Test 2 Takehome - Hashing

Due: November 11th

Honor Statement
By signing below I am confirming that I have neither given nor received assistance from anyone. I understand that if my answers come close to resembling that of another student and/or digital source, where this resemblance is based on the judgment of the instructor (and a conference with each participant), I can be given a score of zero.
Due
Your completed exam will not have ANY handwritten portions (except your signature) whatsoever and will not be graded if it does. Turn your completed take home exam by Monday, November 11, 2019 by 9 p.m. by uploading a digital copy to slack. You will also bring a printed copy to Dr. Griffin by 11:00 a.m. Tuesday. All work will be completed by 9 p.m. Monday night and the printed copy will not have any revisions from the uploaded copy.
Warning
This is a takehome, so make sure you explain or justify every answer (when appropriate).
For example Q1 has an answer of .1, but you may want to add the definition of (λ) to bolster your answer.

M Number	**Signature**
______________________	______________________

Q1

Suppose that an open-address hash table has a capacity of 1000 and it contains 100 elements. What is the table's load factor?

Q2

I plan to put 1000 items in a hash table, and I want the average number of accesses in a successful search to be about 3.0.

- About how big should the array be if I use open addressing with linear probing?
- About how big should the array be if I use chained hashing?

Q3

Define collision in a hash table.

Q4

If you're picking a table size for your hash table, which sizes should you stay away from?

Q5

What is generally a good choice for a load factor (λ) when using open addressing?

Q6

Is the choice for load factor (λ) the same when using open addressing?

Q7

Draw a hash table with open addressing and a size of 9. Use the hash function `k%9`. Insert the keys: 5, 29, 20, 0, 27 and 18 into your table (in that order).

Q8

What is the correct formula for the load factor?

Q9

Draw 3 separate hash tables that results from using a given hash function (below) to hash a given set of keys using the following collision resolution techniques:

1. linear probing
2. double hashing
3. chaining

Ascii Values	
A	65
Z	90
a	97
z	122

Hash Function	Description
$h1(k) = \left(\sum_{i=0}^{s.length()-1} Ascii(s[i]) \right) \% \ TableSize$	The sum of the ascii value of each letter mod table size
$h2(k) = (h1(k) * i) \ \% \ TableSize$	Where `i` is the `i`th collision `+ 1`

Strings to hash:

Epd, DcS, Fcf, Gco, Qzj, Wvc, RUC, ejJ, iwR, zyz

Q10

- **Given:** A hash table that uses linear probing to handle collisions.
- **Problem:** Describe a situation in which searching for a value in this hash table you might not find the value, even though it is in the table.

Q11

- **Problem:** A hash table is having trouble with clustering.
- **Describe:**
  - What is clustering?
  - What type of collision resolution does this typically occur with?
  - What is the generic solution to the clustering problem? Your answer shouldn't just be: "use new hash function". It should be discussed using general terms and definitions.

Q12

- **Given:** The formula for unsuccessful search in a hash table is: `O(1/1−λ)`
- **Describe:** The time for searches increases as the load factor increases. Explain why this happens as described in our study guide.

Q13

- **Problem:** What decides whether a hash function will perform well or not? In other words, how would you benchmark (***test***) a hash function.
- **Proposed Solution:** Discuss type of `input data`, proposed `load factor`, `collisions` and `collision resolution`.

Q14

- **Problem:** Given an example hash function, place it in 1 or more categories
- **Function Names:** Shift, Additive, Rotating ,XOR , or combination
- **Solution**: Look at the different types of Hash Functions and be able to categorize them.

Q15

- **Given:** Suppose you have table size 9 and when you have collisions you increment 3 every time.
- **Problem:** Describe why this is a problem, and how it effects the search space in your table.
- **Proposed Solution:** What could you do the alleviate the problems that occur because of the initial choices made? Describe the relationship between table size and the increment size.

Q16

Suppose you wish to make a hash table of all the students at MCSU University using as the key each student's ID number which is an 8 digit number. At MCSU, they give out student ID numbers consecutively. In other words, the first student to enroll gets ID 0000000, the second student gets the number 00000001, and so on. You map IDs to student records so that lookup (and insertion) are by the key of a student ID which is mapped to the value of a student record. In this problem we talk about how hashing the keys works.

You decide to implement the hash-table using an array/vector with 4999 elements, with **quadratic probing** to resolve conflicts. (By the way, both 4999 and 997 are prime numbers.) For each of the following three possible hash functions, say whether it is a good or bad choice, and explain why in one sentence. Note that digit_x refers to some integer representing one of the eight digits that comprise SIDs at MCSU U (so x takes on values 0 to 7).

1. $\texttt{digit}_0 \texttt{ + digit}_1 \texttt{ + digit}_2 \texttt{ + ... + digit}_7 \texttt{ mod 4999}$
2. `SID number mod 4999`
3. $\texttt{(digit}_0 \texttt{ + digit}_1 \texttt{ * 9971 + digit}_2 \texttt{ * 9972 + ... + digit}_7 \texttt{ * 9977) mod 4999}$

Q17

Given a collision resolution technique (array with linear probing, array with quadratic probing, array with double hashing, and array with chaining), which approach would be most appropriate for an application in which many keys that hash to the same value are likely to be present? Explain why.

Q18

Explain how deletion is performed in both **probing** and **chaining** hash tables.

Q19

A hash table needs to be resized if load factor of a table exceeds **0.7**. What are the important things to do when resizing a hash table?