Justin Stewart & Micah Geertson
CPSC 335
Project 4
05/07/2016

Programming Assignment #4

Table of Contents

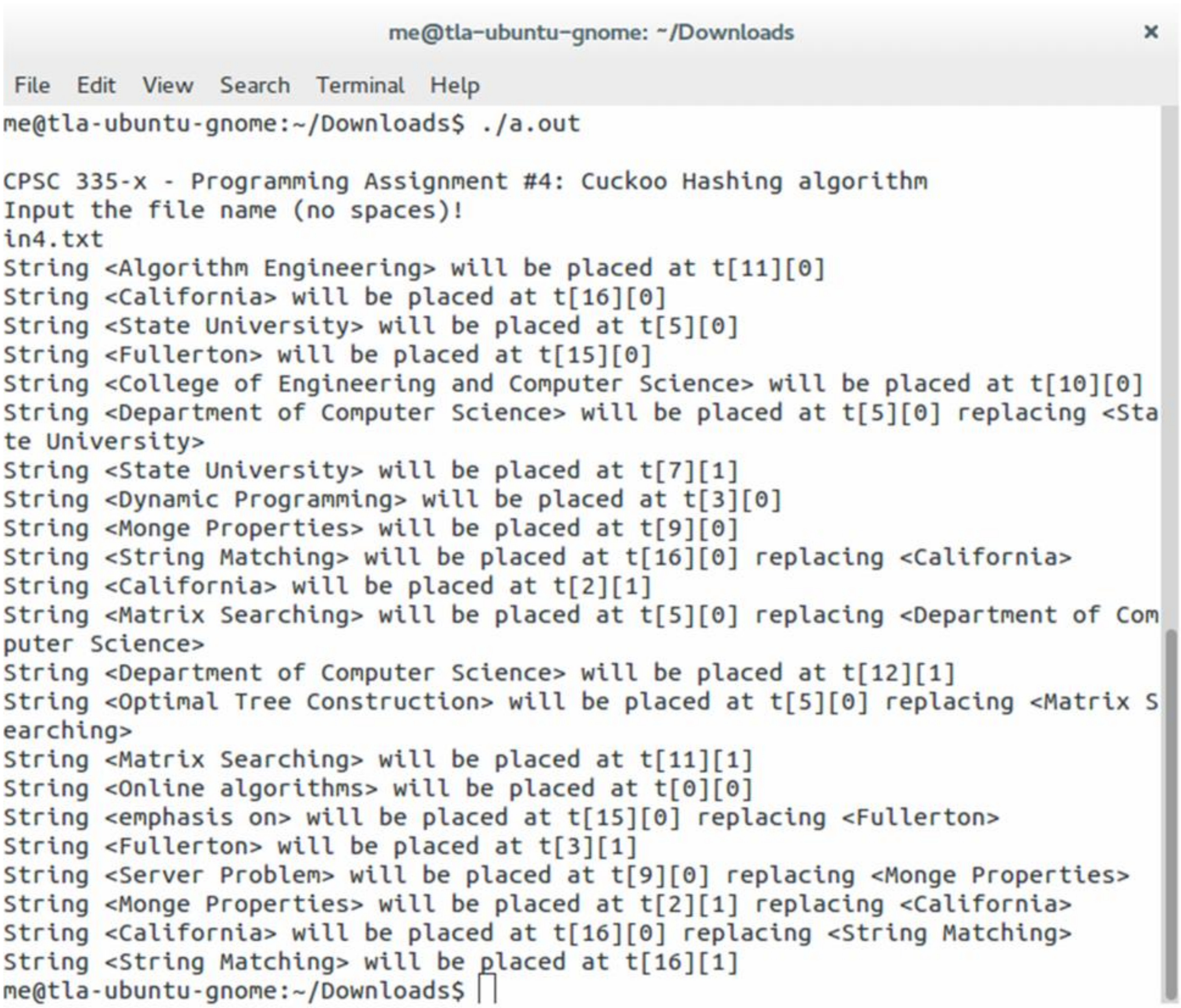
```
def place_in_hash_tables(s)
      placed = false
      counter = 0
      index = 0

      copy s into temp_s

      pos = f(temp_s, index)

      while not placed AND counter less than 2 * tablesize do
            if table[pos][index] is empty do
                  copy temp_s into table[pos][index]
                  placed = true
                  return placed
            else do
                  copy table[pos][index] into temp
                  copy temp_s into table[pos][index]
                  copy temp into temp_s

                  index = (index + 1) mod 2

                  pos = f(temp_s, index)
                  inc counter

      return placed

def f(s, index)
      po = 1
      len = length of s

      if index equals 0 do
            val = s[0]
            val = val mod tablesize

            if val less than 0 do
                  val = val + tablesize

            if len equals 1 do
                  return val

            for i = 1 to len - 1 do
                  temp = s[i]
                  po = po * 31

                  po = po mod tablesize

                  if po less than zero do
                        po = po + tablesize

                  val = val + temp * po
                  val = val mod tablesize

                  if val less than zero do
                        val = val + tablesize
            return val
      else
            val = s[len - 1]
            val = val mod tablesize

            if val less than zero do
                  val = val + tablesize

