Experiment-14

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

Program:

#include <stdio.h>

#include <stdbool.h>

#include <string.h>

#define MAX\_EXPRESSIONS 10

#define MAX\_EXPRESSION\_LENGTH 50

struct Expression {

char name[MAX\_EXPRESSION\_LENGTH];

char expression[MAX\_EXPRESSION\_LENGTH];

};

struct CommonSubexpressionTable {

int numExpressions;

struct Expression expressions[MAX\_EXPRESSIONS];

};

bool isSubexpression(struct CommonSubexpressionTable \*table, char \*expression) {

int i;

for (i = 0; i < table->numExpressions; i++) {

if (strcmp(table->expressions[i].expression, expression) == 0) {

return true;

}

}

return false;

}

void eliminateCommonSubexpressions(struct CommonSubexpressionTable \*table) {

int i,j;

for ( i = 0; i < table->numExpressions; i++) {

for ( j = i + 1; j < table->numExpressions; j++) {

if (strcmp(table->expressions[i].expression, table->expressions[j].expression) == 0) {

printf("Subexpression \"%s\" is common. Replacing %s with %s.\n",

table->expressions[i].expression, table->expressions[j].name, table->expressions[i].name);

strcpy(table->expressions[j].expression, table->expressions[i].name);

}

}

}

}

int main() {

struct CommonSubexpressionTable table;

printf("Enter the number of expressions: ");

scanf("%d", &table.numExpressions);

int i;

for (i = 0; i < table.numExpressions; i++) {

printf("Enter expression name for expression %d: ", i + 1);

scanf("%s", table.expressions[i].name);

printf("Enter expression for %s: ", table.expressions[i].name);

scanf("%s", table.expressions[i].expression);

}

eliminateCommonSubexpressions(&table);

printf("\nAfter common subexpression elimination:\n");

for (i = 0; i < table.numExpressions; i++) {

printf("%s = %s\n", table.expressions[i].name, table.expressions[i].expression);

}

return 0;

}

Out put:

