

14. Write a C Program for code optimization to eliminate common subexpression.

```
#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 (format: a = b + c): ", i + 1);
        scanf("%s = %s %s %s", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
    }

    printf("\nOptimized Code:\n");

    for (i = 0; i < n; i++) {
        int isCommon = 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) {
                isCommon = 1;

                printf("%s = %s\n", code[i].lhs, code[j].lhs);

                break;
            }
        }

        if (!isCommon) {
            printf("%s = %s %s %s\n", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
        }
    }
}
```

```

    }
}

return 0;

}

```

Output:

The screenshot shows a C++ IDE with a source code editor on the left and a terminal window on the right. The source code is a C++ program that takes a list of expressions, optimizes them by removing redundant operations, and prints the optimized code. The terminal shows the program's execution, including prompts for the number of expressions and the input expressions themselves.

```

#b.cpp
#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 (format: a = b + c): ", i + 1);
        scanf("%s = %s %s %s", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
    }
    printf("\nOptimized Code:\n");
    for (i = 0; i < n; i++) {
        int isCommon = 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) {
                isCommon = 1;
                printf("%s = %s\n", code[i].lhs, code[j].lhs);
                break;
            }
        }
        if (!isCommon) {
            printf("%s = %s %s %s\n", code[i].lhs, code[i].op1, code[i].op, code[i].op2);
        }
    }
    return 0;
}

```

```

Enter number of expressions: 4
Enter expression 1 (format: a = b + c): a = b + c
Enter expression 2 (format: a = b + c): t1 = b + c
Enter expression 3 (format: a = b + c): t2 = t1 * d
Enter expression 4 (format: a = b + c): t3 = b + c

Optimized Code:
a = b + c
t1 = a
t2 = t1 * d
t3 = a

-----
Process exited after 42.34 seconds with return value 0
Press any key to continue . . .

```