Files

HashTable.h and HashTable.cpp

```
class HashTable{
    private:
        int size, count;
        string* table;
    public:
        HashTable(int);
        bool find(string);
        void insert(string);
        void grow(string*);
        int magicalNumber(string);
        ~HashTable();
};
```

main.cpp

```
int main(int argc, char *argv)
```

Functionality

The user can either enter a file name in the command line as an argument when running the program or the user can say ./hash-table , wait for the program to start then enter the size of the initial Hash Table. After that, the user must enter 1 to insert a value to the tale and 2 to search for the value. The program will only write out to the command line when the user enters 2 and a value with it.

HashTable(int)
Constructor to create a hash table that takes in the size of the hash table
bool find(string)
Searches through the hash table by creating a key and then going to that value in the array
void insert(string)
Generates a key dependent on the value that is going to be stored and then finds the nearest empty location in the array to insert the value into
void grow(string*)
Doubles the size of the hash table when the table is full but before the user tries to insert another element
int magicalNumber(string)
Used as the hash function to generate a key. The method is called in both the find and the insert functions.
~HashTable()
Deconstructor to delete the array once the program terminates

Method Detail

HashTable

public HashTable(int size)

Creates an array of the size specified in the parameter. Also sets the count equal to zero.

Parameter:

The size of the array to be constructed

Throws:

Out of Memory exception

find

public bool find(string value)

Search for the parameter in the array by comparing it to other contents in the array.

Parameter:

value - element to be searched for in the array.

Returns:

True if the value is found and false if the value is not found.

insert

public void insert(string value)

Uses the array that has already been created and generates a key based on the value that the method took in to store the element.

Parameter:

value – the element you are inserting into the array.

grow

pubic void grow(string* temp)

Overwrites the existing array of the hash table but the array is double the size so that it is no longer full.

Parameter:

temp – the array that is full

magicalNumber

public int magicalNumber(string value)

Uses the char values of the given string to generate a key so that there is some consistency when different words are entered.

Parameter:

value – The string that the user wishes to either store or search for.

~HashTable

public ~HashTable()

Deletes the array that was used in the program.