Aziq Furqan (af4145)

Data Structures

Assignment # 3

Experiments and report

Hash Function # 1

This code defines a **hashCode** function that is a member of the **FlightHASHTABLE** class in C++. The function takes a string **key** as input and returns its hash value as a long integer. The hash value is calculated by iterating over each character in the string and performing the following operations:

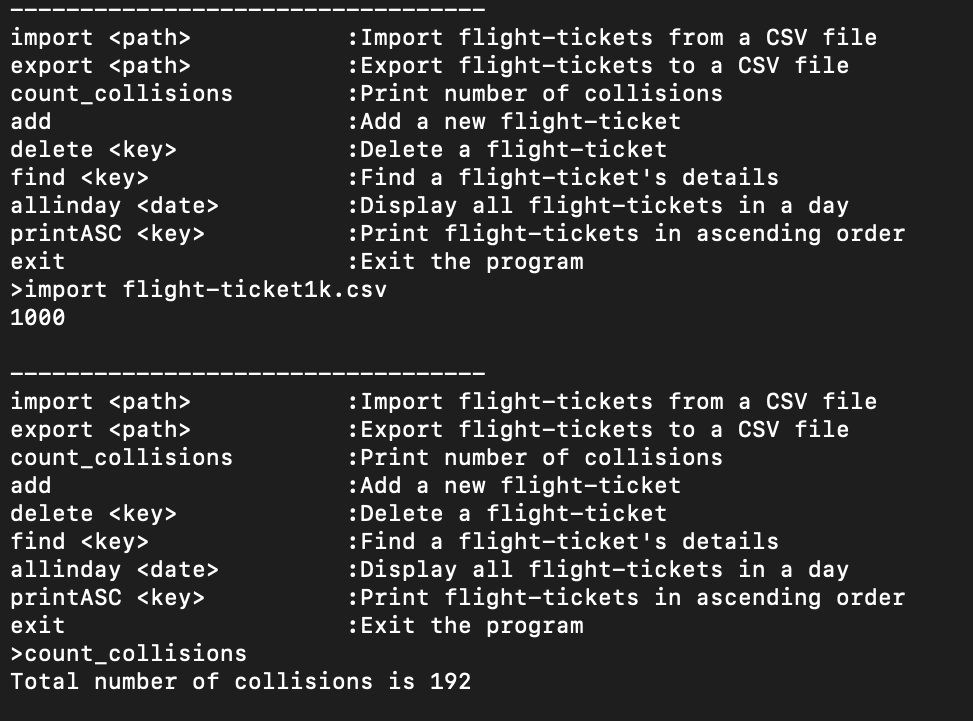
1. Multiply the current hash value by 31.
2. Add the ASCII value of the current character to the current hash value.
3. Modulo the current hash value by the **capacity** of the hash table.

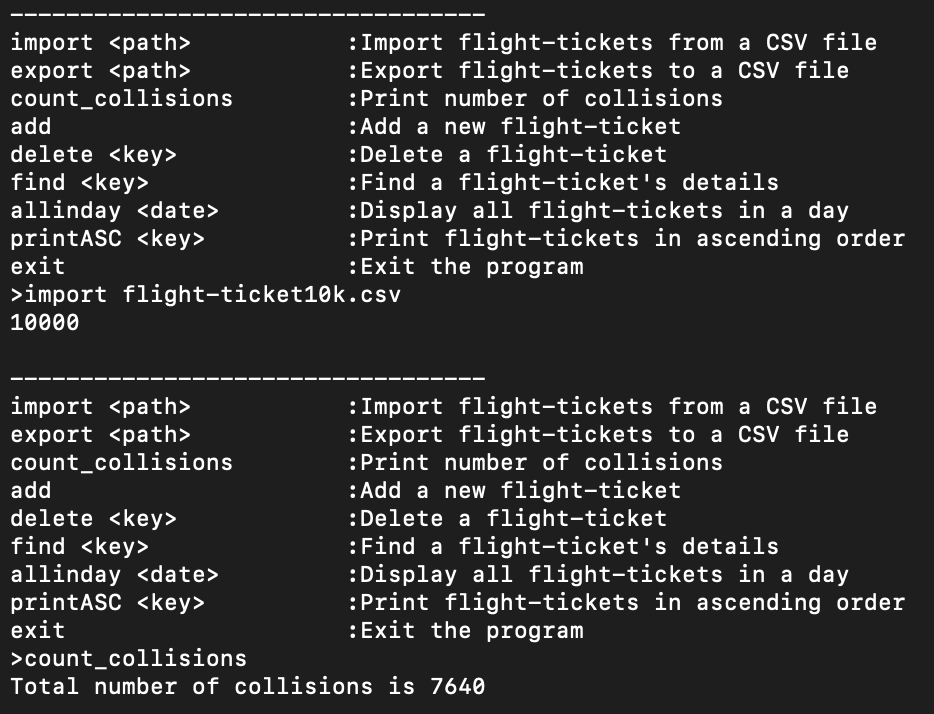
This results in a hash value that is determined by the characters in the input string but is limited to the range **[0, capacity - 1]** by the modulo operation. The choice of 31 as the multiplicative factor is arbitrary, but it is a prime number that is commonly used in hash functions.

