BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CSE208 (Data Structures and Algorithms II Sessional)
July 2021 Term

Offline: Hash Table
Deadline: 19/01/2022, 6:00 AM

---

1 Problem Specification

In this assignment, you have to implement a ***Hash Table*** with the following requirements.

1. You have to implement insert, search, and delete operations on the hash table. The length N of the hash table will be given as an input.
2. Data will be kept as a `(key, value)` pair in the hash table. You need to insert randomly generated 7-character-long unique words into the hash table (these words can be meaningful or meaningless). Therefore, you have to implement a random word generator. The words will be used as the "key", and the order of the incoming words will be used as the "value" (please see the example below). If your word generator produces duplicate words, you must keep only one instance of each.

2 Example

Suppose your word generator has generated the following 5 words.

ancient
puzzled
benefit
ancient
zigzags

The corresponding `(key, value)` pairs will be:

(ancient, 1)
(puzzled, 2)
(benefit, 3)
(zigzags, 4)

Note that the 2nd instance of the word "ancient" has been discarded.

3 Hash Function

Use two hash functions ($Hash1(k)$ and $Hash2(k)$) of your own, or from any good literature where you must try to avoid collisions as much as possible. We expect that 60% of the keys will have unique hash values (i.e., at least 60 unique hash values for 100 keys).

4 Collision Resolution

You need to implement the following three collision resolution methods.

1. **Separate Chaining**
   Place all the elements that hash to the same slot into a linked list. Slot j contains a pointer to the head of the list of all stored elements that hash to j ; if there are no such elements, slot j contains $NULL$.

2. **Double Hashing**
   Use the following hash function.

   $$doubleHash(k, i) = (Hash(k) + i \times auxHash(k)) \ \% \ N$$

   Here, $Hash(k)$ is one of the hash functions described in Section 3. Note that you have to use both the Hash functions mentioned in Section 3 for report generation (see Section 5 for more details). You can use a simple hash function as the auxiliary hash function $auxHash(k)$. The initial probe goes to position $Table[Hash(k)]$, and successive probe positions are offset from previous positions by an amount $auxHash(k)$, modulo N.

3. **Custom Probing**
   For the custom probing, use a hash function of the following form:

   $$customHash(k, i) = (Hash(k) + C_1 \times i \times auxHash(k) + C_2 \times i^2) \ \% \ N$$

   Here C_1 and C_2 are two auxiliary constants of your choice. The other details are same as the Double Hashing.

5 Report Generation

Generate 10000 seven-character-long unique words and insert them into the Hash Table. Using both hash functions ($Hash1(k)$ and $Hash2(k)$), list the number of collisions in a tabular format. Among these 10000 generated words, randomly select 1000 words and search each of these selected words in the hash table. Report the average number of probes (i.e., the number of times you access the hash table) required to search these words. These results should be reported for both hash functions. The report should be generated in the following tabular format. You also need to report the hash functions and the auxiliary hash functions that you have used.

Table 1: Performance of various techniques for collision resolution with two different hash functions.

	Hash1		Hash2	
	Number of collisions	Average probes	Number of collisions	Average probes
Chaining Method				
Double Hashing				
Custom Probing				