            if len equals 1 do
                  return val

            for i = 1 to len - 1 do
                  temp = s[len - i - 1]
                  po = po * 31

                  po = po mod tablesize

                  if po less than zero do
                        po = po + tablesize

                  val = val + temp * po
                  val = val mod tablesize

                  if val less than zero do
                        val = val + tablesize
            return val
```

```
// Assignment 4: Cuckoo Hashing algorithm
// Micah Geertson & Justin Stewart
// An open addressing method called Cuckoo Hashing
// INPUT: an input file containing strings of characters, one string per line
// OUTPUT: a detailed list of where the strings are inserted.

#include <iostream>
#include <cstring>
#include <stdio.h>

using namespace std;

// cuckoo tables' size
const int tablesize = 17;
// combine the two 1-dimensional table into one 2-dimensional table
char  t[tablesize][2][255];

// compute the hash functions
size_t f(char*, size_t);

// place a string in one of the hash tables
bool place_in_hash_tables(char*);

int main() {

    // the strings to be stored in the hash tables
    char s[255] = "";
    char null_st[] = "";
    size_t i, len;
    bool placed;

    // clear the tables
    for (i = 0; i< tablesize; i++) {
        strcpy(t[i][0], null_st);
        strcpy(t[i][1], null_st);
    }

    char filename[255] = "";

    // display the header
    cout << endl << "CPSC 335-x - Programming Assignment #4: ";
    cout << "Cuckoo Hashing algorithm" << endl;

    // read the strings from a file
    cout << "Input the file name (no spaces)!" << endl;
