Software Development Report
for
CSC 121: Programming II: Fall 2022
Hashing
by
Manel Casado

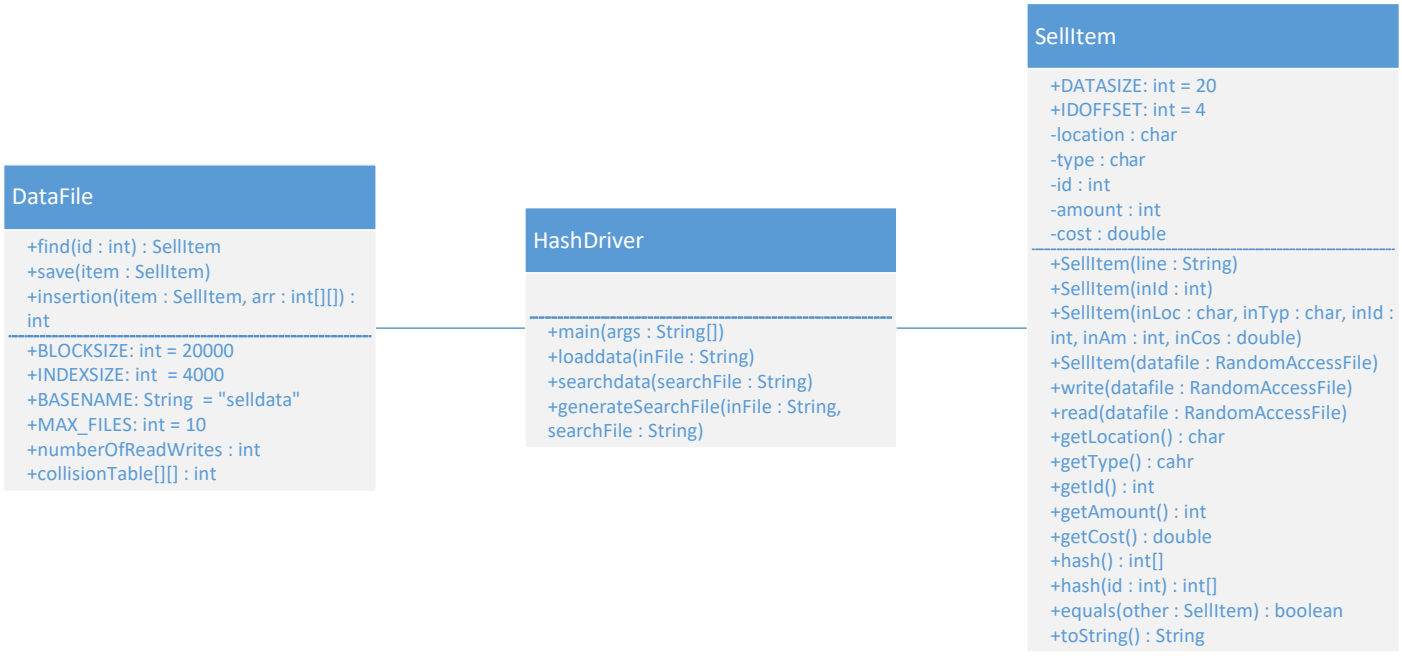
Problem Summary

Continue to study ways of how to process large amounts of data when the amount of data is too large for just one file or when there are so many lookups making the binary search slow.

Implementation Requirements

- Hashing
- Collision handling
- Access random files
- Hashing routine to achieve O(1.3)

System Design



Testing Plan

CASE 1:

I created a sample file with specifically chosen data to be able to test collisions, these are not randomly generated.

I have created a table to represent the hashing. Id's 4001 and 2 are the cases that test the collision handling and both of them work correctly.

Id	File num	Index Position	Hash Collisions using hash key	New index position (after collision fix)
1	0	1	False	1
4001	0	1	True	2
2	0	2	True	3
200	0	200	False	200
20000	1	0	False	0
20001	1	1	False	1
20002	1	2	False	2

Output:

```
Generating idsToSearch.csv This is the test file with Ids to
search.
Generated idsToSearch.csv
Item id= 1, location= E, type= t, amount= 1000, cost900.0
stored.
Item id= 4001, location= I, type= t, amount= 999, cost888.0
stored.
Item id= 2, location= U, type= t, amount= 333, cost222.0
stored.
Item id= 200, location= U, type= t, amount= 100, cost227.0
stored.
Item id= 300, location= N, type= t, amount= 101, cost228.0
stored.
Item id= 20000, location= J, type= t, amount= 101, cost555.0
stored.
Item id= 20001, location= E, type= t, amount= 102, cost229.0
stored.
Item id= 20002, location= G, type= t, amount= 103, cost300.0
stored.
Number of read/writes done: 8
Found: id= 1, location= E, type= t, amount= 1000, cost900.0
Found: id= 4001, location= I, type= t, amount= 999, cost888.0
```

```
Found: id= 2, location= U, type= t, amount= 333, cost222.0
Found: id= 200, location= U, type= t, amount= 100, cost227.0
Found: id= 300, location= N, type= t, amount= 101, cost228.0
Found: id= 20000, location= J, type= t, amount= 101,
cost555.0
Found: id= 20001, location= E, type= t, amount= 102,
cost229.0
Found: id= 20002, location= G, type= t, amount= 103,
cost300.0
Number of read/writes done: 16
```

CASE 2:

Another test I have done is comment "generateSearchFile(args[0], args[1]);" from HashDriver and adding an id to idsToSearch without having it on sampleList (for example 555) and run the program. The program outputs "Item with id = 555 not found" and doesn't update the number of reads and writes because only the valid ones are counted.

Output:

```
Item id= 1, location= E, type= t, amount= 1000, cost900.0
stored.
Item id= 4001, location= I, type= t, amount= 999, cost888.0
stored.
Item id= 2, location= U, type= t, amount= 333, cost222.0
stored.
Item id= 200, location= U, type= t, amount= 100, cost227.0
stored.
Item id= 300, location= N, type= t, amount= 101, cost228.0
stored.
Item id= 20000, location= J, type= t, amount= 101, cost555.0
stored.
Item id= 20001, location= E, type= t, amount= 102, cost229.0
stored.
tem id= 20002, location= G, type= t, amount= 103, cost300.0
stored.
Number of read/writes done: 8
Found: id= 1, location= E, type= t, amount= 1000, cost900.0
Found: id= 4001, location= I, type= t, amount= 999, cost888.0
Found: id= 2, location= U, type= t, amount= 333, cost222.0
Found: id= 200, location= U, type= t, amount= 100, cost227.0
```

```
Found: id= 300, location= N, type= t, amount= 101, cost228.0
Found: id= 20000, location= J, type= t, amount= 101,
cost555.0
Found: id= 20001, location= E, type= t, amount= 102,
cost229.0
Found: id= 20002, location= G, type= t, amount= 103,
cost300.0
Item with id= 555 not found
Number of read/writes done: 16
```

Time Spent

The program and development report took me about 16 hours

Outside resources used

- UML cheat sheet provided in Canvas
- Classes provided on GitHub Classroom
- Textbook for the hashing and collision handling
- Help from my dad with the testing and formating

Security report

- Failed to read the correct file.
- Can't read the file.

Ethical report

- I can't think of any ethical risk since this program doesn't use any sensitive data
- This program more developed could be used to store large amounts of data so it could be used by large companies.

Future Improvements

- Add more testing
- Try with files with random data

Lessons learned

- Hashing benefits
- How to do hashing
- Collision handling

Improvements of Work

- Positive security risks
- Better testing