

Hash Function Performance Report

By: Mohamed Saifeldin Yahia

Introduction:

This report aims to use experimental results to explore the performance of three hash functions when applied to the same 3 supplementary datasets in terms of the average number of collisions per dataset and then in overall.

Analysis:

All the raw data can be seen within the following pages.

Dataset 1: flight-ticket1k.csv

Hash function 2 and 3 yielded the least collisions at 162 each, while hash function 1 yielded 254 collisions.

Dataset 2: flight-ticket10k.csv

Hash function 1 yielded the least collisions at 940, hash function 2 gave 2210 collisions and lastly hash function 3 gave 2214 collisions.

Dataset 3: flight-ticket100k.csv

Hash function 1 yielded the least collisions at 974, hash function 2 gave 2396 collisions and lastly hash function 3 gave 2400 collisions.

Hash function 1 overall performance:

Average collisions for hash function 1 is $(254+940+974)/3 = 722.67$ collisions

Hash function 2 overall performance:

Average collisions for hash function 2 is $(162+2210+2396)/3 = 1589.33$ collisions

Hash function 3 overall performance:

Average collisions for hash function 3 is $(162+2214+2400)/3 = 1592$ collisions

Conclusion:

Despite hash function 1 performing the worst for the first dataset, it yielded the least average collisions by a significant margin across all datasets at 722.67 collisions, hence it is the most effective and will be used for the project.

Hash Function 1 applied to dataset with 1k items:

The screenshot shows a code editor with several tabs open. The active tab is 'flightfunctions.cpp'. The code defines a class 'FlightHASHTABLE' with methods for inserting, deleting, and searching flight tickets. It also includes a hash function and command-line interface logic for importing from CSV and printing collisions.

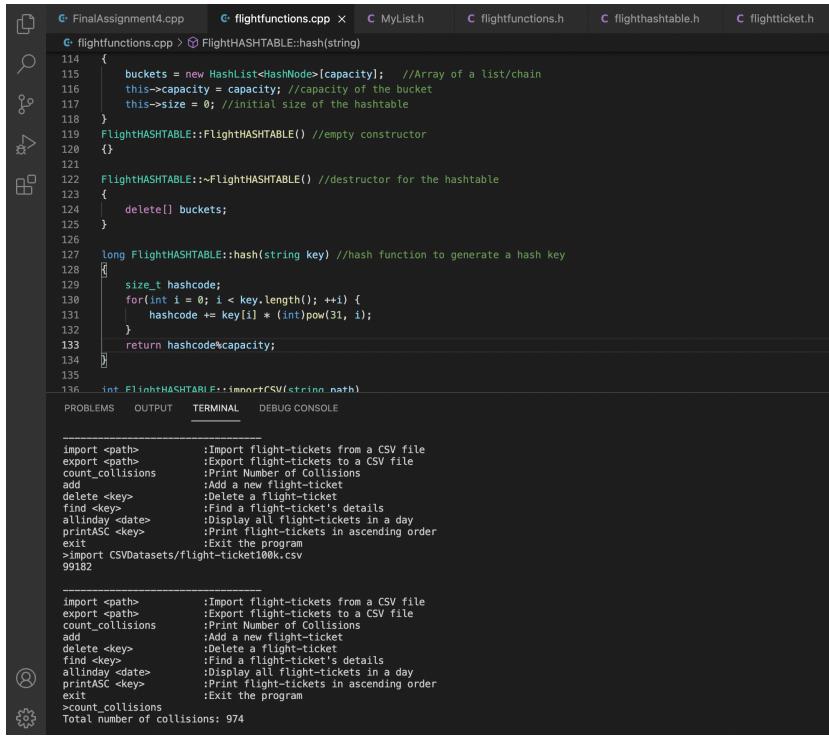
```
114     {
115         buckets = new HashList<classNode>[capacity]; //Array of a list/chain
116         this->capacity = capacity; //capacity of the bucket
117         this->size = 0; //initial size of the hashtable
118     }
119     FlightHASHTABLE::FlightHASHTABLE() //empty constructor
120 {
121
122     FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
123     {
124         delete[] buckets;
125     }
126
127     long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
128     {
129         size_t hashCode;
130         for(int i = 0; i < key.length(); ++i) {
131             hashCode += key[i] * (int)pow(31, i);
132         }
133         return hashCode%capacity;
134     }
135
136     int FlightHASHTABLE::importCSV(string path)
137
138     import <path> :Import flight-tickets from a CSV file
139     export <path> :Export flight-tickets to a CSV file
140     count_collisions :Print Number of Collisions
141     add :Add a new flight-ticket
142     delete <key> :Delete a flight-ticket
143     find <key> :Find a flight-ticket's details
144     allinday <date> :Display all flight-tickets in a day
145     printASC <key> :Print flight-tickets in ascending order
146     exit :Exit the program
147
148     >import CSVdatasets/flight-ticket1k.csv
149
150
151     import <path> :Import flight-tickets from a CSV file
152     export <path> :Export flight-tickets to a CSV file
153     count_collisions :Print Number of Collisions
154     add :Add a new flight-ticket
155     delete <key> :Delete a flight-ticket
156     find <key> :Find a flight-ticket's details
157     allinday <date> :Display all flight-tickets in a day
158     printASC <key> :Print flight-tickets in ascending order
159     exit :Exit the program
160
161     >count_collisions
162
163 Total number of collisions: 254
```

