involves attempting to release a page from the buffer that has already been doubly pinned, which is not permitted as it would result in a decrease in the pin_count of the page. Finally, the test will attempt to pin a page that is not present in the buffer pool, which is expected to result in failure.

The third test begins with the allocation and pinning of new pages, followed by writing of new data into them. Subsequently, some pages are unpinned, instead of all of them. In order to retrieve and read the pages, they must be pinned again, potentially resulting in some pages having two pins. The test concludes with the unpinning of all pages, followed by the additional unpinning of those pages that initially had two pins.

Disk Space Management System

The MiniBase disk management system performs four tests to evaluate its performance and functionality.

The first test focuses on the standard disk space operations such as allocating pages, writing to pages, and deallocating pages to free up some space in disk for future operations. The goal of this test is to verify that the disk space management system can perform these fundamental operations correctly and efficiently.

The next test expands upon the operations performed in the first test by including the ability to read, search for file entries, and delete file entries. It is to verify that the disk space management system can correctly handle these operations, which are essential for retrieving, searching, and removing data from the database.

The third test evaluates error conditions by performing illegal operations that are intended to fail. Some of the error conditions include retrieving previously deleted file entries, deleting the previously deleted file entries again, retrieving/deleting an imaginary file entry, inserting file entries having too long length, and many more. The objective of this test is to verify that the disk space management system can handle and detect these error conditions, and prevent the database from being corrupted.

The last test checks boundary conditions, such as allocating one extra page than the space available on the disk. Additionally, the test involves inserting file entries that exceed the capacity of a page, thereby necessitating the creation of a new page. Subsequently, attempting to allocate the same number of pages again is expected to result in an OutOfSpace exception being thrown. This assesses the performance of the disk space manager in these scenarios.

Each test is designed to assess a different aspect of the disk management system, and the results of all four tests combined provide a comprehensive evaluation of its performance and functionality.

Heap Files

In Minibase, tests for heap files in a DBMS typically focus on evaluating the performance and efficiency of the heap file implementation in terms of various key metrics, such as insertion and retrieval speed, data consistency, file organization, and disk space utilization.

The first test focuses on the insertion of 100 records in the file. It will write the new record to the page, updating the directory header information to indicate the new size of the page and the number of valid records. It will further mark the page as dirty, so that it will be written to disk when it is evicted from the buffer pool. When the page is eventually written to disk, the new record is permanently stored in the file.

In the second test, odd records are deleted from the Heap file. Deleting records from a Heap file involves marking the page containing the record as dirty. The record is then deleted from the page. This triggers an update to the directory header information, such as the number of valid records and free space on the page. When the page is eventually flushed from the buffer pool, the deleted record is permanently removed from the file. Finally, a scan of the records is performed to confirm the deletion.

The focus of the third test is to retrieve all records from the Heap file, which are composed of only even records, following the deletion of all odd records in the preceding test. Subsequently, the values of the tuples are modified by a factor of 7n. The test continues by retrieving all records once again, and then validates whether the values of the tuples have indeed been altered.

The fourth test examines the modifications made to the record length and its potential to raise exceptions. It is imperative to note that in Heap file implementation in Minibase, all records must have the same length. However, in this test, the record length is altered either to a length less than or greater than the required length, leading to the occurrence of exceptions.

Index Files

The first test involves the creation of a new heap file, followed by the insertion of records (tuples) into the heap file. A BTree index file is then created and the key and record id of each record is added to the index file. The data obtained from the index file is compared with the sorted data. The outcome of this comparison confirms that the index file contains properly sorted data.

The second test assesses the functionality of scans in index files, which include two types: identity scan and range scan. An identity scan aims to search for a specific word in the index file and retrieve the corresponding record from the heap file. A range scan retrieves all records within a specified range from the heap file.

The third test consists of inserting 1000 records into a heap file, followed by the creation of a BTree index file with integer values as keys. An index scan is then performed between the range of 100 to 900 to retrieve the records, which are compared with a sorted set of data. This test is similar to the first test, however, it utilizes an “Integer” key rather than “String” type of key.

Sort Test

The sort test in Minibase is used to evaluate the performance of the sorting component of the system. The first test sorts the records in ascending order and runs a check to ensure that the sorted records are the same as the expected records order. In the second test, it sorts the records in the descending order, later checks with the sorted records. , it sorts in descending order of the float field, and in ascending order the int field. In the last test, the sorting algorithm on multiple files rather than the previous tests which only sorted one file.

Join Test and SortMerge Test

The Nested Loops Join method involves comparing each tuple in one table with every tuple in the other table through Nr x Ns iterations. In contrast, the Sort Merge Join utilizes a marker and comparison scheme to join tables, resulting in improved performance for larger datasets. Queries 2 and 6 in the code utilize the Nested Loops Join method, while the remaining queries employ the Sort Merge Join. The output from the query execution is subsequently verified using the Querycheck function. The SM_JoinTest.java file implements these same queries for functional testing purposes.

Interactive Tests

The interactive component of the DBMS tests involves the user making various queries from a menu to assess the performance of the BTree functionality within the database management system. During these tests, temporary files in the format of AAAX, where X is a non-negative integer, are generated to store the data being tested. The interactive menu comprises 20 distinct commands, each of which is described in detail below.

