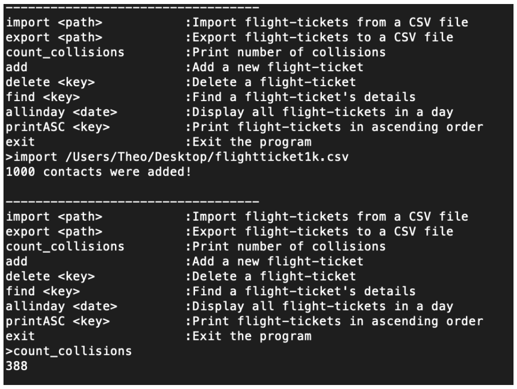
Hash Function #1

The first hash function I decided to implement was the cycle-shift hash function. However instead of strictly sending the entire key through the cycle-shit I decided I would only send the airlines name through the cycle shift and then merely add the flight number to the end of the integer value created when sending the airline name through the cycle shift hash function. My thinking behind this was that the flight number would never exceed that of the array size (at least not in the sample csv files we were given). This would mean that no two flights from the same airline would hash to the same space in the array (unless their key was identical). This would mean the only collisions that would occur are the ones with different airlines. I thought this seemed to be a fairly good strategy.

1) I first created the key with a comma separating the airline name and the flight number. I was then able to split these two strings into their own respective variables.
2) I then applied the cycle shift algorithm to the airline name and got an integer in return.
3) Finally I added the integer value of the flight number too the integer value of the airline name and modulus by the capacity.

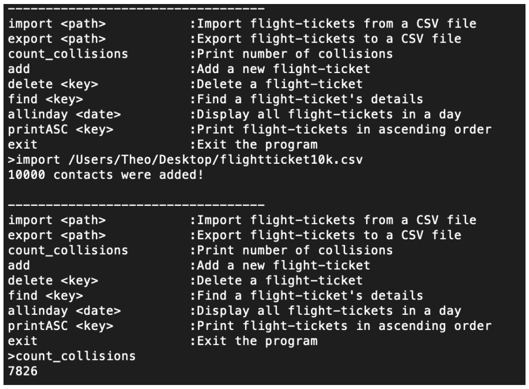
1K



```
---------------------------------
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
>import /Users/Theo/Desktop/flightticket1k.csv
1000 contacts were added!

---------------------------------
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
388
```

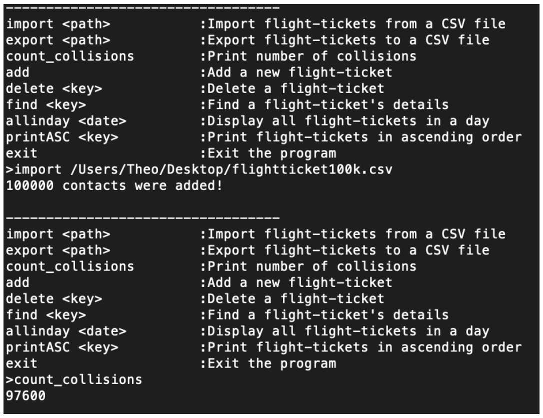
10K



```
---------------------------------
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
>import /Users/Theo/Desktop/flightticket10k.csv
10000 contacts were added!

---------------------------------
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
7826
```

100K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket100k.csv
100000 contacts were added!

