You will submit your solution to this assignment to the Curator System (as `HW02`). Your solution must be either a plain text file (e.g., NotePad++) or a typed MS Word document; submissions in other formats will not be graded.

Partial credit will only be given if you show relevant work.

---

1. Suppose you are implementing a hash table and are trying to choose between using a probing strategy and using chaining (each slot is actually a linked list that can hold as many records as needed, prepended to the list).

You will use the same hash function and hash table size no matter which strategy you select.

a) **[10 points]** A *primary collision* occurs when two records map to the same home slot in the table. Assuming you insert the same set of records into the hash table, in the same order, how will the number of primary collisions be affected if you switch from quadratic probing to chaining? Justify your conclusion carefully.

Answer:
Since a primary collision occurs because the hash function and modding by table size yields the same slot for two keys, the collision resolution strategy does not even come into play. It makes no difference whatsoever whether quadratic probing or chaining is used.

b) **[10 points]** One objection to using chaining is that you'll have to perform a linear traversal of the linked list to find a record, and linear traversals are slow. Considering that, and assuming that a small but significant number of primary collisions will occur, would searching be more efficient if you chose quadratic probing instead of chaining? Justify your conclusion carefully.

Answer:
If you use chaining, when you walk the chain you will only see records that actually have the same home slot. If you use any form of probing, you will still see records that have the same home slot as the one you are looking for, but you may also see records that actually have other home slots. So, it is unlikely that probing will yield more efficient searches than chaining.

2. A hash table implementation uses quadratic probing to resolve collisions. Suppose the table was initially empty, and then records S1, S2, …, and SM were inserted into the table, in that order, all of those records have different key values, but all of those records map to the same home slot.

Then, N additional records R1, R2, …, and RN were inserted into the table, in that order, all of those records have different key values, and all of those records map to the different home slots from each other, none of which are the home slot of the Sk that were inserted earlier.

a) **[10 points]** What is the smallest number of record comparisons that could be performed in a search for the record RN. Justify your conclusion.

Answer:
The S-records will occupy M slots, scattered around in the table. When R1 is inserted, it cannot collide with S1 (different home slots), and it might not collide with any of the other S-records. In that case, we'd only examine 1 slot when inserting R1. The same logic applies to each of the other R-records, so the best case for RN is that we examine 1 slot.

b) **[10 points]** What is the largest number of record comparisons that could be performed in a search for the record RN. Justify your conclusion.

Answer:
The S-records will occupy M slots, scattered around in the table. When R1 is inserted, it cannot collide with S1 (different home slots), but it may very well collide with S2 or any of the other S-records when we probe to insert it. In fact, R1 could collide with each of the S-records, including S1, during probing. So, the worst case is that we could examine M + 1 slots when inserting R1.

Worse, when R2 is inserted, the same thing could happen, and R2 could then collide with R1, meaning that it could examine M + 2 slots when inserting R2.

Following the same pattern, we could examine as many as M + N slots when inserting RN.

3. Assume the same situation as in question 3, except that chaining is used to resolve collisions, with linked lists for the chains.

a) [10 points] What is the smallest number of record comparisons that could be performed in a search for the record RN. Justify your conclusion.

Answer:
If chaining is used, every record is stored in a chain in its home slot. So, RN cannot have collided with S1 since they have different home slots, and RN cannot collide with any other S-record, since they are all in the same slot as S1. And, RN cannot collide with any other R-record, since none of the R-records have the same home slot. Therefore, a search for RN will examine 1 slot.

b) [10 points] What is the largest number of record comparisons that could be performed in a search for the record RN. Justify your conclusion.

Answer:
Same as for the previous question... 1 slot.

4. Consider a hash table consisting of $M = 13$ slots, and suppose nonnegative integer key values are hashed into the table using the hash function `h2()`:

```
uint32_t h2(uint32_t key) {

   uint32_t prime = 0x811C9DC5;
   uint32_t hashvalue = 0;
   uint32_t digit;

   while ( key > 0) {
      digit = key % 10;
      hashvalue *= prime;
      hashvalue ^= ( digit + '0');
      key = key / 10;
   }
   return hashvalue;
}
```

You are strongly advised to write C code to use the function above to hash the key values given below.

I computed the home slots as follows:

Key	Hash value	Slot
146:	1987658771	7
152:	3698507770	9
215:	2340152298	3
303:	3308709048	8
418:	3038450617	7
116:	4228405450	9
149:	2648651900	11
352:	3698507768	7
232:	3586592031	9
182:	1308540155	0

a) **[20 points]** Suppose that collisions are resolved by using linear probing. The integer key values listed below are to be inserted, in the order given. For each key, show the home slot (the slot to which the key hashes, before any probing), the sequence of slots that are examined (including the home slot and any slots examined if probing is necessary), and the final contents of the hash table after the given key values have been inserted in the table:

Key Value	Home Slot	Slots examined during insertion
146	7	7
152	9	9
215	3	3
303	8	8
418	7	7, 8, 9, 10
116	9	9, 10, 11
149	11	11,12
352	7	7, 8, 9, 10, 11, 12, 0
232	9	9, 10, 11, 12, 0, 1
182	0	0, 1, 2

Total number of slots examined: 29

Final Hash Table:

Slot	0	1	2	3	4	5	6	7	8	9	10	11	12
Contents	352	232	182	215				146	303	152	418	116	149

b) **[20 points]** Repeat the previous question, but now resolve collisions using quadratic probing, with the probe function:

$$\frac{k^2 + k}{2}$$

Key Value	**Home Slot**	**Slots examined during insertion**
146	7	7
152	9	9
215	3	3
303	8	8
418	7	7, 8, 10
116	9	9, 10, 12
149	11	11
352	7	7, 8, 10, 0
232	9	9, 10, 12, 2
182	0	0, 1

Total number of slots examined: 21

Final Hash Table:

Slot	0	1	2	3	4	5	6	7	8	9	10	11	12
Contents	352	182	232	215				146	303	152	418	149	116