    cin >> filename;

    // open the file for reading
    FILE *file = fopen(filename, "r");
    if (file != NULL)
    {
        /* read line by line from the file */
        while (fgets(s, 255, file) != NULL) {
            // place null character at the end of the line instead of <return>
            len = strlen(s);
            s[len - 2] = '\0';
            // insert the string in the cuckoo table
            placed = place_in_hash_tables(s);
            // check whether the placement was successful
            if (!placed) {
                cout << "Placement has failed" << endl;
                return -1;
            }
        }
        fclose(file);
    }
    else
    {
        perror(filename); /* why didn't the file open? */
    }
    return 0;

}


bool place_in_hash_tables(char *s) {

    bool placed;
    size_t pos;
    int index;
    char temp_s[255], temp[255];

    strcpy(temp_s, s);

    // use a counter to detect loops
    int counter = 0;

    // start with table T1
    index = 0;

    placed = false;

    pos = f(temp_s, index);

    while ((!placed) && (counter < 2 * tablesize)) {

        if (strcmp(t[pos][index], "") == 0) {
            // the entry at index <pos> in the <index> hash table is available so store the
            string <temp_s> there
            cout << "String <" << temp_s << "> will be placed at";
            cout << " t[" << pos << "][" << index << "]" << endl;
            strcpy(t[pos][index], temp_s);
            placed = true;
            return placed;
        }
        else {
            // the entry at index <pos> in the <index> hash table is not available so
            // obtain the string stored over there in variable <temp> and store the string
            <temp_s> there
            // now the string <temp> needs to be placed in the other table
            cout << "String <" << temp_s << "> will be placed at" << " t[" << pos;
            cout << "][" << index << "]" << " replacing <" << t[pos][index] << ">";
            cout << endl;
            // YOU NEED TO WRITE THE CODE TO STORE IN temp THE STRING STORED AT
            // t[pos][index] AND STORE IN t[pos][index] THE STRING temp_s
            strcpy(temp, t[pos][index]);
            strcpy(t[pos][index], temp_s);
            strcpy(temp_s, temp);
            // NOW temp_s CONTAINING THE EVICTED STRING NEEDS TO BE STORED
            // IN THE OTHER TABLE
            // WRITE THE CODE TO SET index TO INDICATE THE OTHER TABLE
            index = (index + 1) % 2;
            // WRITE THE CODE TO CALCULATE IN pos THE HASH VALUE FOR temp_s
            pos = f(temp_s, index);
            counter++;
        }
    }
    return placed;
};


// compute the hash functions
size_t f(char *s, size_t index) {
    // s is the string (the key) to which we apply the hash function
    // index indicates which hash function will be used
    // index == 0 means the first hash function
    // index == 1 means the second hash function
    size_t po, len;
    int i, val, temp;
    po = 1;

    len = strlen(s);

    if (index == 0) {

        val = s[0];
        val = val % tablesize;
        if (val < 0) val += tablesize;

        if (len == 1)
            return val;

        for (i = 1; i < len; i++)
        {
            temp = s[i];
            po *= 31;

            po = po % tablesize;
            if (po < 0) po += tablesize;

            val += temp * po;
            val = val % tablesize;
            if (val < 0) val += tablesize;
        }
        return val;
    }
    else {
        // YOU NEED TO IMPLEMENT THE STEPS TO CALCULATE THE SECOND
        // HASH FUNCTION
        val = s[len - 1];
        val = val % tablesize;
        if (val < 0) val += tablesize;

        if (len == 1)
            return val;

        for (i = 1; i < len; i++) {
            temp = s[len - i - 1];
            po *= 31;

            po = po % tablesize;
            if (po < 0) po += tablesize;

            val += temp * po;
            val = val % tablesize;

            if (val < 0) val += tablesize;
        }
        return val;
    }
}
```

	Table T1	Table T2
[0]	Online Algorithms	
[1]		One of the greatest
[2]	Self-stabilization	Algorithm Engineering
[3]	are known	Fullerton
[4]	Quantum Nature of Universe	Server Problem
[5]	In physics and	astronomy
[6]		Optimal Tree Construction
[7]	to scientists	State University
[8]		macroscopic quantum objects
[9]	Monge Properties	
[10]	College of Engineering and Computer Science	
[11]	Some related problem	Matrix Searching
[12]		Department of Computer Science
[13]		Cuckoo hashing is fun
[14]	Greatest mysteries	Dynamic Programming
[15]	emphasis on	mysteries in science
[16]	California	String Matching

Output:

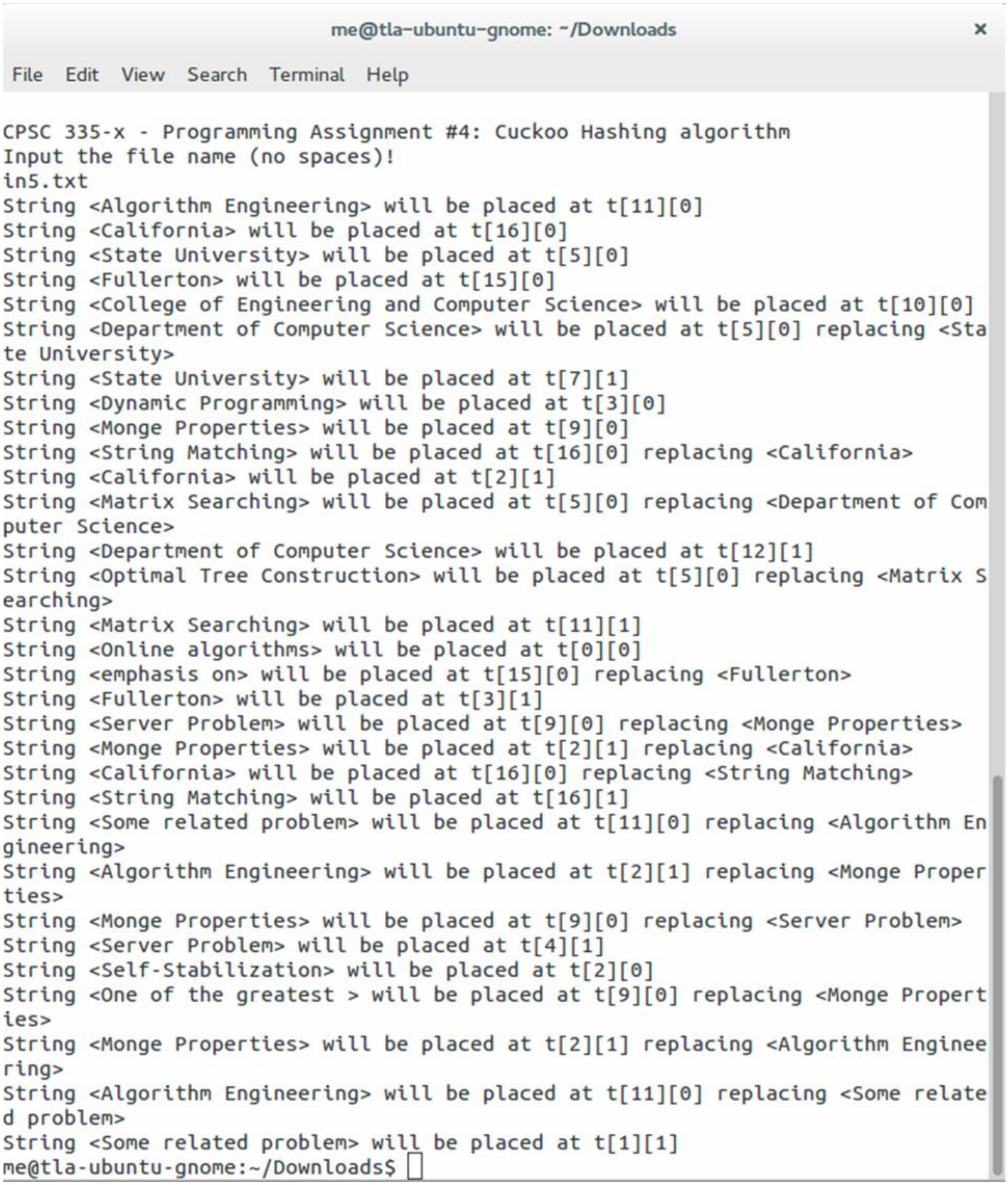
In4.txt



me@tla-ubuntu-gnome: ~/Downloads
File Edit View Search Terminal Help

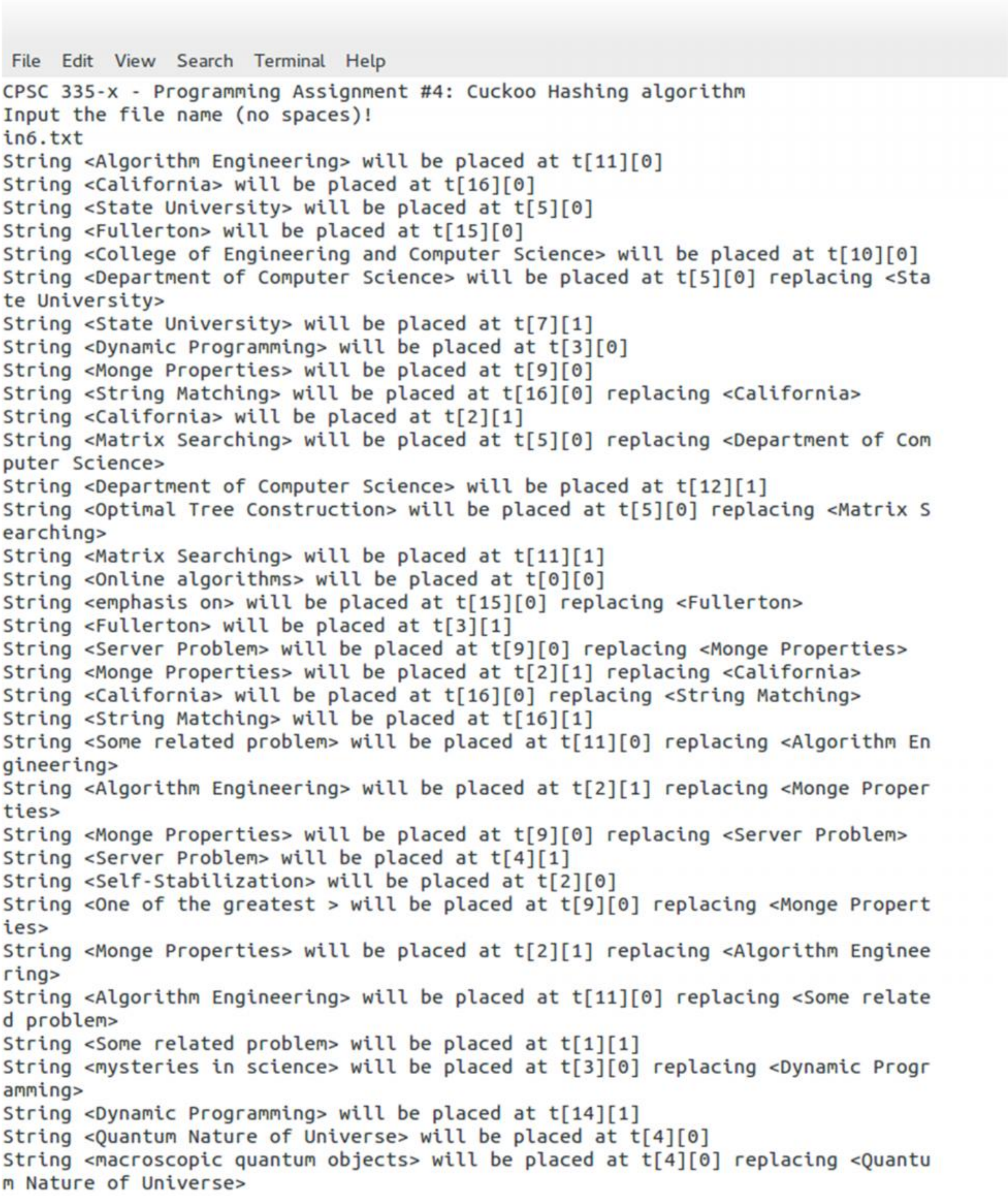
```
me@tla-ubuntu-gnome:~/Downloads$ ./a.out

CPSC 335-x - Programming Assignment #4: Cuckoo Hashing algorithm
Input the file name (no spaces)!
