

## Assignment - 6

Title: The Dictionary ADT.

Problem: Implement all the functions of a dictionary using hashing. Data: set of (key value) pairs, keys are mapped to values, keys must be unique standard operations: Insert (key, value), find (key). Delete (key)

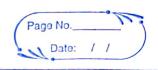
Objective: To understand the implementation of all the functions of a dictionary and et andard operations on Dictionary

Outcome: At the end of their this assignment students will be able to perform standard operations on Dictionary ADT.

The Dictionary ADT: A dictionary is as ordered or unordered list of key-element pairs where keys are used to locate elements is the list

Dictionary is a data structure, which is generally an association of wrighe keys with some values one may friend a value to a key delete a key (and naturally an associated value) and took up for a value by the key. Values are not required to be wrighe.

A Dictionary can be implemented in various



ways: using a list, binary search tree, hash table etc.

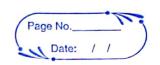
\* Hashing Hashing is a technique to convert a vange of key values into a range of indexes of an array.

In hashing, large keys are converted into small keys by using hash functions. The values are then stored in a data structure called hash table. The idea of hashing is to distribute entries uniformly across as array. Fach element is assigned a key. By using that key you can access the element in O(1) time.

Hash function: A hash function is any function that can be used to may a data set of an arbitrary size to a data set of a fixed size which falls into the hash table. The values returned by a hash values, hash code, hash sum or simply hashes.

\* Hash Table: A hash table is a data structure
that is used to store keys/value pairs. It uses
a hash function to compute an index into an
array in which an element will be inserted
or searched;

Collision Handling: - Since a hash function gets us a small number for a big key, there is possibility that two keys result in same value



The situation where a newly inserted key maps to an already occupied slot is hash table is called collision.

Following are the ways to handle collision:

1) Chaining: The idea is to make each cell of hash table point to a linked list of records that have same pash functions value.

Chaining is simple, but requires additional memory outside the table.

Den Addressing: In open addressing, all clements are stored in the hash table it self. Each table entry contains either a record or NIL. when searching for an element, we one by one examine table slots until the desired element is found or it is clear that the element is not in the table.

Implementation of Hash Table:

Consider a dictionary, where keys are integers in the range to, N-II. Then an array of size N can be used to represent the dictionary Cach entry in this array is thought of as a "brucket": An element c' with key k' is inserted in ACKI. Bucket entries associated with keys not present in dictionary contains a Special NO-SUCH-KEY object. If the dictionary contains contains elements with the same key then two or more different elements may be mapped to the same bucket of A. In this case



we say that a collision between these elements has occured. One easy way to deal with collisions is to allow a sequence of elements with the same key, k, to be stored in ACkI.

Assuming that an arbitrary element with key k satisfies queries find i tem(k) and remove item(k), these operations are now performed in O(1) time, while insert item(k) needs only to find where on the existing list & n(h) to insert the new item, e. The drawback of this is that the size of the bucket array is the size of the set from which key are drawn, which may be huge. Algorithman Winter 2 to 12 aldor winter Mash Node dass Declarations: class Hash Node 12 To A of to contil med int key in a serior el Mash Node (int key int value) ? this -> key: key; this -> value = value; this -> next = NULL;



```
Insertions:
2 void insert (int key, string value)
    int k = hey-hash (key);

if (arr(k)!=1) {

int t = travense (k);
 arr[t] -> hey = key:

arr[t] -> meaning = value; 3 +1
else &
 void remove (int key)?
```



Tatlana	Dunat -1	1 tomas	0 14
Testlase	Expected	Outcome	Result
Insert -21, 24, 30	0-30	As	Pars
09,04,14,28,18,	<u>1-2 </u> 2-18	expected	
[5,12	3-12		
	4-24		
	5-4	May	
	6-14 7-15		
	8-28 9-9		
 ,	9-9		*
 Conclusion: A	fter success	fully complet	ing this
 assignment	we have	learned imp	dementation
 Stard standa	rd operati	ions on Dict	ing this dementation g and various nonary ADT
	make the side	a (	
		, , , , , , , , , , , , , , , , , , , ,	
	- La-		
 /			
13.40		-	
	,		
	.,		4