1. Both options, the 0th and 1st choice, create a new file and modify the way entries are deleted in it. The BTreeFile class has two algorithms for deleting files: the naive delete algorithm or the full delete algorithm. The naive delete algorithm removes a designated data entry iteratively without rearranging the file, while allowing duplicates. The full delete algorithm, on the other hand, deletes specific entries via a recursive process and reorganizes the BTree. If the tree is empty, it deletes the root.
2. This option outputs the structure of the B+ tree created from all its elements. The first line displays "1" and the root node’s id, followed by the rest of the tree's pages in a hierarchical manner.
3. This option displays the contents of the BTree by printing each of its leaves, which are represented as <key, rid> pairs.
4. This option will display all the contents of the specified page. It can be either Index page or Leaf page.
5. This option inserts an integer value as an entry into the BTree of the current file.
6. This option deletes an integer value from the BTree of the current file, if it is present in the file.
7. This option inserts n records in a sorted order (ascending), starting from 0 to n-1, into the file.
8. This option inserts n records in a sorted order (descending), starting from n-1 to 0, into the file.

9. This option inserts n records randomly.
10. This option inserts n records randomly and deletes n records randomly from the file.
11. This option deletes any records whose key value falls within a specified range.
12. This option enables the user to designate an integer range for elements to be scanned and displayed in order as they are scanned.
13. This option retrieves the next element in the current scan.
14. This option removes the last element that was scanned.
15. This option is similar to option 10, with the exception that it uses string-based keys.
16. This option terminates the currently open AAAX file. It does not delete the file, but prevents any further insert, delete, scan, or print operations from being performed until it is reopened.
17. This option opens the AAAX file, with X representing the integer entered.
18. This option destroys the entire B+ tree file.
19. This option terminates the interactive component of the tests.

Conclusion

The focus of Phase 1 of this project was on evaluating the specific functionality and components of the Minibase DBMS. The tests carried out in this phase included evaluations of the buffer manager, disk manager, heap file management, sorting functions, and both join and sort-merge join functionalities. This phase not only familiarized the user with the interface and visualization functions of Minibase, but also introduced the important concepts and topics relevant to database management systems in general. The examination of the tests led to a deeper understanding of the limitations and capabilities of not only Minibase, but also of database management systems in general.

Bibliography

1. Ramakrishnan, Raghu., and Johannes. Gehrke. *Database Management Systems* .3rd ed. Boston: McGraw-Hill, 2003.
2. D. Heinrich, S. Werner, M. Stelzner, C. Blochwitz, T. Pionteck and S. Groppe, "*Hybrid FPGA approach for a B+ tree in a Semantic Web database system*," 2015 10th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), Bremen, Germany, 2015, pp. 1-8, doi: 10.1109/ReCoSoC.2015.7238093.
3. I. Jaluta, "*Transaction management in B-tree-indexed database systems*," 2014 International Conference on Information Science, Electronics and Electrical Engineering, Sapporo, Japan, 2014, pp. 1968-1975, doi: 10.1109/InfoSEEE.2014.6946267.
4. Datta, A., Mukherjee, S., & Viguier, I. R. (1999). *Buffer management in real-time active database systems*. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 29(2), 216–224. https://doi.org/10.1109/3468.747857

