SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING-IOT Including CS&BCT

COURSE NAME : **19CS307**- DATA STRUCTURES

II YEAR / III SEMESTER

Unit V- **SORTING AND SEARCHING**

Topic :**Rehashing**

Rehashing

- **Rehashing means hashing again**
- The size of the array is increased (doubled) and all the values are hashed again and stored in the new double sized array to maintain a low load factor and low complexity.
- **Rehashing**: Try H_1, H_2, …, H_m in sequence if collision occurs. Here H_i is a hash function.
- **Double hashing** is one of the best methods for dealing with collisions.
  - If the slot is full, then a second hash function is calculated and combined with the first hash function.
  - $H(k, i) = (H_1(k) + i\, H_2(k)) \% m$

Cont..

- The probing sequence will be:
- index = (index + 1 * indexH) % hashTableSize;
  index = (index + 2 * indexH) % hashTableSize;

Implementation of hash table with double hashing

- **initialization**
- string hashTable[21];
- int hashTableSize = 21;
- **Insert**
- void insert(string s)
- {
- //Compute the index using the hash function1
- int index = hashFunc1(s);
- int indexH = hashFunc2(s);
- //Search for an unused slot and if the index exceeds the hashTableSize roll back
- while(hashTable[index] != "")
- index = (index + indexH) % hashTableSize;
- hashTable[index] = s; }

Search

- void search(string s)
- {
- //Compute the index using the hash function int index = hashFunc1(s);
- int indexH = hashFunc2(s);
- //Search for an unused slot and if the index exceeds the hashTableSize roll back
- while(hashTable[index] != s and hashTable[index] != "")
- index = (index + indexH) % hashTableSize;

Cont..

- //Is the element present in the hash table if(hashTable[index] == s)
- cout << s << " is found!" << endl;
- else
- cout << s << " is not found!" << endl; }

Application

- *Associative arrays*: Hash tables are commonly used to implement many types of in-memory tables
- *Object representation*: Several dynamic languages, such as Perl, Python, JavaScript, and Ruby use hash tables to implement objects.
- Hash Functions are used in various algorithms to make their computing faster

Thank you