MIT Art Design and Technology University

MIT School of Computing, Pune

Department of Computer Science and Engineering

Second Year B. Tech

Academic Year 2022-2023. (SEM-II)

Subject: Advance Data Structures Laboratory

Assignment 8

Assignment Title: Create a hash table and handle the collisions using linear probing with or without replacement.

Aim: Implement hashing and linear probing with or without replacement to handle the collisions.

Prerequisite:

1. Basic concepts of hashing.
2. Linear probing with or without replacement to handle the collisions.

Objectives:

Implement a Program in python for the following operations on Graph:

1. Create a hash table using any data structure.
2. Recognize and define the basic attributes of a hashing.
3. Collision handing in hashing using linear probing with or without replacement.

Outcomes:

Upon Completion of the assignment the students will be able to

1. Create hash table using any data structure.

2. Understand and analyse Linear probing with or without replacement to handle the collisions.

Theory:

Hashing is a method of directly computing the address of the record through key by using a suitable mathematical function called the hash function.

Hashing is a technique or process of mapping keys, and values into the hash table by using a hash function. It is done for faster access to elements. The efficiency of mapping depends on the efficiency of the hash function used.

It uses hash tables to store the data in an array format. Each value in the array has been assigned a unique index number.

Let a hash function H(x) maps the value x at the index x%10 in an Array.

For example, if the list of values is [11,12,13,14,15].

It will be stored at positions {1,2,3,4,5} in the array or Hash table respectively.

Collision

When the two different values have the same value, then the problem occurs between the two values, known as a collision. To resolve collisions, we have some techniques known as collision techniques. Linear Probing is one of the collision handling techniques.

Linear Probing

The simplest approach to resolve a collision is linear probing. In this technique, if a value is already stored at a location generated by h(k), it means collision occurred then we do a sequential search to find the empty location.

Here the idea is to place a value in the next available position. Because in this approach searches are performed sequentially so it's known as linear probing.

Here array or hash table is considered circular because when the last slot reached an empty location not found then the search proceeds to the first location of the array.

Clustering is a major drawback of linear probing.

Below is a hash function that calculates the next location. If the location is empty then store value otherwise finds the next location.

Following hash function is used to resolve the collision in:

$h(k, i) = [h(k) + i] \bmod m$

Where

m = size of the hash table,

h(k) = (k mod m),

i = the probe number that varies from 0 to m–1.

Therefore, for a given key k, the first location is generated by [h(k) + 0] mod m, the first time i=0.

If the location is free, the value is stored at this location. If value successfully stores then probe count is 1 means location is founded on the first go.

If location is not free then second probe generates the address of the location given by [h(k) + 1]mod m.

Similarly, if the generated location is occupied, then subsequent probes generate the address as [h(k) + 2]mod m, [h(k) + 3]mod m, [h(k) + 4]mod m, [h(k) + 5]mod m, and so on, until a free location is found.

Probes is a count to find the free location for each value to store in the hash table.

Linear Probing without replacement:

- A hash table in which a collision is resolved by putting the item in the next empty place is called linear probing.
- This strategy looks for the next free location until it is found.
- Function - **(Hash(x) + i) MOD M**

  Where i = 1,2,3,4….till empty location is found

Linear Probing – with replacement

- Address index is already occupied by the key?
  - There are two possibilities –
    - Either it is home address (collision)
    - Or not key's home address
- If the existing key's actual address is different, then the NEW KEY having the address of that slot is placed at that position; and the key with other address is placed in the next empty position.

Conclusion:

We have created hash table and collision handling method, linear probing with or without replacement.

Frequently Asked Questions:

1. Insert the following sequence of keys in the hash table
   {9, 7, 11, 13, 12, 8}
   Use linear probing without replacement technique for collision resolution
   $h(k, i) = [h(k) + i] \bmod m$
   $h(k) = 2k + 5$
   $m=10$
2. List out applications of linear probing.
3. Store the following data into a hash table of size 10 and bucket size 1. Use linear probing with replacement for collision resolution. The Hashing function is key % 10
   12, 01, 04, 03, 07, 08, 10, 02, 05, 14

MIT SCHOOL OF ENGINEERING
Rajbaug, Loni-Kalbhor, Pune

MIT-ADT UNIVERSITY
PUNE, INDIA

Name : Sourav Shailesh Toshniwal
Class : SY - 1
Roll no. : 2213047
Subject: ADSL

Assignment - 8.