Appendix

```
Last login: Mon Feb  6 08:59:18 on console
(base) himaligajare@MacBook-Air src % script
Script started, output file is typescript
(base) himaligajare@MacBook-Air src % make test
cd tests; make bmtest dbtest; whoami; make hftest bttest indextest jointest sorttest sortmerge
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java BMTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.BMTest

Running Buffer Management tests....
Replacer: Clock


  Test 1 does a simple test of normal buffer manager operations:
  - Allocate a bunch of new pages
  - Write something on each one
  - Read that something back from each one
   (because we're buffering, this is where most of the writes happen)
  - Free the pages again
  Test 1 completed successfully.

  Test 2 exercises some illegal buffer manager operations:
  - Try to pin more pages than there are frames
*** Pinning too many pages
  --> Failed as expected

  - Try to free a doubly-pinned page
*** Freeing a pinned page
  --> Failed as expected

  - Try to unpin a page not in the buffer pool
*** Unpinning a page not in the buffer pool
  --> Failed as expected

  Test 2 completed successfully.

  Test 3 exercises some of the internals of the buffer manager
  - Allocate and dirty some new pages, one at a time, and leave some pinned
  - Read the pages
  Test 3 completed successfully.

...Buffer Management tests completely successfully.

/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java DBTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.DBTest

Running Disk Space Management tests....

Replacer: Clock


  Test 1 creates a new database and does some tests of normal operations:
  - Add some file entries
  - Allocate a run of pages
  - Write something on some of them
  - Deallocate the rest of them
  Test 1 completed successfully.

  Test 2 opens the database created in test 1 and does some further tests:
  - Delete some of the file entries
  - Look up file entries that should still be there
  - Read stuff back from pages we wrote in test 1
  Test 2 completed successfully.

  Test 3 tests for some error conditions:
  - Look up a deleted file entry
**** Looking up a deleted file entry
  --> Failed as expected

  - Try to delete a deleted entry again
**** Delete a deleted file entry again
```

```
  --> Failed as expected

  - Try to delete a nonexistent file entry
**** Deleting a nonexistent file entry
  --> Failed as expected

  - Look up a nonexistent file entry
**** Looking up a nonexistent file entry
  --> Failed as expected

  - Try to add a file entry that's already there
**** Adding a duplicate file entry
  --> Failed as expected

  - Try to add a file entry whose name is too long
**** Adding a file entry with too long a name
  --> Failed as expected

  - Try to allocate a run of pages that's too long
**** Allocating a run that's too long
  --> Failed as expected

  - Try to allocate a negative run of pages
**** Allocating a negative run
  --> Failed as expected

  - Try to deallocate a negative run of pages
**** Deallocating a negative run
  --> Failed as expected

  Test 3 completed successfully.

  Test 4 tests some boundary conditions.
    (These tests are very implementation-specific.)
  - Make sure no pages are pinned
  - Allocate all pages remaining after DB overhead is accounted for
  - Attempt to allocate one more page
**** Allocating one additional page
  --> Failed as expected

  - Free some of the allocated pages
  - Allocate some of the just-freed pages
  - Free two continued run of the allocated pages
  - Allocate back number of pages equal to the just freed pages

  - Add enough file entries that the directory must surpass a page
  - Make sure that the directory has taken up an extra page: try to
    allocate more pages than should be available
**** Allocating more pages than are now available
   --> Failed as expected

  - At this point, all pages should be claimed.  Try to allocateone more.
**** Allocating one more page than there is
   --> Failed as expected
  - Free the last two pages: this tests a boundary condition in the space map.
  Test 4 completed successfully.

...Disk Space Management tests completely successfully.

himaligajare
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java HFTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.HFTest

Running Heap File tests....

Replacer: Clock


  Test 1: Insert and scan fixed-size records

  - Create a heap file

  - Add 100 records to the file

  - Scan the records just inserted

  Test 1 completed successfully.


  Test 2: Delete fixed-size records

  - Open the same heap file as test 1

  - Delete half the records

  - Scan the remaining records

  Test 2 completed successfully.


  Test 3: Update fixed-size records

  - Open the same heap file as tests 1 and 2

  - Change the records

  - Check that the updates are really there

  Test 3 completed successfully.


  Test 4: Test some error conditions

  - Try to change the size of a record
```

```
**** Shortening a record
  --> Failed as expected

**** Lengthening a record
  --> Failed as expected

  - Try to insert a record that's too long

**** Inserting a too-long record
  --> Failed as expected

  Test 4 completed successfully.


...Heap File tests completely successfully.

/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java BTTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.BTTest
Replacer: Clock


Running  tests....

 ***************** The file name is: AAA0  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
```

```
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :5
Please input the integer key to insert:
22
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :5
Please input the integer key to insert:
12
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)
```

```
[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :5
Please input the integer key to insert:
33
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
```

```
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :5
