Symbol Tables

Akshay Singhal Compiler Design

Contents-

What is symbol table?

1. Symbol table is a data structure used by compiler to keep track of semantics of variable.
2. Symbol table is built in lexical and syntax analysis phases.
3. The symbol table is used by various phases. For example-

- Semantic analysis phase refers symbol table for type conflict issue.
- Code generation refers symbol table knowing how much run-time space is called. What type of run-time space is allocated?

Symbol Table Entries-

The items to be stored in symbol table are-

1. Variable names
2. Constants
3. Procedure names
4. Function names
5. Literal constants and strings
6. Compiler generated temporaries
7. Labels in source languages

Compiler uses following types of information from symbol table-

1. Data type
2. Name
3. Declaring procedures
4. Offset in storage
5. If structure or record then pointer to structure table
6. For parameters, whether parameter passing is by value or reference ?
7. Number and type of arguments passed to the function
8. Base address

Requirements for Symbol Table Management-

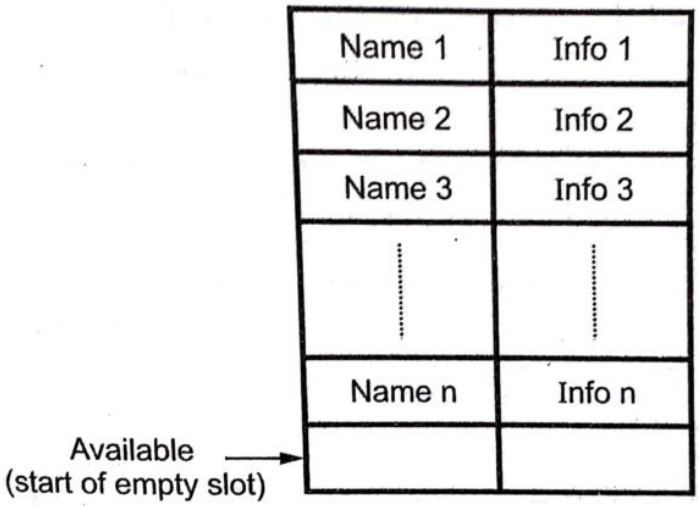
- For quick insertion of identifier and related information
- For quick searching of identifier

Data Structures for Symbol Tables-

Following are commonly used data structures for symbol table construction-

1. List data structure for symbol table-

- Linear list is a simplest kind of mechanism to implement the symbol table.
- In this method, an array is used to store names and associated information.
- New names can be added in the order as they arrive.
- The pointer ‘available’ is maintained at the end of all stored records.
- Following figure shows list data structure using arrays-

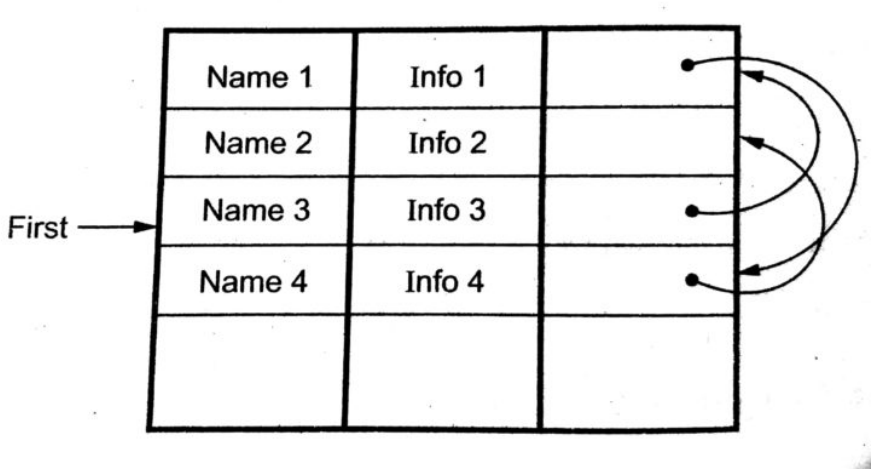


- To retrieve the information about some name we start from beginning of array and go on searching up to available pointer. If we reach at pointer available without finding a name, we get an error “use of undeclared variable”.
- While inserting a name, we should ensure that it should not be already there. If it is there, another error occurs i.e. “Multiple defined Name”.