in4.txt
String <Algorithm Engineering> will be placed at t[11][0]
String <California> will be placed at t[16][0]
String <State University> will be placed at t[5][0]
String <Fullerton> will be placed at t[15][0]
String <College of Engineering and Computer Science> will be placed at t[10][0]
String <Department of Computer Science> will be placed at t[5][0] replacing <Sta
te University>
String <State University> will be placed at t[7][1]
String <Dynamic Programming> will be placed at t[3][0]
String <Monge Properties> will be placed at t[9][0]
String <String Matching> will be placed at t[16][0] replacing <California>
String <California> will be placed at t[2][1]
String <Matrix Searching> will be placed at t[5][0] replacing <Department of Com
puter Science>
String <Department of Computer Science> will be placed at t[12][1]
String <Optimal Tree Construction> will be placed at t[5][0] replacing <Matrix S
earching>
String <Matrix Searching> will be placed at t[11][1]
String <Online algorithms> will be placed at t[0][0]
String <emphasis on> will be placed at t[15][0] replacing <Fullerton>
String <Fullerton> will be placed at t[3][1]
String <Server Problem> will be placed at t[9][0] replacing <Monge Properties>
String <Monge Properties> will be placed at t[2][1] replacing <California>
String <California> will be placed at t[16][0] replacing <String Matching>
String <String Matching> will be placed at t[16][1]
me@tla-ubuntu-gnome:~/Downloads$
```

In5.txt

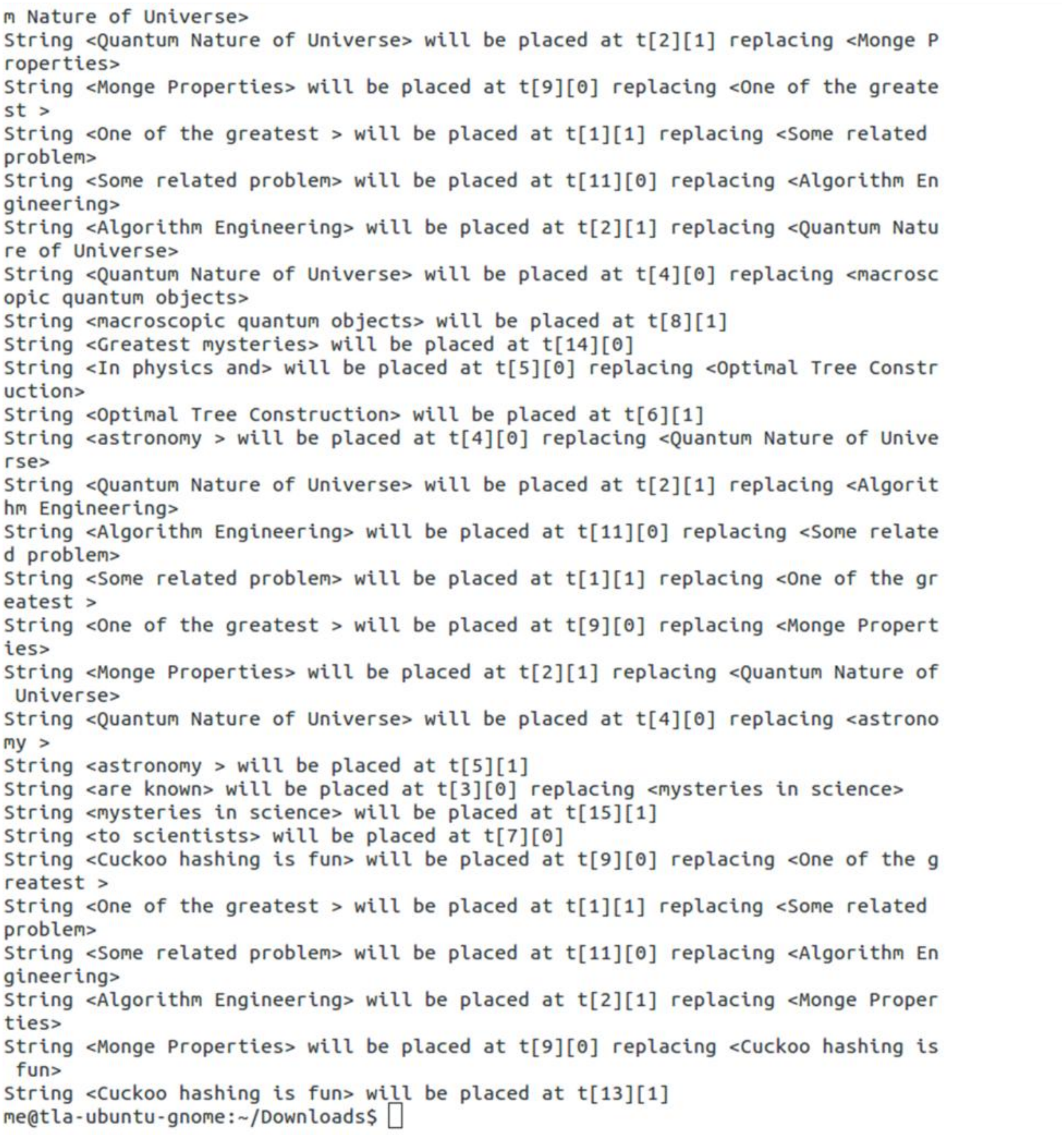


```
CPSC 335-x - Programming Assignment #4: Cuckoo Hashing algorithm
Input the file name (no spaces)!