Hash function 1 applied to dataset with 10k items:

The screenshot shows a code editor with several tabs open. The active tab is 'flightfunctions.cpp'. The code defines a class 'FlightHASHTABLE' with methods for inserting, deleting, and searching flight tickets. It also includes a hash function and command-line interface logic for importing from CSV and printing collisions.

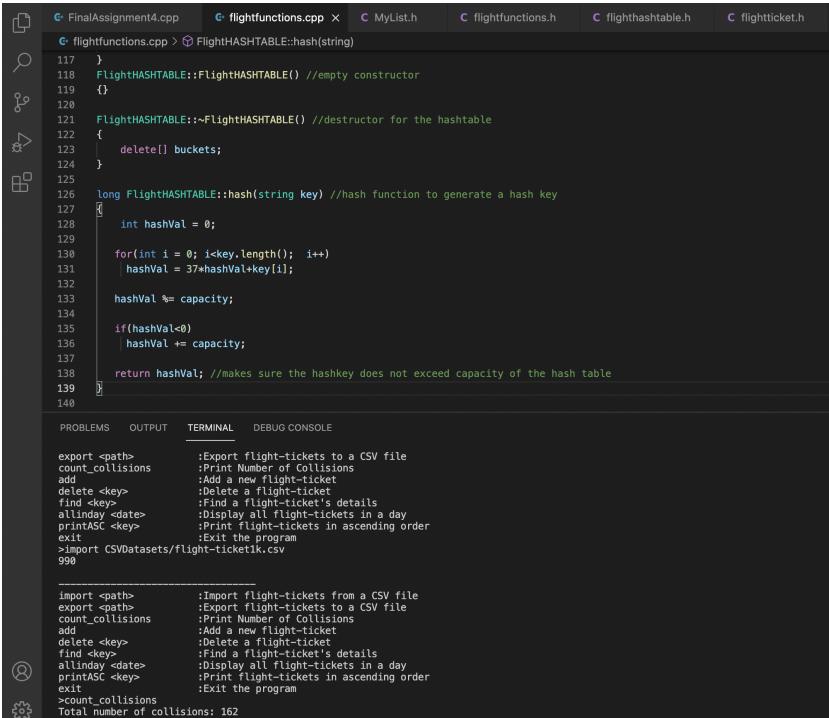
```
114     {
115         buckets = new HashList<classNode>[capacity]; //Array of a list/chain
116         this->capacity = capacity; //capacity of the bucket
117         this->size = 0; //initial size of the hashtable
118     }
119     FlightHASHTABLE::FlightHASHTABLE() //empty constructor
120 {
121
122     FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
123     {
124         delete[] buckets;
125     }
126
127     long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
128     {
129         size_t hashCode;
130         for(int i = 0; i < key.length(); ++i) {
131             hashCode += key[i] * (int)pow(31, i);
132         }
133         return hashCode%capacity;
134     }
135
136     int FlightHASHTABLE::importCSV(string path)
137
138     import <path> :Import flight-tickets from a CSV file
139     export <path> :Export flight-tickets to a CSV file
140     count_collisions :Print Number of Collisions
141     add :Add a new flight-ticket
142     delete <key> :Delete a flight-ticket
143     find <key> :Find a flight-ticket's details
144     allinday <date> :Display all flight-tickets in a day
145     printASC <key> :Print flight-tickets in ascending order
146     exit :Exit the program
147
148     >import CSVdatasets/flight-ticket10k.csv
149
150
151     import <path> :Import flight-tickets from a CSV file
152     export <path> :Export flight-tickets to a CSV file
153     count_collisions :Print Number of Collisions
154     add :Add a new flight-ticket
155     delete <key> :Delete a flight-ticket
156     find <key> :Find a flight-ticket's details
157     allinday <date> :Display all flight-tickets in a day
158     printASC <key> :Print flight-tickets in ascending order
159     exit :Exit the program
160
161     >count_collisions
162
163 Total number of collisions: 940
```

