The Cuckoo Hashing Algorithm

There are several versions of cuckoo hashing. The version we learned in class is the simplest, where there are two hash functions, and thus only two places where any given item could be stored in the table. Let us consider the set of keys to be printable ASCII strings of length no more than 80. Let us consider the hash table size to be 17 .

If key is the string representing the key, then let $keysize$ be the size of the string $key$and $key[i]$be the ASCII code of the (i+1)th character in the string key read from left to right: $key = key_0\, key_1 \ldots key_{keysize-1}$

Let us consider two different hash functions, f_1 and f_2. Function f_1 computes a large number and then brings the result into the proper range, which is $0..tablesize - 1$, using the formulas below:
$val = \sum_{i=0}^{keysize-1} \quad key[i] \cdot 41^{\,i} = 41^0 \cdot key_0 + 41^1 \cdot key_1 + \ldots + 41^{keysize-2} \cdot key_{keysize-2} + 41^{\,keysize-1} \cdot key_{keysize-1}$

$$f_1 = val \;\%\; tablesize$$

if $f_1 < 0$then f_1=f_1+ tablesize

Function f_2 also computes a large number and then brings the result into the proper range, which is $0..tablesize - 1$, using the formulas below:

$$val = \sum_{i=0}^{keysize-1} \quad key[keysize - i - 1] \cdot 41^{\,i}$$

$$= 41^{\,keysize-1} \cdot key_0 + 41^{\,keysize-2} \cdot key_1 + \ldots + 41^1 \cdot key_{keysize-2} + key_{keysize-1}$$

$$f_2 = val \;\%\; tablesize$$

if $f_2 < 0$then f_2=f_2+ tablesize

Both functions f_1 and f_2 compute first a large number then it brings the result into the proper range which is $0..tablesize - 1$. But we can bring the intermediate results into this range after each calculation, we do not need to wait until we compute the final result. Also, we can ring the power term 41^{index}into this range before multiplying it with key_{index}

You need to insert the strings below (also given in the input file in6.txt) into the hash table provided next. Please put an empty line at the end of the file.

Algorithm Engineering
California State University
Fullerton

College of Engineering and Computer Science
School of Computer Science
Greedy pattern
Monge properties
String matching
Matrix searching
Optimal tree construction
Online algorithms
emphasis on
Server Problem
Some related problem
Self-Stabilization
one of the greatest mysteries
in science
Quantum nature of universe
in physics
are known
Cuckoo Hashing is fun!

into the hash table (next page) using f_1 for the first table and f_2 for the second table. Show the result of the insertion in the table shown on the next page.

Hint: consider a two-dimensional table of strings t, where t[0] is T1 and t[1] is T2. Consider a variable index that oscillates between 0 and 1as it would have oscillated between T1 and T2. In C++, the value of index could be changed using the tertiary operator: index = index? 0:1. Depending on the value of index, either apply hash function f_1 (index == 0) or f_2 (index == 1).

	Table T1	Table T2
[0]		
[1]		String matching
[2]	Matrix searching	
[3]	School of Computer Science	Algorithm Engineering
[4]	Some related problem	
[5]		Online algorithms
[6]	Emphasis on	
[7]		
[8]	Fullerton	
[9]		
[10]	Server Problem	Greedy pattern
[11]		California State University
[12]	Optimal tree construction	
[13]		
[14]		
[15]	College of Engineering and Computer Science	
[16]	Monge properties	