Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

3D5A DATA STRUCUTRES AND ALGORITHM

Assignment 1 Report

Prepared By:

PULKIT SHARMA

19323659

Computer Engineering

Task 1

Implemented the given hash function in the assignment to store the given Irish surnames in the given file “names.csv”.

```
C:\Users\spulk\Desktop\DEV\task1.exe
CSV loaded....!!!!

Capacity                        -> 100
Num terms                       -> 51
Collisions                      -> 11
Load                            -> 0.390

Enter name to check frequency or 'quit' to exit
Dun
 Dun is in the hash table with frequency 1
sadsadsa
sadsadsa is not in the hash table
quit

--------------------------------
Process exited after 10.92 seconds with return value 0
Press any key to continue . . .
```

- Taking size of the table as 100, collisions reported were 11 with load factor 0.39.
- There are collisions due to the entries not spreading efficiently in the hash table.

Task 2

Implemented a different hashing function(hash2) with the same size of table, the results were as follows:

```
C:\Users\spulk\Desktop\DEV\task2.exe
CSV loaded....!!!!

Capacity                        -> 100
Num terms                       -> 51
Collisions                      -> 7
Load                            -> 0.390

Enter name to check frequency or 'quit' to exit
```

- Taking size of the table as 100, collisions reported were 7 with load factor 0.39.

```
int hash = 0;
  int count =1;
  int n;
  while (*s) {
      hash = (((hash + *s) * count))% TABLE_SIZE ;
      s++;
      count*=3;
  }
  return hash;
```

- I used this self-made hash function after reading different types of hashing functions from https://www.geeksforgeeks.org/hashing-data-structure/
- This function takes the value of count in the multiple of 3 to make it more random than the hash function in first part.
- This random function helps spread out the indexes in the hash table more efficiently resulting in less collisions.
- There is no problem in keeping the value in range because it is taken care of in the update_table function of the code.
- Therefore, this function becomes more efficient than the one provided.

Task 3

Implemented a different hashing function(hash2) with the same size of table, the results were as follows:

C:\Users\spulk\Desktop\DEV\task3.exe

```
CSV loaded....!!!!

Capacity                        -> 100
Num terms                       -> 51
Collisions                      -> 10
Load                            -> 0.390

Enter name to check frequency or 'quit' to exit
```

- Using the same table parameters with double hashing algorithm results in less collisions than normal hashing.
- Taking size of the table as 100, collisions reported were 10 with load factor 0.39.

```
int hash2(char *s){
    int hash = 0;
    while(*s){
        hash = hash + *s;
        s++;
    }
    hash = 11 - (hash % 11);
    return hash;
}
```

- The double hashing function is further helping in spreading the entries in the hash table.
- This hashing function is modding the summation of ascii values by 11 and then subtracting the result by 11.
- This is a self-made random hashing function which clearly is helping making the hashing more efficient.
- NOTE: The program is running only for truncated csv and not people csv because of memory limitations.

Task 4

Implemented the code correctly with some minor bugs. The surname is now the key and the program is taking the input from user.

```
C:\Users\spulk\Desktop\DEV\Assignment1_Task4.exe
csv file loaded!

Capacity        : 100
Num Terms       : 21
Occupied        : 21
Collisions      : 3
Load            : 0.210%
Enter surname to get full name or type 'quit' to escape
--> Digbie
Digbie Mentioned        9826    838072r129      0
--> quit

--------------------------------
Process exited after 19.36 seconds with return value 0
Press any key to continue . . .
```

- It is only working on Truncated data on Dev C++ because of memory limitation.
- It is successfully hashing the data into the table with only 3 collisions.

Note: The programs were run on Linux in the lab while demonstrating it to the lab attendant. The laptop didn't meet requirements for running Linux virtual machine, therefore I used dev c++ for making the report.