CSCE 221 Cover Page
Homework Assignment

First Name: Jialu Last Name: Zhao UIN: 426000712

User Name: zhaojailu123 E-mail address: zhaojialu123@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero, read more in the Aggie Honor System Office http://aggiehonor.tamu.edu/

Type of sources	web sources			
People				
Web pages (provide URL)	https://en.wikipedia.org/wiki/Hash_table			
Printed material				
Other Sources				

I certify that I have listed all the sources that I used to develop the solutions/code to the submitted work.

"*On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work.*"

Your Name (signature) Jialu Zhao Date 04/08/2017

CSCE 221 — Programming Assignment 5 (100 points)

Due: April 13th, 2017 at 11:59 pm

1. Assignment number and its description

(1)Read input.csv containing grades.

(2)Use the Regex class to parse each row in input.csv.

(3)Create a hash table using student's UIN as a key and student's score as a value (key-value pair). Resolve collisions using the chaining method.

Read the lecture notes and text book about Hashing, chap. 9.

Display the statistics about the hash table: minimum, maximum, and average length of the linked lists in the hash table.

(4)Read the roster.csv containing students grades (= class grading book).

(5)Use the Regex to parse each row in roster.csv.

(6)Look up the hash table by using the parsed UIN and recover the corresponding quiz score.

(7)Create a new file output.csv with appended scores, see more details below:

(a)read a line from roster.csv

(b)extract UIN field and create the corresponding key

(c)search the hash table using the key

(i)if the search is successful, update the line by appending the corresponding score

(ii)if the search is unsuccessful, copy the roster line without any changes

2. Description of data structures and algorithms used by your program.

(a)data structures and algorithms on regex

(i)data structures:

Regular Expressions Regular expressions are a standardized way to express patterns to be matched against sequences of characters.

The standard C++ library provides support for regular expressions in the <regex> header through a series of operations. All these operations make use of some typical regex parameters:

Regular expression (pattern): The pattern which is searched for in the target sequence. This must be an object of a basic_regex type (such as regex), generally constructed from a string with a special syntax that describes what constitutes a match (see ECMAScript syntax).

(ii)Algorithm1(parse in input.csv)

```
Algorithm:
regex pattern{R"(([\s\S]+,)([\s\S]+)(,)([\s\S]+))"};
step 1: set pattern, which has four substrings
step 2: using regex_search to find 4 different substrings
step 3: insert the substrings that we need to hash table
```

(iii)Algorithm2(parse in roster.csv)

```
Algorithm:
regex pattern{R"(([\s\S]+),([\s\S]+),([\s\S]+),)"};
step 1: set pattern, which has three substrings
step 2: using regex_search to find 3 different substrings
step 3: search the substrings that we need in hash table
```

(b)data structures and algorithms on hash table

(i)data structures:

a hash table is a data structure which implements an associative array abstract data type, a structure that can map keys to values. A hash table uses a hash function to compute an index into an array of buckets or slots, from which the desired value can be found.

Ideally, the hash function will assign each key to a unique bucket, but most hash table designs employ an imperfect hash function, which might cause hash collisions where the hash function generates the same index for more than one key. We use chain method to solve collisions in this programming assignment.

(ii) Algorithm:
Hashing:
Given a key, the algorithm computes an index that suggests where the entry can be found:

```
index = f(key, array_size)
```

Often this is done in two steps:

```
hash = hashfunc(key)
index = hash % array_size
```

3. Description of input and output data.
(1)Input: roster.csv and input.csv, all of them are excel file.
(a)roster.csv:
from the roster, we read it row by row and we can get the size of our hash table.
(b)input.csv:
from the input, we read it row by row and insert UIN as keys and grades as values one by one.
(2)Output:output.csv, which is also excel file; statistic data of linkedlist in the hash table
(a)output.csv:
a new file output.csv with appended scores
(b) statistic data:
Display the statistics about the hash table, including maximum, minimum and average length of the linkedlist in the hash table
4. How have you tested your program for corrections.
(a)output.csv

14	Aubrey Everc	tempor.augu	800000027				
15	Aurelia Donc	metus.eu.era	800000010				
16	Alfonso Livin	mattis@dolo	800000092				
17	Alice Carr	Fusce@tristi	800000031				
18	Armand Bow	nec@tellusP	800000048				
19	Aubrey Everc	tempor.augu	800000027				
20	Aurelia Donc	metus.eu.era	800000010				
21	Barbara Wall	Donec@plac	800000005	84			
22	Basia Burch	ut.aliquam@	800000061				
23	Basil Moon	Donec.nibh@	800000081				
24	Beatrice Mor	augue@temp	800000055	47			
25	Bertha Rodri	Duis.volutpa	800000032				
26	Branden Ras	nisl@eu.ca	800000077				
27	Candace Bish	adipiscing@l	800000057	49			
28	Carson Thom	Suspendisse.	800000049				
29	Cathleen Bov	habitant@ele	800000020				
30	Celeste Nguy	Sed.nulla@Fi	800000094				
31	Chadwick Cla	et@viverra.e	800000073				
32	Charles Merc	vitae@sapien	800000030				
33	Chiquita Bea	erat@aliquai	800000056				
34	Clayton Hern	eu.lacus.Quis	800000006				
35	Conan Daugh	aliquam.arcu	800000053	96			
36	Danielle Bate	sed.orci@idli	800000012				
37	Deacon Richa	eget.dictum@	800000066	90			
38	Debra Hall	Etiam.imperc	800000058				
39	Delilah Mcin	nunc@quam	800000011				

output

(b)statistic data:

```
The maximum length of the linkedlist in the hash table:1
The minimum length of the linkedlist in the hash table:0
The average length of the linkedlist in the hash table:0.17
```

- Are the computational results about the hashing consistent with the expected running time for the hashing algorithm? Justify your answer.

The average length is 0.17 which is really small and I consider it to be neglegible, so the running time is a constant. And the expected running time is O(1), so the compuattaional results about the hashing consistent with the expected running time for the hashing algorithm.
4.C++ features or standard library classes
(a)The standard C++ library provides support for regular expressions in the <regex> header through a series of operations.
(b)read from excel file:

```
#include <sstream>
#include <fstream>
```

5. list of class declaration

(1)TemplatedDoublyLinkedList.h

changed from doubly linked list by making two elements for each class: key and value. Used to make linked list of hash table in order to solve collision by chain method.

Also has get_length function which help to get the statistics in the hash table.

(2)HashTable.h

HashTable class, with search and insert class to insert node to hash table and search specific elements in the hash table.

Also has get_min, get_max, and get_average function which help to display statistics.

6. How to compile and run program

compile: g++ -std=c++17 main.cpp

run: ./a.out

7.conclusion:

(1)The maximum length of the linkedlist in the hash table is 1;

(2)The minimum length of the linkedlist in the hash table is 0;

(3)The average length of the linkedlist in the hash tabls is 0.17;

(4) The computational results about the hashing consistent with the expected running time for the hashing algorithm;

(5) The running time of the hashing is O(1).