in5.txt
String <Algorithm Engineering> will be placed at t[11][0]
String <California> will be placed at t[16][0]
String <State University> will be placed at t[5][0]
String <Fullerton> will be placed at t[15][0]
String <College of Engineering and Computer Science> will be placed at t[10][0]
String <Department of Computer Science> will be placed at t[5][0] replacing <Sta
te University>
String <State University> will be placed at t[7][1]
String <Dynamic Programming> will be placed at t[3][0]
String <Monge Properties> will be placed at t[9][0]
String <String Matching> will be placed at t[16][0] replacing <California>
String <California> will be placed at t[2][1]
String <Matrix Searching> will be placed at t[5][0] replacing <Department of Com
puter Science>
String <Department of Computer Science> will be placed at t[12][1]
String <Optimal Tree Construction> will be placed at t[5][0] replacing <Matrix S
earching>
String <Matrix Searching> will be placed at t[11][1]
String <Online algorithms> will be placed at t[0][0]
String <emphasis on> will be placed at t[15][0] replacing <Fullerton>
String <Fullerton> will be placed at t[3][1]
String <Server Problem> will be placed at t[9][0] replacing <Monge Properties>
String <Monge Properties> will be placed at t[2][1] replacing <California>
String <California> will be placed at t[16][0] replacing <String Matching>
String <String Matching> will be placed at t[16][1]
String <Some related problem> will be placed at t[11][0] replacing <Algorithm En
gineering>
String <Algorithm Engineering> will be placed at t[2][1] replacing <Monge Proper
ties>
String <Monge Properties> will be placed at t[9][0] replacing <Server Problem>
String <Server Problem> will be placed at t[4][1]
String <Self-Stabilization> will be placed at t[2][0]
String <One of the greatest > will be placed at t[9][0] replacing <Monge Propert
ies>
String <Monge Properties> will be placed at t[2][1] replacing <Algorithm Enginee
ring>
String <Algorithm Engineering> will be placed at t[11][0] replacing <Some relate
d problem>
String <Some related problem> will be placed at t[1][1]
me@tla-ubuntu-gnome:~/Downloads$ 
```

In6.txt



```
CPSC 335-x - Programming Assignment #4: Cuckoo Hashing algorithm
Input the file name (no spaces)!