Hash function 1 applied to dataset with 100k items:



```
114     {
115         buckets = new HashList<HashNode>[capacity]; //Array of a list/chain
116         this->capacity = capacity; //capacity of the bucket
117         this->size = 0; //initial size of the hashtable
118     }
119     FlightHASHTABLE::FlightHASHTABLE() //empty constructor
120 {
121
122     FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
123     {
124         delete[] buckets;
125     }
126
127     long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
128     {
129         size_t hashCode;
130         for(int i = 0; i < key.length(); ++i) {
131             hashCode += key[i] * (int)pow(31, i);
132         }
133         return hashCode/capacity;
134     }
135
136     int FlightHASHTABLE::importCSV(string path)
137
138     import <path>           :Import flight-tickets from a CSV file
139     export <path>            :Export flight-tickets to a CSV file
140     count_collisions        :Print Number of Collisions
141     add                      :Add a new flight-ticket
142     delete <key>             :Delete a flight-ticket
143     find <key>               :Find a flight-ticket's details
144     allinaday <date>         :Display all flight-tickets in a day
145     printASC <key>           :Print flight-tickets in ascending order
146     exit                     :Exit the program
147     >import CSVDatasets/flight-ticket100K.csv
148
149
150     import <path>           :Import flight-tickets from a CSV file
151     export <path>            :Export flight-tickets to a CSV file
152     count_collisions        :Print Number of Collisions
153     add                      :Add a new flight-ticket
154     delete <key>             :Delete a flight-ticket
155     find <key>               :Find a flight-ticket's details
156     allinaday <date>         :Display all flight-tickets in a day
157     printASC <key>           :Print flight-tickets in ascending order
158     exit                     :Exit the program
159
160     >count_collisions
161
162 Total number of collisions: 974
```

Hash function 2 applied to dataset with 1k items:



```
117     }
118     FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {
120
121     FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
122     {
123         delete[] buckets;
124     }
125
126     long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
127     {
128         int hashVal = 0;
129
130         for(int i = 0; i < key.length(); i++)
131             hashVal = 37*hashVal+key[i];
132
133         hashVal %= capacity;
134
135         if(hashVal<0)
136             hashVal += capacity;
137
138         return hashVal; //makes sure the hashkey does not exceed capacity of the hash table
139     }
140
141
142     export <path>           :Export flight-tickets to a CSV file
143     count_collisions        :Print Number of Collisions
144     add                      :Add a new flight-ticket
145     delete <key>             :Delete a flight-ticket
146     find <key>               :Find a flight-ticket's details
147     allinaday <date>         :Display all flight-tickets in a day
148     printASC <key>           :Print flight-tickets in ascending order
149     exit                     :Exit the program
150     >export CSVDatasets/flight-ticket1K.csv
151
152
153     import <path>           :Import flight-tickets from a CSV file
154     export <path>            :Export flight-tickets to a CSV file
155     count_collisions        :Print Number of Collisions
156     add                      :Add a new flight-ticket
157     delete <key>             :Delete a flight-ticket
158     find <key>               :Find a flight-ticket's details
159     allinaday <date>         :Display all flight-tickets in a day
160     printASC <key>           :Print flight-tickets in ascending order
161     exit                     :Exit the program
162
163     >count_collisions
164
165 Total number of collisions: 16
```

Hash function 2 applied to dataset with 10k items:

The screenshot shows a code editor interface with several tabs at the top: FinalAssignment4.cpp, flightfunctions.cpp, myList.h, flightfunctions.h, flighthtable.h, and flightticket.h. The main pane displays the content of flightfunctions.cpp, which includes an implementation of a hash function for a FlightHASHTABLE. Below the code, there are tabs for PROBLEMS, OUTPUT, TERMINAL, and DEBUG CONSOLE. The OUTPUT tab shows the command-line interface of the program, which includes a help menu and a command >import CSVDatasets/flight-ticket10K.csv. The DEBUG CONSOLE tab shows the total number of collisions: 2210.

```
117 }
118 FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {}
120
121 FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
122 {
123     delete[] buckets;
124 }
125
126 long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
127 {
128     int hashVal = 0;
129
130     for(int i = 0; i<key.length(); i++)
131         hashVal = 37*hashVal+key[i];
132
133     hashVal %= capacity;
134
135     if(hashVal<0)
136         hashVal += capacity;
137
138     return hashVal; //makes sure the hashkey does not exceed capacity of the hash table
139 }
140
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>import CSVDatasets/flight-ticket10K.csv
9922

import <path> :Import flight-tickets from a CSV file
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>count_collisions
Total number of collisions: 2210
```

Hash function 2 applied to dataset with 100k items:

The screenshot shows a code editor interface with several tabs at the top: FinalAssignment4.cpp, flightfunctions.cpp, myList.h, flightfunctions.h, flighthtable.h, and flightticket.h. The main pane displays the content of flightfunctions.cpp, which includes an implementation of a hash function for a FlightHASHTABLE. Below the code, there are tabs for PROBLEMS, OUTPUT, TERMINAL, and DEBUG CONSOLE. The OUTPUT tab shows the command-line interface of the program, which includes a help menu and a command >import CSVDatasets/flight-ticket100K.csv. The DEBUG CONSOLE tab shows the total number of collisions: 2396.

```
117 }
118 FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {}
120
121 FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
122 {
123     delete[] buckets;
124 }
125
126 long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
127 {
128     int hashVal = 0;
129
130     for(int i = 0; i<key.length(); i++)
131         hashVal = 37*hashVal+key[i];
132
133     hashVal %= capacity;
134
135     if(hashVal<0)
136         hashVal += capacity;
137
138     return hashVal; //makes sure the hashkey does not exceed capacity of the hash table
139 }
140
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>import CSVDatasets/flight-ticket100K.csv
99182

import <path> :Import flight-tickets from a CSV file
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>count_collisions
Total number of collisions: 2396
```

Hash function 3 applied to dataset with 1k items:

```
G FinalAssignment4.cpp G flightfunctions.cpp x C myList.h C flightfunctions.h C flighthtable.h C flightticket.h
G flightfunctions.cpp > FlightHASHTABLE::count_collisions()
114     buckets = new HashList<hashNode>[capacity]; //Array of a list/chain
115     this->capacity = capacity; //Capacity of the bucket
116     this->size = 0; //Initial size of the hashtable
117 }
118 FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {}
120 FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
121 {
122     delete[] buckets;
123 }
124 }

long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
125 {
126     int m = key.length();
127     unsigned int h = 0;
128     for (int i = 0; i < m; i++)
129     {
130         h = (h << 5) | (h >> 27);
131         h += (unsigned int)key[i];
132     }
133     return h % this->capacity;
134 }
135
136
137

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
_____
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>import CSVdatasets/flight-ticket1k.csv
990

import <path> :Import flight-tickets from a CSV file
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>count_collisions
Total number of collisions: 162
```

Hash function 3 applied to dataset with 10k items:

```
G FinalAssignment4.cpp G flightfunctions.cpp x C myList.h C flightfunctions.h C flighthtable.h C flightticket.h
G flightfunctions.cpp > FlightHASHTABLE::count_collisions()
114     buckets = new HashList<hashNode>[capacity]; //Array of a list/chain
115     this->capacity = capacity; //Capacity of the bucket
116     this->size = 0; //Initial size of the hashtable
117 }
118 FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {}
120 FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
121 {
122     delete[] buckets;
123 }
124 }

long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
125 {
126     int m = key.length();
127     unsigned int h = 0;
128     for (int i = 0; i < m; i++)
129     {
130         h = (h << 5) | (h >> 27);
131         h += (unsigned int)key[i];
132     }
133     return h % this->capacity;
134 }
135
136
137

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
_____
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>import CSVdatasets/flight-ticket10k.csv
992

import <path> :Import flight-tickets from a CSV file
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinaday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>count_collisions
Total number of collisions: 2214
```

Hash function 3 applied to dataset with 100k items:

The screenshot shows a code editor with several tabs open. The active tab is `flightfunctions.cpp`, which contains C++ code for a hash table. Below the code editor is a terminal window displaying command-line interactions with the program.

```
FinalAssignment4.cpp flighthashtable.h flightfunctions.h flightfunctions.h
flightfunctions.cpp > FlightHASHTABLE::count_collisions()
114     buckets = new HashList<HashNode>[capacity]; //Array of a list/chain
115     this->capacity = capacity; //Capacity of the bucket
116     this->size = 0; //Initial size of the hashtable
117 }
118 FlightHASHTABLE::FlightHASHTABLE() //empty constructor
119 {}
120
121 FlightHASHTABLE::~FlightHASHTABLE() //destructor for the hashtable
122 {
123     delete[] buckets;
124 }
125
126 long FlightHASHTABLE::hash(string key) //hash function to generate a hash key
127 {
128     int m = key.length();
129     unsigned int h = 0;
130     for (int i = 0; i < m; i++)
131     {
132         h = (h << 5) | (h >> 27);
133         h += (unsigned int)key[i];
134     }
135     return h % this->capacity;
136 }
137

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>import CSVdatasets/flight-ticket100K.csv
99182

import <path> :Import flight-tickets from a CSV file
export <path> :Export flight-tickets to a CSV file
count_collisions :Print Number of Collisions
add :Add a new flight-ticket
delete <key> :Delete a flight-ticket
find <key> :Find a flight-ticket's details
allinday <date> :Display all flight-tickets in a day
printASC <key> :Print flight-tickets in ascending order
exit :Exit the program
>count_collisions
Total number of collisions: 2400
```