Name: AYUSH KUSHWAHA

Reg No.: CH.EN.U4CSE22077

Date: 08-09-2025

-----------------------------------------------------------------------------------  
  **LAB – 6(Symbol Table)**  
   
**Aim:** To implement Symbol table using C program.

Algorithm:

* Start the Program.
* Get the input from the user with the terminating symbol ‘$’.
* Allocate memory for the variable by dynamic memory allocation function.
* If the next character of the symbol is an operator then only the memory is allocated.
* While reading, the input symbol is inserted into symbol table along with its memory address.
* The steps are repeated till “$” is reached.
* To reach a variable, enter the variable to the searched and symbol table has been generated.
* Checked for corresponding variable, the variable along its address is displayed as result.
* Stop the program

**Code:**  
   
**SymbolTable.c**  
   
#include <stdio.h>

#include <ctype.h>

#include <stdlib.h>

int main() {

int i = 0, n = 0;

char c;

char expression[100]; // buffer for input expression

printf("Input the expression ending with $ sign: ");

while ((c = getchar()) != '$' && i < 99) {

expression[i++] = c;

}

n = i; // number of characters read

printf("\nGiven Expression: ");

for (i = 0; i < n; i++) {

printf("%c", expression[i]);

}

printf("\n\nSymbol Table display\n");

printf("Symbol \t Address \t Type\n");

for (i = 0; i < n; i++) {

c = expression[i];

if (isalpha(c)) {

// Identifier

printf("%c \t %p \t identifier\n", c, (void\*)&expression[i]);

} else if (c == '+' || c == '-' || c == '\*' || c == '=') {

// Operator

printf("%c \t %p \t operator\n", c, (void\*)&expression[i]);

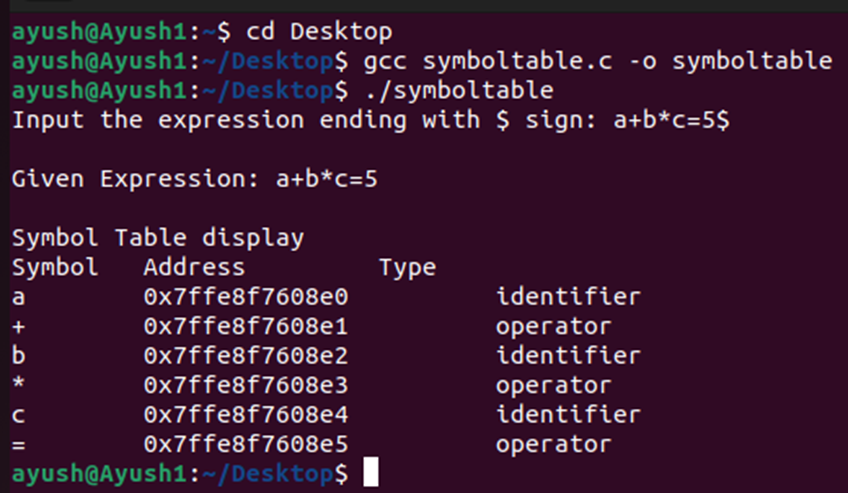
}

// You can add more checks here (digits, delimiters, etc.)

}

return 0;

}

**Output:**  


Result: Hence, symbol table using C programming has been successfully executed.