WASHINGTON STATE UNIVERSITY

School of Electrical and Computer Engineering
CptS223: Advanced Data Structures C/C++
Spring 2018
Homework #7
Due: Friday 03/30/2018 @ 11:59pm

Rules/In structions: This is an individual assignment and all work must be your own. Be sure to show your work when appropriate. This assignment must be turned in by the due date via Blackboard in *PDF* format. You may use any editor you like (or even print it out, *legibly* write in answers, and scan it in), but convert it to *PDF* for submission.

A maximum of 24 hours delay is permitted at the cost of 10% penalty. Beyond this 24-hour delay, not submissions will be graded. That is, assignments submitted after 24 hours after the above due date will not get any credits.

Let '*M*' denote the hash table size. Consider the following four different hash table implementations:

a. **Implementation (I)** uses *chaining*, and the hash function is $hash(x) = x \bmod M$.
b. **Implementation (II)** uses *open addressing by linear probing*, and the hash function is $h_i(x) = (hash(x) + f(i)) \bmod M$, where $hash(x) = x \bmod M$, and $f(i) = i$.
c. **Implementation (III)** uses *open addressing by quadratic probing*, and the hash function is $h_i(x) = (hash(x) + f(i)) \bmod M$, where $hash(x) = x \bmod M$ and $f(i) = i^2$.
d. **Implementation (IV)** uses *open addressing by double hashing*, and the main hash function is $h_i(x) = (hash(x) + f(i)) \bmod M$, where $hash(x) = x \bmod M$, and $f(i) = i \times hash_2(x)$, and $hash_2(x) = 15 - (x \bmod 7)$.

Starting with an initially empty hash table of size '*M*' for each of the above four implementations, INSERT the keys {42, 27, 24, 47, 37, 16, 3, 91, 79} (in this specified order). While doing these insertions assume that the table size '*M*' is kept fixed at '*11*' throughout (i.e., the code never calls the rehash function).

1. Show the final hash tables that result under each of the scenarios after all the insertions. For this, you can use the space provided in the following tables or you can draw your own version of the final hash tables.

	Chaining
0	
1	
2	79->24
3	91->3->47
4	37
5	16->27
6	
7	
8	
9	42
10	

	Linear Probing
0	
1	
2	24
3	47
4	37
5	27
6	16
7	3
8	91
9	42
10	79

	Quadratic Probing
0	79
1	91
2	24
3	47
4	37
5	27
6	16
7	3
8	
9	42
10	

	Double Hashing
0	91
1	
2	24
3	47
4	37
5	27
6	3
7	16
8	79
9	42
10	

2. For each implementation, for doing each insertion, count the number of comparisons that corresponding implementation needs to perform before you do that insertion. This number of comparisons is the cost associated with each insert operation. More the collision, more the number of comparisons. For open addressing schemes, this is the same as counting the number of failed probes. Record and show your answer in the table provided below.

	Number of comparisons incurred while inserting:									
Implementation ↓	**42**	**27**	**24**	**47**	**37**	**16**	**3**	**91**	**79**	**Total**
Chaining	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Linear Probing	**1**	**1**	**1**	**1**	**1**	**2**	**5**	**6**	**9**	**27**
Quadratic Probing	**1**	**1**	**1**	**1**	**1**	**2**	**3**	**4**	**4**	**18**
Double Hashing	**1**	**1**	**1**	**1**	**1**	**3**	**4**	**3**	**4**	**19**