Exp. No. 18
Write a C program to implement the back end of the compiler.
Program:

```
#include <stdio.h>
#include <string.h>

int main() {
   int n, i;
   char a[50][10]; // Increased size to prevent overflow

   // Input number of intermediate codes
   printf("Enter the number of intermediate codes: ");
   scanf("%d", &n);
   getchar(); // Consume the newline left by scanf

   // Input 3-address code lines
   for (i = 0; i < n; i++) {
      printf("Enter 3-address code %d: ", i + 1);
      fgets(a[i], sizeof(a[i]), stdin); // Read the entire line, including
spaces
      a[i][strcspn(a[i], "\n")] = '\0'; // Remove newline character
   }

   // Generate assembly-like code
   printf("\nGenerated Assembly Code:");
   for (i = 0; i < n; i++) {
      printf("\n MOV %c, R%d", a[i][3], i); // Move first operand to
register

      // Check and generate appropriate operation
      if (a[i][4] == '-') {
         printf("\n SUB %c, R%d", a[i][5], i);
      } else if (a[i][4] == '+') {
         printf("\n ADD %c, R%d", a[i][5], i);
      } else if (a[i][4] == '*') {
         printf("\n MUL %c, R%d", a[i][5], i);
      } else if (a[i][4] == '/') {
         printf("\n DIV %c, R%d", a[i][5], i);
      }
```

```
        // Move result back to the target variable
        printf("\n MOV R%d, %c", i, a[i][1]);
    }

    printf("\n");
    return 0;
}
```

```
Output
Enter the number of intermediate codes: 2
Enter 3-address code 1: a=b+c
Enter 3-address code 2: d=n*d

Generated Assembly Code:
 MOV +, R0
 MOV R0, =
 MOV *, R1
 MOV R1, =


=== Code Execution Successful ===
```