Please input the integer key to insert:
66
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :5
Please input the integer key to insert:
5
```

```
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :2



---------------The B+ Tree Structure---------------
1     3
--------------- End ---------------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print
```

```
          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 3
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (5,  [ 5 5 ] )
1 (key, [pageNo, slotNo]):   (12,  [ 12 12 ] )
2 (key, [pageNo, slotNo]):   (22,  [ 22 22 ] )
3 (key, [pageNo, slotNo]):   (33,  [ 33 33 ] )
4 (key, [pageNo, slotNo]):   (66,  [ 66 66 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
```

```
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :6
Please input the integer key to delete:
22
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
```

```
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 3
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (5,   [ 5 5 ] )
1 (key, [pageNo, slotNo]):   (12,   [ 12 12 ] )
2 (key, [pageNo, slotNo]):   (33,   [ 33 33 ] )
3 (key, [pageNo, slotNo]):   (66,   [ 66 66 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records
```

```
[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :4
Please input the page number:
5
Sorry!!! This page is neither Index nor Leaf page.
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :4
```

```
Please input the page number:
3

**************To Print an Leaf Page ********
Current Page ID: 3
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (5,   [ 5 5 ] )
1 (key, [pageNo, slotNo]):   (12,   [ 12 12 ] )
2 (key, [pageNo, slotNo]):   (33,   [ 33 33 ] )
3 (key, [pageNo, slotNo]):   (66,   [ 66 66 ] )
************** END ********

-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :11
Please input the LOWER integer key(>=0):
3
Please input the HIGHER integer key(>=0)
13
-------------------------- MENU ------------------
```

```
[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 3
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (33,  [ 33 33 ] )
1 (key, [pageNo, slotNo]):   (66,  [ 66 66 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------
```

```
[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :0
 ***************** The file name is: AAA1  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
```

```
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :7
Please input the number of keys to insert:
77
 ***************** The file name is: AAA2  **********
-------------------------- MENU -------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):
```

```
[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 6
Left Link      : -1
Right Link     : 7
0 (key, [pageNo, slotNo]):   (0,   [ 0 0 ] )
1 (key, [pageNo, slotNo]):   (1,   [ 1 1 ] )
2 (key, [pageNo, slotNo]):   (2,   [ 2 2 ] )
3 (key, [pageNo, slotNo]):   (3,   [ 3 3 ] )
4 (key, [pageNo, slotNo]):   (4,   [ 4 4 ] )
5 (key, [pageNo, slotNo]):   (5,   [ 5 5 ] )
6 (key, [pageNo, slotNo]):   (6,   [ 6 6 ] )
7 (key, [pageNo, slotNo]):   (7,   [ 7 7 ] )
8 (key, [pageNo, slotNo]):   (8,   [ 8 8 ] )
9 (key, [pageNo, slotNo]):   (9,   [ 9 9 ] )
10 (key, [pageNo, slotNo]):   (10,  [ 10 10 ] )
11 (key, [pageNo, slotNo]):   (11,  [ 11 11 ] )
12 (key, [pageNo, slotNo]):   (12,  [ 12 12 ] )
13 (key, [pageNo, slotNo]):   (13,  [ 13 13 ] )
14 (key, [pageNo, slotNo]):   (14,  [ 14 14 ] )
15 (key, [pageNo, slotNo]):   (15,  [ 15 15 ] )
16 (key, [pageNo, slotNo]):   (16,  [ 16 16 ] )
17 (key, [pageNo, slotNo]):   (17,  [ 17 17 ] )
18 (key, [pageNo, slotNo]):   (18,  [ 18 18 ] )
19 (key, [pageNo, slotNo]):   (19,  [ 19 19 ] )
20 (key, [pageNo, slotNo]):   (20,  [ 20 20 ] )
21 (key, [pageNo, slotNo]):   (21,  [ 21 21 ] )
22 (key, [pageNo, slotNo]):   (22,  [ 22 22 ] )
23 (key, [pageNo, slotNo]):   (23,  [ 23 23 ] )
24 (key, [pageNo, slotNo]):   (24,  [ 24 24 ] )
25 (key, [pageNo, slotNo]):   (25,  [ 25 25 ] )
26 (key, [pageNo, slotNo]):   (26,  [ 26 26 ] )
27 (key, [pageNo, slotNo]):   (27,  [ 27 27 ] )
28 (key, [pageNo, slotNo]):   (28,  [ 28 28 ] )
29 (key, [pageNo, slotNo]):   (29,  [ 29 29 ] )
30 (key, [pageNo, slotNo]):   (30,  [ 30 30 ] )
************** END ********


**************To Print an Leaf Page ********
Current Page ID: 7
Left Link      : 6
Right Link     : -1
0 (key, [pageNo, slotNo]):   (31,  [ 31 31 ] )
1 (key, [pageNo, slotNo]):   (32,  [ 32 32 ] )
2 (key, [pageNo, slotNo]):   (33,  [ 33 33 ] )
3 (key, [pageNo, slotNo]):   (34,  [ 34 34 ] )
```

```
4 (key, [pageNo, slotNo]):   (35,  [ 35 35 ] )
5 (key, [pageNo, slotNo]):   (36,  [ 36 36 ] )
6 (key, [pageNo, slotNo]):   (37,  [ 37 37 ] )
7 (key, [pageNo, slotNo]):   (38,  [ 38 38 ] )
8 (key, [pageNo, slotNo]):   (39,  [ 39 39 ] )
9 (key, [pageNo, slotNo]):   (40,  [ 40 40 ] )
10 (key, [pageNo, slotNo]):   (41,  [ 41 41 ] )
11 (key, [pageNo, slotNo]):   (42,  [ 42 42 ] )
12 (key, [pageNo, slotNo]):   (43,  [ 43 43 ] )
13 (key, [pageNo, slotNo]):   (44,  [ 44 44 ] )
14 (key, [pageNo, slotNo]):   (45,  [ 45 45 ] )
15 (key, [pageNo, slotNo]):   (46,  [ 46 46 ] )
16 (key, [pageNo, slotNo]):   (47,  [ 47 47 ] )
17 (key, [pageNo, slotNo]):   (48,  [ 48 48 ] )
18 (key, [pageNo, slotNo]):   (49,  [ 49 49 ] )
19 (key, [pageNo, slotNo]):   (50,  [ 50 50 ] )
20 (key, [pageNo, slotNo]):   (51,  [ 51 51 ] )
21 (key, [pageNo, slotNo]):   (52,  [ 52 52 ] )
22 (key, [pageNo, slotNo]):   (53,  [ 53 53 ] )
23 (key, [pageNo, slotNo]):   (54,  [ 54 54 ] )
24 (key, [pageNo, slotNo]):   (55,  [ 55 55 ] )
25 (key, [pageNo, slotNo]):   (56,  [ 56 56 ] )
26 (key, [pageNo, slotNo]):   (57,  [ 57 57 ] )
27 (key, [pageNo, slotNo]):   (58,  [ 58 58 ] )
28 (key, [pageNo, slotNo]):   (59,  [ 59 59 ] )
29 (key, [pageNo, slotNo]):   (60,  [ 60 60 ] )
30 (key, [pageNo, slotNo]):   (61,  [ 61 61 ] )
31 (key, [pageNo, slotNo]):   (62,  [ 62 62 ] )
32 (key, [pageNo, slotNo]):   (63,  [ 63 63 ] )
33 (key, [pageNo, slotNo]):   (64,  [ 64 64 ] )
34 (key, [pageNo, slotNo]):   (65,  [ 65 65 ] )