----------------------------------
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
97600
```

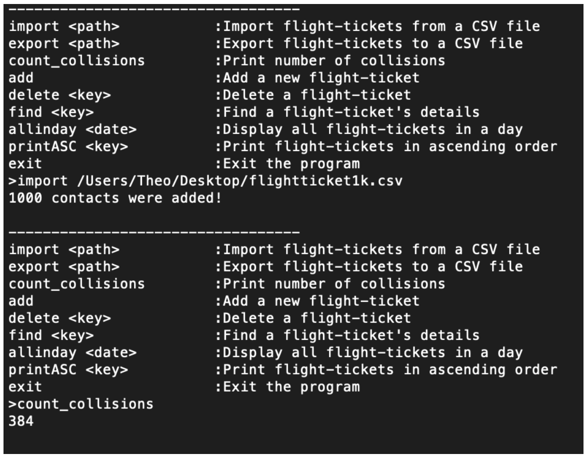
1K - 388
10K - 7,826
100K - 97,600

Hash Function #2

For this hash function i decided to take the same approach as hash function #1 except apply the Polynomial Hash code instead. I still split the key into airline name and flight number. And only applied the hash function to the airline name.

1) I first created the key with a comma separating the airline name and the flight number. I was then able to split these two strings into their own respective variables.
2) I then applied the Polynomial Hash algorithm to the airline name and got an integer in return.
3) Finally I added the integer value of the flight number too the integer value of the airline name and modulus by the capacity.

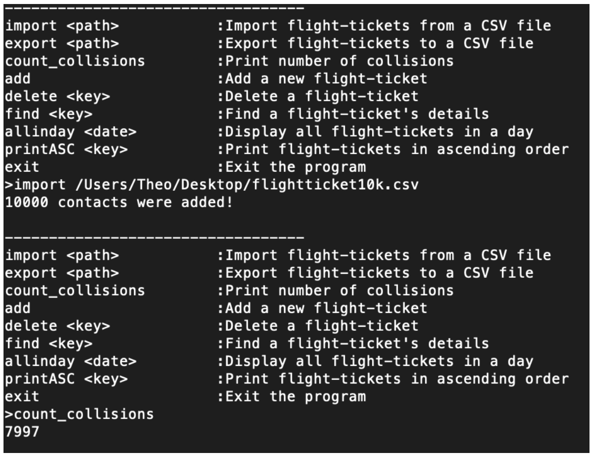
1K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket1k.csv
1000 contacts were added!

----------------------------------
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
384
```

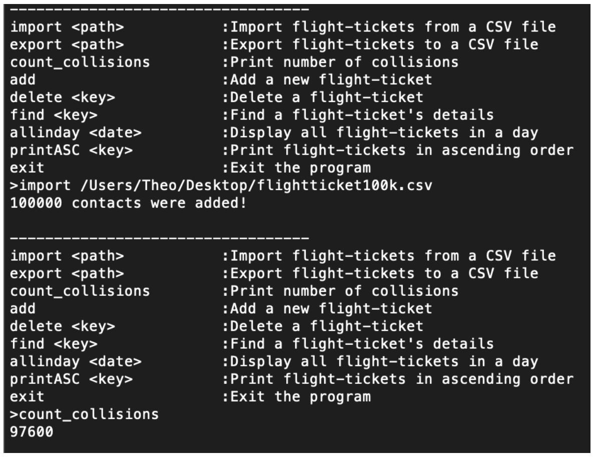
10K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket10k.csv
10000 contacts were added!

----------------------------------
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
7997
```

100K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket100k.csv
100000 contacts were added!

----------------------------------
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
97600
```

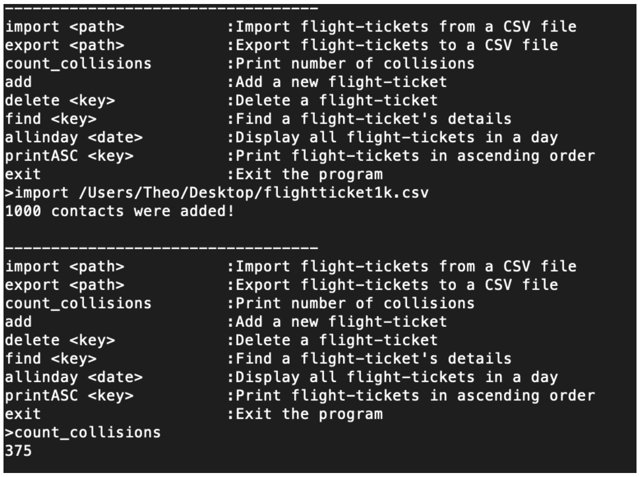
1K - 384
10K - 7,997
100K - 97,600

Hash Function #3

For the third hash function I decided to manipulate the Polynomial Hash we learned in class by adding the values of first and last letters in the airline name before applying the hash function. My reasoning behind doing this is that it increases the range in which the initial value will be mapped to. If a-z is 0-26, then adding two letters gives a range of 0-52. I know that this may not necessarily impact the hash function as you % by the capacity anyway. However I wanted to see if this had a impact compared to the Hash Function #2.

