## **LEC-13 Hashing and Hash Table**

**Hashing:**

* Hashing is a technique used to map data of arbitrary size to fixed-size values, typically integers.
* It is widely used for data storage, retrieval, and searching in various applications.
* Hashing reduces the search space and provides efficient data access.

**Hash Function:**

* A hash function is a mathematical algorithm that takes an input (or 'key') and produces a fixed-size hash value.
* Ideally, a good hash function distributes keys uniformly across the hash table to minimize collisions (two keys hashing to the same value).

**Hash Table:**

* A hash table is a data structure that uses a hash function to map keys to values, enabling efficient data retrieval.
* It consists of an array of buckets, where each bucket can hold multiple key-value pairs.
* Hashing allows for constant-time average-case access, making it suitable for applications like dictionaries, databases, and caches.

**Collisions:**

* Collisions occur when two different keys hash to the same value, causing data to be stored in the same bucket.
* Handling collisions is a critical aspect of hash table design.
* Common collision resolution techniques include chaining (using linked lists) and open addressing (probing neighboring buckets).

## **Example:**

Suppose we have a hash table to store student records using their IDs.

**Hash Function:**

A simple hash function could be to take the remainder of the ID when divided by the size of the hash table.

**🡪Purpose of Hashing :**

To do insertion and searching or retrieval constant time -0(1)

**🡪Some terminologies related to Hashing:**

1. Key(s) / search key(s)
2. Hash function
3. Hash Table

e.g. n= 6

|  |
| --- |
| 5 |
|  |
| 23 |
|  |
|  |
| 20 |

Hash key

slots

Hash Table

Mostly we use following Hash functions

H(k) = (k model 10, k mod n etc)

Lets take:

H(k) = (k mod 10)

Keys/Data = 23,20,5

H(23)=23 mod 10

H(23)= 3

Similarly :

H(20) = 0,H(5)=5

Issue:

Key= 3

|  |
| --- |
| H(3)=3 |

We have already value at index 3. This is called collision.

Solution to

collision:

Chaining Method Open Addressing

(Opened hashing) (closed hashing)

AVL Linked List Linear Probing Quadratic Probing Double hashing

Lets correct the previous collision using ‘Chaining Method’ :

n =s

Hash function- H(k) = k mod n

Keys / Data =23,20,5,3,13

H(23)=3, H(20)=0 H(5)=0, H(3)=3, H(13)=3

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

|  |
| --- |
|  |
| . |
|  |
|  |
| . |

23 🡪3🡪13

20🡪5