35 (key, [pageNo, slotNo]):   (66,  [ 66 66 ] )
36 (key, [pageNo, slotNo]):   (67,  [ 67 67 ] )
37 (key, [pageNo, slotNo]):   (68,  [ 68 68 ] )
38 (key, [pageNo, slotNo]):   (69,  [ 69 69 ] )
39 (key, [pageNo, slotNo]):   (70,  [ 70 70 ] )
40 (key, [pageNo, slotNo]):   (71,  [ 71 71 ] )
41 (key, [pageNo, slotNo]):   (72,  [ 72 72 ] )
42 (key, [pageNo, slotNo]):   (73,  [ 73 73 ] )
43 (key, [pageNo, slotNo]):   (74,  [ 74 74 ] )
44 (key, [pageNo, slotNo]):   (75,  [ 75 75 ] )
45 (key, [pageNo, slotNo]):   (76,  [ 76 76 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)
```

```
[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :8
Please input the number of keys to insert:
5
 **************** The file name is: AAA3  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records
```

```
[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 10
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (1,  [ 1 1 ] )
1 (key, [pageNo, slotNo]):   (2,  [ 2 2 ] )
2 (key, [pageNo, slotNo]):   (3,  [ 3 3 ] )
3 (key, [pageNo, slotNo]):   (4,  [ 4 4 ] )
4 (key, [pageNo, slotNo]):   (5,  [ 5 5 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
```

```
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :9
Please input the number of keys to insert:
10
 ***************** The file name is: AAA4  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
```

```
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 12
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (0,   [ 0 0 ] )
1 (key, [pageNo, slotNo]):   (1,   [ 1 1 ] )
2 (key, [pageNo, slotNo]):   (2,   [ 2 2 ] )
3 (key, [pageNo, slotNo]):   (3,   [ 3 3 ] )
4 (key, [pageNo, slotNo]):   (4,   [ 4 4 ] )
5 (key, [pageNo, slotNo]):   (5,   [ 5 5 ] )
6 (key, [pageNo, slotNo]):   (6,   [ 6 6 ] )
7 (key, [pageNo, slotNo]):   (7,   [ 7 7 ] )
8 (key, [pageNo, slotNo]):   (8,   [ 8 8 ] )
9 (key, [pageNo, slotNo]):   (9,   [ 9 9 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record
```

```
           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :10
Please input the number of keys to insert:
3
Please input the number of keys to delete:
2
 ***************** The file name is: AAA5  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3
```

```
---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 14
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (0,  [ 0 0 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :2
```

```
---------------The B+ Tree Structure---------------
1     14
--------------- End ---------------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :17
2

-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print
```

```
          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :0
 ***************** The file name is: AAA6  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---
```

```
[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :1
 ***************** The file name is: AAA7  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :12
Please input the LOWER integer key (null if -3):
3
Please input the HIGHER integer key (null if -2):
6
-------------------------- MENU ------------------


[0]   Naive delete (new file)
```

```
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :13
AT THE END OF SCAN!
------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records
```

```
[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3
The Tree is Empty!!!
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :2
The Tree is Empty!!!
-------------------------- MENU ------------------
```

```
[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :7
Please input the number of keys to insert:
10
 ***************** The file name is: AAA8  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
```

```
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :2



---------------The B+ Tree Structure---------------
1     18
--------------- End ---------------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---
```

```
[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 18
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (0,  [ 0 0 ] )
1 (key, [pageNo, slotNo]):   (1,  [ 1 1 ] )
2 (key, [pageNo, slotNo]):   (2,  [ 2 2 ] )
3 (key, [pageNo, slotNo]):   (3,  [ 3 3 ] )
4 (key, [pageNo, slotNo]):   (4,  [ 4 4 ] )
5 (key, [pageNo, slotNo]):   (5,  [ 5 5 ] )
6 (key, [pageNo, slotNo]):   (6,  [ 6 6 ] )
7 (key, [pageNo, slotNo]):   (7,  [ 7 7 ] )
8 (key, [pageNo, slotNo]):   (8,  [ 8 8 ] )
9 (key, [pageNo, slotNo]):   (9,  [ 9 9 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
```

```
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :15
Please input the number of keys to insert:
10
Please input the number of keys to delete:
2
 ***************** The file name is: AAA9  **********
-------------------------- MENU ------------------



[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
```

```
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :2



---------------The B+ Tree Structure---------------
1     20
--------------- End ---------------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------
```

```
**************To Print an Leaf Page ********
Current Page ID: 20
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (**0,  [ 0 0 ]
1 (key, [pageNo, slotNo]):   (**2,  [ 2 2 ]
2 (key, [pageNo, slotNo]):   (**3,  [ 3 3 ]
3 (key, [pageNo, slotNo]):   (**4,  [ 4 4 ]
4 (key, [pageNo, slotNo]):   (**6,  [ 6 6 ]
5 (key, [pageNo, slotNo]):   (**7,  [ 7 7 ]
6 (key, [pageNo, slotNo]):   (**8,  [ 8 8 ]
7 (key, [pageNo, slotNo]):   (**9,  [ 9 9 ]
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
```

```
Hi, make your choice :4
Please input the page number:
3

**************To Print an Leaf Page ********
Current Page ID: 3
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (,  [ 33 33 ]
1 (key, [pageNo, slotNo]):   (,  [ 66 66 ]
************** END ********

-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :4
Please input the page number:
5
Sorry!!! This page is neither Index nor Leaf page.
-------------------------- MENU ------------------