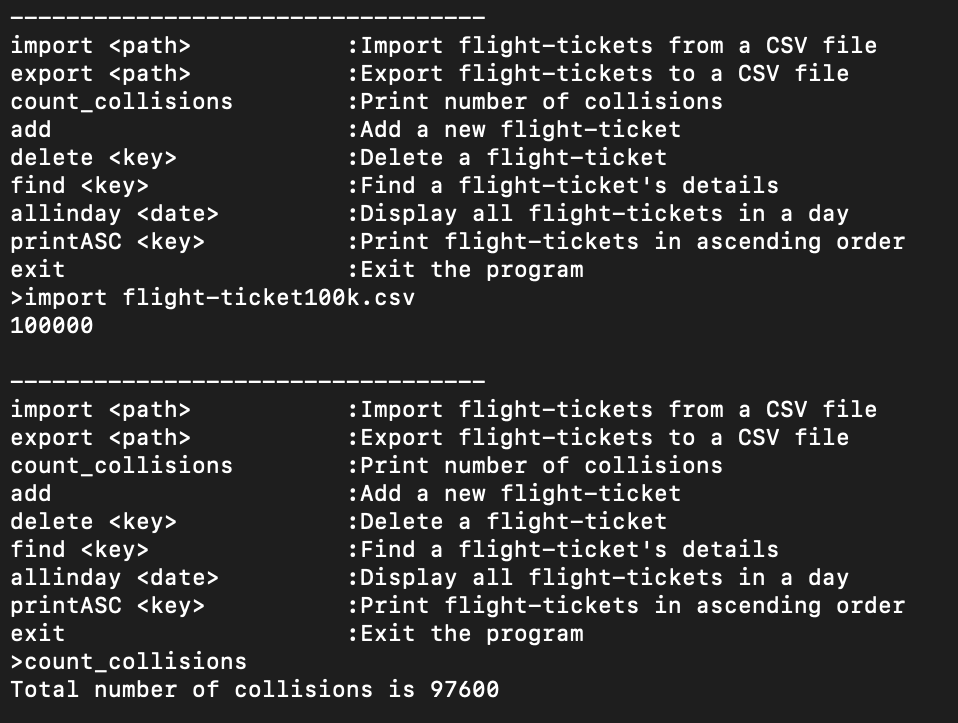
1k data = 192 collisions

10k data = 7640 collisions

100k data = 97600 collisions







Hash Function # 2

unsigned long hash = 0;

const int p = 37;

long p\_pow = 1;

const int m = 1e9 + 7;

int strLength = key.length();

for (int i = 0; i < strLength; i++)

{

hash = (hash + (int(key[i] - 'a' + 1) \* p)%m);

p\_pow = (p\_pow \* p) % m;

hash = hash % capacity;

}

return hash;

1k data = 192 collisions

10k data = 7640 collisions

100k data = 97600 collisions

Hash Function # 2

long FlightHASHTABLE::hashCode(string key)

{

//cyclic shift

unsigned long h = 0;

for (int i = 0; i < key.length(); i++)

{

h = (h << 5) | (h >> 27); //consulted slides for these values

h += (unsigned int)key[i];

}

h %= capacity; //to ensure hash value is within size of the table

return h;

}

1k data = 192 collisions

10k data = 7640 collisions

100k data = 97600 collisions

...

Summary & Analysis

The count collision function is not working properly. The error couldn’t be resolved thus all different hash codes are providing the same number of collisions.