- The advantage of list organization is that it takes minimum amount of space.

2. Self organizing list-

- This symbol table implementation is using linked list. A link field is added to each record.
- We search the records in the order pointed by the link of link field.
- A pointer “First” is maintained to point to first record of the symbol table.

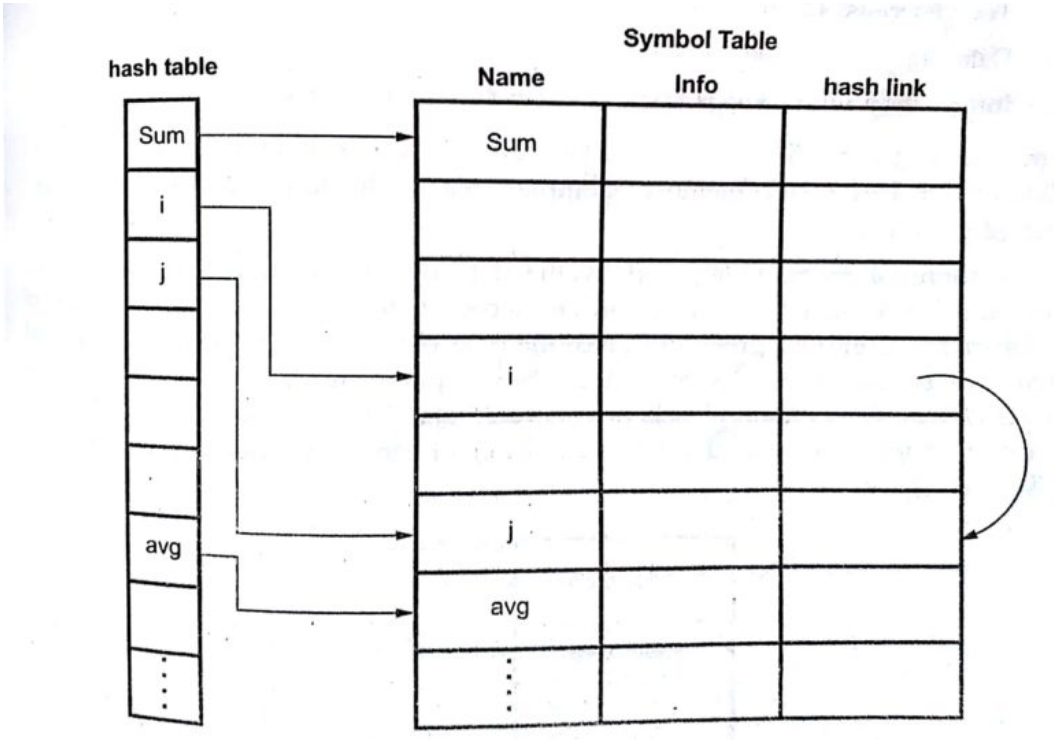


The reference to these names can be Name 3, Name 1, Name 4, Name 2.

- When the name is referenced or created, it is moved to the front of the list.
- The most frequently referred names will tend to be front of the list. Hence, access time to most frequently referred names will be the least.

3. Hash Tables-

- Hashing is an important technique used to search the records of symbol table.
- In hashing scheme, two tables are maintained- a hash table and a symbol table.
- The hash table consist of k entries from 0, 1 to k-1. These entries are basically pointers to symbol table pointing to the names of symbol table.
- To determine whether the 'Name' is in symbol table, we use a hash function 'h' such that h(name) will result any integer between 0 to k-1. We can search any name by position = h(name). Using this position, we can obtain the exact locations of name in symbol table.



- This method is superior to list organization.

To get more videos, notes and other study material of Compiler Design, click here.

To practice more Previous Years GATE Problems from Compiler Design, click here.

To watch more videos, visit our YouTube channel LearnVidFun.