```

```
[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :4
Please input the page number:
1
Sorry!!! This page is neither Index nor Leaf page.
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

          ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
```

```
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :4
Please input the page number:
2
Sorry!!! This page is neither Index nor Leaf page.
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):
```

```
[19]  Quit!
Hi, make your choice :4
Please input the page number:
7

**************To Print an Leaf Page ********
Current Page ID: 7
Left Link      : 6
Right Link     : -1
0 (key, [pageNo, slotNo]):   (,  [ 31 31 ]
1 (key, [pageNo, slotNo]):   (,  [ 32 32 ]
2 (key, [pageNo, slotNo]):   (,  [ 33 33 ]
3 (key, [pageNo, slotNo]):   (,  [ 34 34 ]
4 (key, [pageNo, slotNo]):   (,  [ 35 35 ]
5 (key, [pageNo, slotNo]):   (,  [ 36 36 ]
6 (key, [pageNo, slotNo]):   (,  [ 37 37 ]
7 (key, [pageNo, slotNo]):   (,  [ 38 38 ]
8 (key, [pageNo, slotNo]):   (,  [ 39 39 ]
9 (key, [pageNo, slotNo]):   (,  [ 40 40 ]
10 (key, [pageNo, slotNo]):   (,  [ 41 41 ]
11 (key, [pageNo, slotNo]):   (,  [ 42 42 ]
12 (key, [pageNo, slotNo]):   (,  [ 43 43 ]
13 (key, [pageNo, slotNo]):   (,  [ 44 44 ]
14 (key, [pageNo, slotNo]):   (,  [ 45 45 ]
15 (key, [pageNo, slotNo]):   (,  [ 46 46 ]
16 (key, [pageNo, slotNo]):   (,  [ 47 47 ]
17 (key, [pageNo, slotNo]):   (,  [ 48 48 ]
18 (key, [pageNo, slotNo]):   (,  [ 49 49 ]
19 (key, [pageNo, slotNo]):   (,  [ 50 50 ]
20 (key, [pageNo, slotNo]):   (,  [ 51 51 ]
21 (key, [pageNo, slotNo]):   (,  [ 52 52 ]
22 (key, [pageNo, slotNo]):   (,  [ 53 53 ]
23 (key, [pageNo, slotNo]):   (,  [ 54 54 ]
24 (key, [pageNo, slotNo]):   (,  [ 55 55 ]
25 (key, [pageNo, slotNo]):   (,  [ 56 56 ]
26 (key, [pageNo, slotNo]):   (,  [ 57 57 ]
27 (key, [pageNo, slotNo]):   (,  [ 58 58 ]
28 (key, [pageNo, slotNo]):   (,  [ 59 59 ]
29 (key, [pageNo, slotNo]):   (,  [ 60 60 ]
30 (key, [pageNo, slotNo]):   (,  [ 61 61 ]
31 (key, [pageNo, slotNo]):   (,  [ 62 62 ]
32 (key, [pageNo, slotNo]):   (,  [ 63 63 ]
33 (key, [pageNo, slotNo]):   (,  [ 64 64 ]
34 (key, [pageNo, slotNo]):   (,  [ 65 65 ]
35 (key, [pageNo, slotNo]):   (,  [ 66 66 ]
36 (key, [pageNo, slotNo]):   (,  [ 67 67 ]
37 (key, [pageNo, slotNo]):   (,  [ 68 68 ]
38 (key, [pageNo, slotNo]):   (,  [ 69 69 ]
39 (key, [pageNo, slotNo]):   (,  [ 70 70 ]
40 (key, [pageNo, slotNo]):   (,  [ 71 71 ]
41 (key, [pageNo, slotNo]):   (,  [ 72 72 ]
42 (key, [pageNo, slotNo]):   (,  [ 73 73 ]
43 (key, [pageNo, slotNo]):   (,  [ 74 74 ]
44 (key, [pageNo, slotNo]):   (,  [ 75 75 ]
45 (key, [pageNo, slotNo]):   (,  [ 76 76 ]
************** END ********

-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :12
Please input the LOWER integer key (null if -3):
2
Please input the HIGHER integer key (null if -2):
20
btree.KeyNotMatchException: key types do not match
       at btree.BT.keyCompare(BT.java:62)
       at btree.BTreeFile.findRunStart(BTreeFile.java:1118)
       at btree.BTreeFile.new_scan(BTreeFile.java:1793)
       at tests.BTDriver.runAllTests(BTTest.java:267)
       at tests.BTDriver.runTests(BTTest.java:80)
       at tests.BTTest.main(BTTest.java:648)
       !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
       !!         Something is wrong                    !!
       !!     Is your DB full? then exit. rerun it!     !!
       !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :7
Please input the number of keys to insert:
22
 ***************** The file name is: AAA10  **********
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
```

```
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

          ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 22
Left Link      : -1
Right Link     : -1
0 (key, [pageNo, slotNo]):   (0,  [ 0 0 ] )
1 (key, [pageNo, slotNo]):   (1,  [ 1 1 ] )
2 (key, [pageNo, slotNo]):   (2,  [ 2 2 ] )
3 (key, [pageNo, slotNo]):   (3,  [ 3 3 ] )
4 (key, [pageNo, slotNo]):   (4,  [ 4 4 ] )
5 (key, [pageNo, slotNo]):   (5,  [ 5 5 ] )
6 (key, [pageNo, slotNo]):   (6,  [ 6 6 ] )
7 (key, [pageNo, slotNo]):   (7,  [ 7 7 ] )
8 (key, [pageNo, slotNo]):   (8,  [ 8 8 ] )
9 (key, [pageNo, slotNo]):   (9,  [ 9 9 ] )
10 (key, [pageNo, slotNo]):   (10,  [ 10 10 ] )
11 (key, [pageNo, slotNo]):   (11,  [ 11 11 ] )
12 (key, [pageNo, slotNo]):   (12,  [ 12 12 ] )
13 (key, [pageNo, slotNo]):   (13,  [ 13 13 ] )
14 (key, [pageNo, slotNo]):   (14,  [ 14 14 ] )
15 (key, [pageNo, slotNo]):   (15,  [ 15 15 ] )
16 (key, [pageNo, slotNo]):   (16,  [ 16 16 ] )
17 (key, [pageNo, slotNo]):   (17,  [ 17 17 ] )
18 (key, [pageNo, slotNo]):   (18,  [ 18 18 ] )
19 (key, [pageNo, slotNo]):   (19,  [ 19 19 ] )
20 (key, [pageNo, slotNo]):   (20,  [ 20 20 ] )
21 (key, [pageNo, slotNo]):   (21,  [ 21 21 ] )
************** END ********
```

```
------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :12
Please input the LOWER integer key (null if -3):
2
Please input the HIGHER integer key (null if -2):
17
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print
```

```
           ---Integer Key (for choices [6]-[14]) ---

[5]    Insert a Record
[6]    Delete a Record
[7]    Test1 (new file): insert n records in order
[8]    Test2 (new file): insert n records in reverse order
[9]    Test3 (new file): insert n records in random order
[10]   Test4 (new file): insert n records in random order
       and delete m records randomly
[11]   Delete some records

[12]   Initialize a Scan
[13]   Scan the next Record
[14]   Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]   Test5 (new file): insert n records in random order
         and delete m records randomly.

[16]   Close the file
[17]   Open which file (input an integer for the file name):
[18]   Destroy which file (input an integer for the file name):

[19]   Quit!
Hi, make your choice :13
SCAN RESULT: 2 [ 2 2 ]
-------------------------- MENU ------------------


[0]    Naive delete (new file)
[1]    Full delete(Default) (new file)