1. Insert the following sequence of keys in the hash table {9, 7, 11, 13, 12, 8} Use linear probing without replacement for collision resolution. $h(k,i) = [h(k)+i] \% m$.
$h(k) = 2k+5$
$m = 10$.

Ans For given sequence we have:
$h(9) = (2(9)+5) \% 10 = 3$
$h(7) = (2(7)+5) \% 10 = 9$
$h(11) = (2(11)+5) \% 10 = 7$
$h(13) = (2(13)+5) \% 10 = 1$
$h(12) = (2(12)+5) \% 10 = 9$
$h(8) = (2(8)+5) \% 10 = 1$.
Now, insert them into table.

12	13	8	9				11		7

2. List out applications of linear probing.

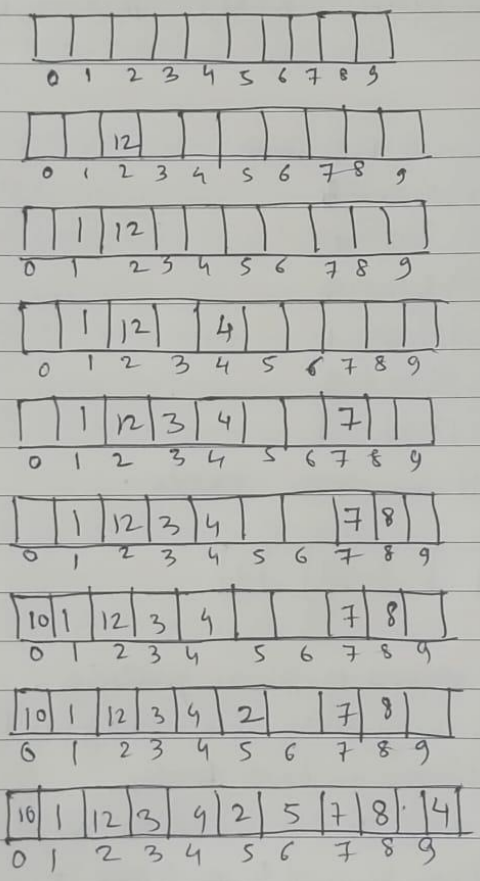
Ans 1) Hash Table : Linear probing is a widely used technique for resolving collisions in hash tables.
2) Cache : Linear probing can be used to implement a LRU
3) Symbol table : It can be used to implement symbol table which stores keys & values.

4) Spell checkers : Linear probing can be used to store and search words in a dictionary.

5) Database indexing: Linear probing can be used to implement indexing in databases for efficient data retrieval.

3) Store the following data into into a hash table of size 10 and bucket size 1. Use linear probing with replacement for collision resolution. The hashing function is key % 10.
12, 01, 04, 03, 07, 08, 10, 02, 05, 14.

Ans



Code:

```
list1=[]
j=1
y=int(input("Enter the array size:"))


for i in range(y):
    list1.append(0)
print("Initial array is:",list1)


for k in range (y):
    x=int(input("Enter the value:"))
    h=x%y
    z=h
    if list1[h]==0:
        list1[h]=x
        print(list1)
    else:
        while (list1[h]!=0):
            h=(h+j)%y
        if list1[z]%y==z:
            list1[h]=x
        else:
            list1[h]=list1[z]
            list1[z]=x
        print(list1)
```

Output:

```
PS C:\SOURAV\CODE\C++ language codes\ADS assignment> python -u "c:\SOURAV\CODE\C++
Enter the array size:5
Initial array is: [0, 0, 0, 0, 0]
Enter the value:1
[0, 1, 0, 0, 0]
Enter the value:2
[0, 1, 2, 0, 0]
Enter the value:3
[0, 1, 2, 3, 0]
Enter the value:4
[0, 1, 2, 3, 4]
Enter the value:5
[5, 1, 2, 3, 4]
PS C:\SOURAV\CODE\C++ language codes\ADS assignment>
```