15. Write a C program to implement the back end of the compiler.

```
#include <stdio.h>
#include <string.h>
struct TAC {
   char lhs[10], op1[10], op[5], op2[10];
} code[20];
int main() {
   int n, i, j;
   printf("Enter number of expressions: ");
   scanf("%d", &n);
   for (i = 0; i < n; i++) {
      printf("Enter expression %d (a = b + c): ", i + 1);
      scanf("%s = %s %s %s", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
   }

   printf("\nOptimized Assembly Code:\n");
   for (i = 0; i < n; i++) {
      int found = 0;
      for (j = 0; j < i; j++) {
         if (strcmp(code[i].op, code[j].op) == 0 &&
            strcmp(code[i].op1, code[j].op1) == 0 &&
            strcmp(code[i].op2, code[j].op2) == 0) {
            printf("MOV %s, %s\n", code[i].lhs, code[j].lhs);
            found = 1;
            break;
         }
      }
      if (!found) {
```

```
        if (strcmp(code[i].op, "+") == 0)
            printf("ADD %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "-") == 0)
            printf("SUB %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "*") == 0)
            printf("MUL %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "/") == 0)
            printf("DIV %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
    }
  }
  return 0;
}
```

Output:

```
printf("Enter number of expressions: ");
scanf("%d", &n);
for (i = 0; i < n; i++) {
    printf("Enter expression %d (a = b + c): ", i + 1);
    scanf("%s = %s %s %s", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
}

printf("\nOptimized Assembly Code:\n");
for (i = 0; i < n; i++) {
    int found = 0;
    for (j = 0; j < i; j++) {
        if (strcmp(code[i].op, code[j].op) == 0 &&
            strcmp(code[i].op1, code[j].op1) == 0 &&
            strcmp(code[i].op2, code[j].op2) == 0) {
            printf("MOV %s, %s\n", code[i].lhs, code[j].lhs);
            found = 1;
            break;
        }
    }
    if (!found) {
        if (strcmp(code[i].op, "+") == 0)
            printf("ADD %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "-") == 0)
            printf("SUB %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "*") == 0)
            printf("MUL %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
        else if (strcmp(code[i].op, "/") == 0)
            printf("DIV %s, %s, %s\n", code[i].lhs, code[i].op1, code[i].op2);
    }
}
return 0;
```

```
C:\Users\ROJAYADAV\OneDri
Enter number of expressions: 4
Enter expression 1 (a = b + c): a = b + c
Enter expression 2 (a = b + c): t1 = b + c
Enter expression 3 (a = b + c): t2 = t1 * d
Enter expression 4 (a = b + c): t3 = b + c

Optimized Assembly Code:
ADD a, b, c
MOV t1, a
MUL t2, t1, d
MOV t3, a

--------------------------------
Process exited after 44.93 seconds with return val
Press any key to continue . . .
```