[2]    Print the B+ Tree Structure
[3]    Print All Leaf Pages
[4]    Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]    Insert a Record
[6]    Delete a Record
[7]    Test1 (new file): insert n records in order
[8]    Test2 (new file): insert n records in reverse order
[9]    Test3 (new file): insert n records in random order
[10]   Test4 (new file): insert n records in random order
       and delete m records randomly
[11]   Delete some records

[12]   Initialize a Scan
[13]   Scan the next Record
[14]   Delete the just-scanned record

           ---String Key (for choice [15]) ---
```

```
[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :14
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :3



---------------The B+ Tree Leaf Pages---------------

**************To Print an Leaf Page ********
Current Page ID: 22
Left Link      : -1
```

```
Right Link     : -1
0 (key, [pageNo, slotNo]):   (0,  [ 0 0 ] )
1 (key, [pageNo, slotNo]):   (1,  [ 1 1 ] )
2 (key, [pageNo, slotNo]):   (3,  [ 3 3 ] )
3 (key, [pageNo, slotNo]):   (4,  [ 4 4 ] )
4 (key, [pageNo, slotNo]):   (5,  [ 5 5 ] )
5 (key, [pageNo, slotNo]):   (6,  [ 6 6 ] )
6 (key, [pageNo, slotNo]):   (7,  [ 7 7 ] )
7 (key, [pageNo, slotNo]):   (8,  [ 8 8 ] )
8 (key, [pageNo, slotNo]):   (9,  [ 9 9 ] )
9 (key, [pageNo, slotNo]):   (10,  [ 10 10 ] )
10 (key, [pageNo, slotNo]):   (11,  [ 11 11 ] )
11 (key, [pageNo, slotNo]):   (12,  [ 12 12 ] )
12 (key, [pageNo, slotNo]):   (13,  [ 13 13 ] )
13 (key, [pageNo, slotNo]):   (14,  [ 14 14 ] )
14 (key, [pageNo, slotNo]):   (15,  [ 15 15 ] )
15 (key, [pageNo, slotNo]):   (16,  [ 16 16 ] )
16 (key, [pageNo, slotNo]):   (17,  [ 17 17 ] )
17 (key, [pageNo, slotNo]):   (18,  [ 18 18 ] )
18 (key, [pageNo, slotNo]):   (19,  [ 19 19 ] )
19 (key, [pageNo, slotNo]):   (20,  [ 20 20 ] )
20 (key, [pageNo, slotNo]):   (21,  [ 21 21 ] )
************** END ********



------------- All Leaf Pages Have Been Printed --------


-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record
```

```
           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :12
Please input the LOWER integer key (null if -3):
19
Please input the HIGHER integer key (null if -2):
55
-------------------------- MENU ------------------


[0]   Naive delete (new file)
[1]   Full delete(Default) (new file)

[2]   Print the B+ Tree Structure
[3]   Print All Leaf Pages
[4]   Choose a Page to Print

           ---Integer Key (for choices [6]-[14]) ---

[5]   Insert a Record
[6]   Delete a Record
[7]   Test1 (new file): insert n records in order
[8]   Test2 (new file): insert n records in reverse order
[9]   Test3 (new file): insert n records in random order
[10]  Test4 (new file): insert n records in random order
      and delete m records randomly
[11]  Delete some records

[12]  Initialize a Scan
[13]  Scan the next Record
[14]  Delete the just-scanned record

           ---String Key (for choice [15]) ---

[15]  Test5 (new file): insert n records in random order
        and delete m records randomly.

[16]  Close the file
[17]  Open which file (input an integer for the file name):
[18]  Destroy which file (input an integer for the file name):

[19]  Quit!
Hi, make your choice :19

... Finished .
```

```
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java IndexTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.IndexTest

Running Index tests....

Replacer: Clock

------------------------ TEST 1 --------------------------
BTreeIndex created successfully.

BTreeIndex file created successfully.

Test1 -- Index Scan OK
------------------- TEST 1 completed ---------------------

------------------------ TEST 2 --------------------------
BTreeIndex opened successfully.

Test2 -- Index Scan OK
------------------- TEST 2 completed ---------------------

------------------------ TEST 3 --------------------------
BTreeIndex created successfully.

BTreeIndex file created successfully.

Test3 -- Index scan on int key OK

------------------- TEST 3 completed ---------------------


...Index tests
completely successfully
.


Index tests completed successfully
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java JoinTest.java
Note: JoinTest.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.JoinTest
Replacer: Clock



Any resemblance of persons in this database to people living or dead
is purely coincidental. The contents of this database do not reflect
the views of the University, the Computer  Sciences Department or the
developers...
```

```
**********************Query1 strating *********************
Query: Find the names of sailors who have reserved boat number 1.
       and print out the date of reservation.

  SELECT S.sname, R.date
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid AND R.bid = 1


(Tests FileScan, Projection, and Sort-Merge Join)
[Mike Carey, 05/10/95]
[David Dewitt, 05/11/95]
[Jeff Naughton, 05/12/95]

Query1 completed successfully!
*******************Query1 finished!!!*****************



**********************Query2 strating *********************
Query: Find the names of sailors who have reserved a red boat
       and return them in alphabetical order.

  SELECT   S.sname
  FROM     Sailors S, Boats B, Reserves R
  WHERE    S.sid = R.sid AND R.bid = B.bid AND B.color = 'red'
  ORDER BY S.sname
Plan used:
 Sort (Pi(sname) (Sigma(B.color='red')  |><|  Pi(sname, bid) (S  |><|  R)))

(Tests File scan, Index scan ,Projection,  index selection,
 sort and simple nested-loop join.)

After Building btree index on sailors.sid.

[David Dewitt]
[Mike Carey]
[Raghu Ramakrishnan]
[Yannis Ioannidis]

Query2 completed successfully!
*******************Query2 finished!!!*****************



**********************Query3 strating *********************
Query: Find the names of sailors who have reserved a boat.

  SELECT S.sname
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid

(Tests FileScan, Projection, and SortMerge Join.)
```

```
[Mike Carey]
[Mike Carey]
[Mike Carey]
[David Dewitt]
[David Dewitt]
[Jeff Naughton]
[Miron Livny]
