# Assignment-7

Title: To write a program for implementation of symbol table and perform various operations

Problem: The symbol table is generated by compiler. It is a set of name-attribute pairs Perform the following operations on symbol table.

1) Determine if the particular name is in the table.
2) Retrieve the attribute of that name.
3) Insert new name and its attribute.
4) Delete a name and its attribute.

Objective:

1) To understand concept of symbol table
2) Why symbol table is needed.

Outcome:

1) Use of symbol table.
2) Various methods of implementing symbol table

Concept Related Theory:

A symbol table is a data structure used by a language translator such as compiler or interpreter where each identifier in a program's source code is associated with information related to its declaration or appearance in the source.

Scanned with CamScanner

A symbol table may only ~~ra~~ exist during the translation process or it may be embedded in the output of that process. Such as in an ABI object file for later interpretation.
Symbol table is used to store information relatedto to various entities like as function name, variable name, object, classes etc.
Symbol table is simply a table which can be either linear or hash table. It maintains an entry for each name in the format
<symbol name, type attribute>

$$\begin{tikzcd}
 & \text{Symbol table} \arrow[dl] \arrow[dr] & \\
\text{Static Symbol table} & & \text{Dynamic Symbol Table}
\end{tikzcd}$$

* Implementation of Symbol Table.
• Unordered Array implementation.
• Ordered Array implementation
• Unordered or ordered list
• BST
• Balanced BST
• Hashing.

Algorithms :

* Linear probing without replacement with chaining.
1) Start and make all the chain -1.
2) Find the respective bucket for the given key.
  2.1) If the bucket is empty, then insert

Scanned with CamScanner

Page No.
Date: / /

the key-value pair at that location.
2.2) Else insert the key-value pair at next empty location and make the value of chain of previous location with same key equal to the new location.
3) Fill all the buckets ~~use~~ in the same manner and make sure the chain gives the location of the next value with same key.
4) End [Time complexity = O(1)].

* Linear Probing with replacement with chaining
1) Start
2) Make all the chains of the buckets equal to -1
3) Find the bucket for the given key.
3.1) If the bucket is empty, then inserted the key-value pair at that location
3.2) Else replace the element store in that bucket.
3.2.1) If the element has different key.
3.2.2) If the element has same key then store the new ~~ky~~ key value pair at next vacant location.
4) The chain of ~~th~~ bucket will give you the of next value with same key.
5) End (Time complexity = O(1))

Scanned with CamScanner

Page No.________ Date: / /

Test Cases :

| Description | Expected | Actual | Result |
|---|---|---|---|
| 1) Insert - 25, 35, 36, 55, 57 78, 99, 89, 74 | 0 - 78 - (-1)<br>1 - 99 - (2)<br>2 - 89 - (-1)<br>3 -<br>4 - 74 - (-1)<br>5 - 25 - (6)<br>6 - 35 - (8)<br>7 - 36 - (-1)<br>8 - 55 - (-1)<br>9 - 57 - (-1) | Same as expected | Pass. |
| 2) Insert - 25, 35, 36, 55, 57 78, 99, 89, 74 | 0 - 55<br>1 - 35 (0)<br>2 - 89<br>3 -<br>4 -<br>5 - 25 (1)<br>6 - 35<br>7 - 57<br>8 - 78<br>9 - 99 (2) | Same as expected | Pass |

Conclusion :- We were able to implement the symbol table, with chaining successfully.