4) I first created the key with a comma separating the airline name and the flight number. I was then able to split these two strings into their own respective variables.
5) I then applied the Polynomial Hash algorithm to the addition of the firsta and last letters in the airline name and got an integer in return.
6) Finally I added the integer value of the flight number too the integer value of the airline name and modulus by the capacity.

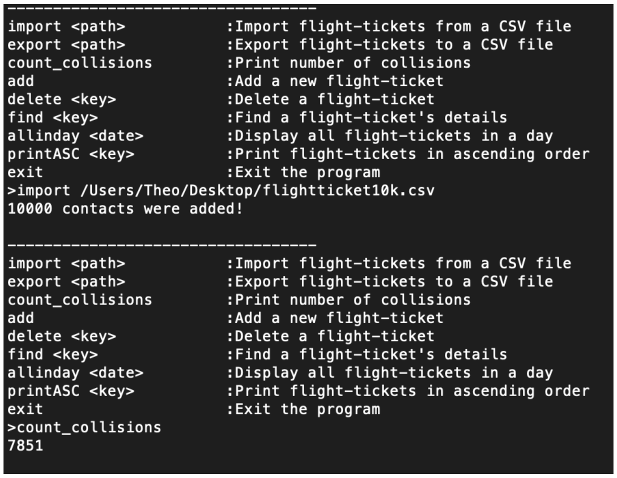
1K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket1k.csv
1000 contacts were added!

----------------------------------
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
375
```

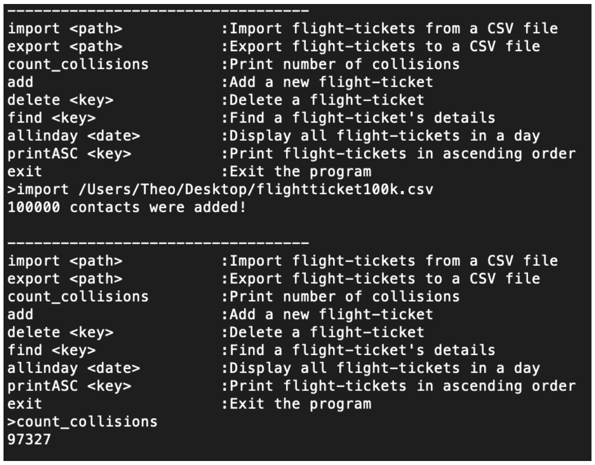
10K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket10k.csv
10000 contacts were added!

----------------------------------
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
7851
```

100K



```
----------------------------------
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
>import /Users/Theo/Desktop/flightticket100k.csv
100000 contacts were added!

----------------------------------
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
97327
```

1K - 375
10K - 7,851
100K - 97,327

Conclusion

Hash 1	**Hash 2**	**Hash 3**
1K - 388 10K - 7,826 100K - 97,600	1K - 384 10K - 7,997 100K - 97,600	1K - 375 10K - 7,851 100K - 97,327

From this you can tell all hash functions did not perform well when the import number for the values continues to increase. In fact the 100K hash functions were quite terrible. However when the import number is a mere 1K the hash functions did not preform that badly.

If I were to pick the best hash function out of the three I would pick Hash #3. In both the 1K and 100K trials this hash had the fewest collisions. Although Hash1 was not that different in both cases and in fact had fewer collisions in the 10K import file, I would still give the edge to Hash 3.

I thought it was interesting however that the Hash3 which added the first and last letters together before applying the polynomial hash function outperformed the normal polynomial hashfunction (hash 2) in every trial. Although not drastically I still thought this was interesting.