# Compiler Design Lab

Symbol Table

Course Code: CSE332

# Variable and Memory

```
int a,b,c;
a = 10, b = 20, c = 30;
```

Variable inside memory (RAM)

| | | | |
|---|---|---|---|
| VALUE | 10 | 20 | 30 |
| MEMORY ADDRESS | 000000000000012AB | 000000000000012CD | 000000000000012EF |

Symbol Table

| Variable Name | Memory Address |
|---|---|
| a | 000000000000012AB |
| b | 000000000000012CD |
| c | 000000000000012EF |

# Pointer Variable

```
int a = 10;
int *p;
p = &a;
```

## Variable inside memory (RAM)

| | | | |
|---|---|---|---|
| VALUE | 10 | 0000000000001234 | |
| MEMORY ADDRESS | 0000000000001234 | 0000000000005678 | |

## Symbol Table

| Variable Name | Memory Address |
|---|---|
| a | 0000000000001234 |
| p | 0000000000005678 |

# Implementation of Symbol Table

AIM:  To write a C program to implement a symbol table.

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

Sample Input: a=b+c+3

Sample Output:

| Symbol | Memory Address | Type |
|---|---|---|
| a | 0000000000001234 | Identifier |
| = |  | Operator |
| b | 00000000007614E0 | Identifier |
| + |  | Operator |
| c | 0000000000761470 | Identifier |
| 3 |  | Integer |

# Algorithm

- Get the input from the user
- Iterate through each character of the input and store it into a character variable 'c'
- if ‘c’ is an alpha character
  - allocate memory for c
  - add the memory address of 'c' to the address table
  - add the value of 'c' to the values table
  - print the address, value and type of 'c'
- else
  - print the value and type of 'c'
- Get an input 's' from the user
- Search 's' in the address table
- If s found in the address table
  - print the address of s
- else
  - print "Symbol Not Found"