in6.txt
String <Algorithm Engineering> will be placed at t[11][0]
String <California> will be placed at t[16][0]
String <State University> will be placed at t[5][0]
String <Fullerton> will be placed at t[15][0]
String <College of Engineering and Computer Science> will be placed at t[10][0]
String <Department of Computer Science> will be placed at t[5][0] replacing <Sta
te University>
String <State University> will be placed at t[7][1]
String <Dynamic Programming> will be placed at t[3][0]
String <Monge Properties> will be placed at t[9][0]
String <String Matching> will be placed at t[16][0] replacing <California>
String <California> will be placed at t[2][1]
String <Matrix Searching> will be placed at t[5][0] replacing <Department of Com
puter Science>
String <Department of Computer Science> will be placed at t[12][1]
String <Optimal Tree Construction> will be placed at t[5][0] replacing <Matrix S
earching>
String <Matrix Searching> will be placed at t[11][1]
String <Online algorithms> will be placed at t[0][0]
String <emphasis on> will be placed at t[15][0] replacing <Fullerton>
String <Fullerton> will be placed at t[3][1]
String <Server Problem> will be placed at t[9][0] replacing <Monge Properties>
String <Monge Properties> will be placed at t[2][1] replacing <California>
String <California> will be placed at t[16][0] replacing <String Matching>
String <String Matching> will be placed at t[16][1]
String <Some related problem> will be placed at t[11][0] replacing <Algorithm En
gineering>
String <Algorithm Engineering> will be placed at t[2][1] replacing <Monge Proper
ties>
String <Monge Properties> will be placed at t[9][0] replacing <Server Problem>
String <Server Problem> will be placed at t[4][1]
String <Self-Stabilization> will be placed at t[2][0]
String <One of the greatest > will be placed at t[9][0] replacing <Monge Propert
ies>
String <Monge Properties> will be placed at t[2][1] replacing <Algorithm Enginee
ring>
String <Algorithm Engineering> will be placed at t[11][0] replacing <Some relate
d problem>
String <Some related problem> will be placed at t[1][1]
String <mysteries in science> will be placed at t[3][0] replacing <Dynamic Progr
amming>
String <Dynamic Programming> will be placed at t[14][1]
String <Quantum Nature of Universe> will be placed at t[4][0]
String <macroscopic quantum objects> will be placed at t[4][0] replacing <Quantu
m Nature of Universe>
```



```
m Nature of Universe>
String <Quantum Nature of Universe> will be placed at t[2][1] replacing <Monge P
roperties>
String <Monge Properties> will be placed at t[9][0] replacing <One of the greate
st >
String <One of the greatest > will be placed at t[1][1] replacing <Some related
problem>
String <Some related problem> will be placed at t[11][0] replacing <Algorithm En
gineering>
String <Algorithm Engineering> will be placed at t[2][1] replacing <Quantum Natu
re of Universe>
String <Quantum Nature of Universe> will be placed at t[4][0] replacing <macrosc
opic quantum objects>
String <macroscopic quantum objects> will be placed at t[8][1]
String <Greatest mysteries> will be placed at t[14][0]
String <In physics and> will be placed at t[5][0] replacing <Optimal Tree Constr
uction>
String <Optimal Tree Construction> will be placed at t[6][1]
String <astronomy > will be placed at t[4][0] replacing <Quantum Nature of Unive
rse>
String <Quantum Nature of Universe> will be placed at t[2][1] replacing <Algorit
hm Engineering>
String <Algorithm Engineering> will be placed at t[11][0] replacing <Some relate
d problem>
String <Some related problem> will be placed at t[1][1] replacing <One of the gr
eatest >
String <One of the greatest > will be placed at t[9][0] replacing <Monge Propert
ies>
String <Monge Properties> will be placed at t[2][1] replacing <Quantum Nature of
 Universe>
String <Quantum Nature of Universe> will be placed at t[4][0] replacing <astrono
my >
String <astronomy > will be placed at t[5][1]
String <are known> will be placed at t[3][0] replacing <mysteries in science>
String <mysteries in science> will be placed at t[15][1]
String <to scientists> will be placed at t[7][0]
String <Cuckoo hashing is fun> will be placed at t[9][0] replacing <One of the g
reatest >
String <One of the greatest > will be placed at t[1][1] replacing <Some related
problem>
String <Some related problem> will be placed at t[11][0] replacing <Algorithm En
gineering>
String <Algorithm Engineering> will be placed at t[2][1] replacing <Monge Proper
ties>
String <Monge Properties> will be placed at t[9][0] replacing <Cuckoo hashing is
 fun>
String <Cuckoo hashing is fun> will be placed at t[13][1]
me@tla-ubuntu-gnome:~/Downloads$
```