[Yannis Ioannidis]
[Raghu Ramakrishnan]
[Raghu Ramakrishnan]

Query3 completed successfully!
*******************Query3 finished!!!*****************



**********************Query4 strating *********************
Query: Find the names of sailors who have reserved a boat
       and print each name once.

  SELECT DISTINCT S.sname
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid

(Tests FileScan, Projection, Sort-Merge Join and Duplication elimination.)

[David Dewitt]
[Jeff Naughton]
[Mike Carey]
[Miron Livny]
[Raghu Ramakrishnan]
[Yannis Ioannidis]

Query4 completed successfully!
*******************Query4 finished!!!*****************



**********************Query5 strating *********************
Query: Find the names of old sailors or sailors with a rating less
       than 7, who have reserved a boat, (perhaps to increase the
       amount they have to pay to make a reservation).

  SELECT S.sname, S.rating, S.age
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid and (S.age > 40 || S.rating < 7)

(Tests FileScan, Multiple Selection, Projection, and Sort-Merge Join.)

[Mike Carey, 9, 40.3]
[Mike Carey, 9, 40.3]
[Mike Carey, 9, 40.3]
[David Dewitt, 10, 47.2]
```

```
[David Dewitt, 10, 47.2]
[Jeff Naughton, 5, 35.0]
[Yannis Ioannidis, 8, 40.2]

Query5 completed successfully!
*******************Query5 finished!!!*****************



**********************Query6 strating *********************
Query: Find the names of sailors with a rating greater than 7
  who have reserved a red boat, and print them out in sorted order.

  SELECT   S.sname
  FROM     Sailors S, Boats B, Reserves R
  WHERE    S.sid = R.sid AND S.rating > 7 AND R.bid = B.bid
           AND B.color = 'red'
  ORDER BY S.name

Plan used:
 Sort(Pi(sname) (Sigma(B.color='red')  |><|  Pi(sname, bid) (Sigma(S.rating > 7)  |><|  R)))

(Tests FileScan, Multiple Selection, Projection,sort and nested-loop join.)

After nested loop join S.sid|><|R.sid.
After nested loop join R.bid|><|B.bid AND B.color=red.
After sorting the output tuples.
[David Dewitt]
[Mike Carey]
[Raghu Ramakrishnan]
[Yannis Ioannidis]

Query6 completed successfully!
*******************Query6 finished!!!*****************



Finished joins testing
join tests completed successfully
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
TestDriver.java SortTest.java
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.SortTest

Running Sort tests....

Replacer: Clock

------------------------ TEST 1 --------------------------
Test1 -- Sorting OK
------------------- TEST 1 completed ---------------------

------------------------ TEST 2 --------------------------
Test2 -- Sorting OK
```

```
------------------- TEST 2 completed ---------------------

------------------------ TEST 3 --------------------------
 -- Sorting in ascending order on the int field --
Test3 -- Sorting of int field OK

 -- Sorting in descending order on the float field --
Test3 -- Sorting of float field OK

------------------- TEST 3 completed ---------------------

------------------------ TEST 4 --------------------------
Test4 -- Sorting OK
------------------- TEST 4 completed ---------------------


...Sort tests
completely successfully
.
Sorting tests completed successfully
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/javac -classpath .:..
SM_JoinTest.java TestDriver.java
Note: SM_JoinTest.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
/Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java  -classpath .:..
tests.SM_JoinTest
Replacer: Clock

Any resemblance of persons in this database to people living or dead
is purely coincidental. The contents of this database do not reflect
the views of the University, the Computer  Sciences Department or the
developers...

*********************Query1 strating ********************
Query: Find the names of sailors who have reserved boat number 1.
       and print out the date of reservation.

  SELECT S.sname, R.date
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid AND R.bid = 1


(Tests FileScan, Projection, and Sort-Merge Join)
[Mike Carey, 05/10/95]
[David Dewitt, 05/11/95]
[Jeff Naughton, 05/12/95]

Query1 completed successfully!
******************Query1 finished!!!*****************

*********************Query3 strating ********************
Query: Find the names of sailors who have reserved a boat.

  SELECT S.sname
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid

(Tests FileScan, Projection, and SortMerge Join.)

[Mike Carey]
[Mike Carey]
[Mike Carey]
[David Dewitt]
[David Dewitt]
[Jeff Naughton]
[Miron Livny]
[Yannis Ioannidis]
[Raghu Ramakrishnan]
[Raghu Ramakrishnan]

Query3 completed successfully!
*******************Query3 finished!!!*****************



**********************Query4 strating *********************
Query: Find the names of sailors who have reserved a boat
       and print each name once.

  SELECT DISTINCT S.sname
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid

(Tests FileScan, Projection, Sort-Merge Join and Duplication elimination.)

[David Dewitt]
[Jeff Naughton]
[Mike Carey]
[Miron Livny]
[Raghu Ramakrishnan]
[Yannis Ioannidis]

Query4 completed successfully!
*******************Query4 finished!!!*****************



**********************Query5 strating *********************
Query: Find the names of old sailors or sailors with a rating less
       than 7, who have reserved a boat, (perhaps to increase the
       amount they have to pay to make a reservation).

  SELECT S.sname, S.rating, S.age
  FROM   Sailors S, Reserves R
  WHERE  S.sid = R.sid and (S.age > 40 || S.rating < 7)

(Tests FileScan, Multiple Selection, Projection, and Sort-Merge Join.)
```

```
[Mike Carey, 9, 40.3]
[Mike Carey, 9, 40.3]
[Mike Carey, 9, 40.3]
[David Dewitt, 10, 47.2]
[David Dewitt, 10, 47.2]
[Jeff Naughton, 5, 35.0]
[Yannis Ioannidis, 8, 40.2]

Query5 completed successfully!
*******************Query5 finished!!!*****************

Finished joins testing
join tests completed successfully
(base) himaligajare@MacBook-Air src %
```