

MANOJ RL 220701161

EXP NO : 12

DATE :

## IMPLEMENT CODE OPTIMIZATION TECHNIQUES COPY PROPAGATION

### AIM:

The aim is to implement code optimization techniques like Dead Code Elimination (DCE) and Common Subexpression Elimination (CSE) to improve the efficiency and performance of a program. These techniques are applied to intermediate code (e.g., Three-Address Code or TAC) during the compilation process.

### ALGORITHM:

- The desired header files are declared.
- The two file pointers are initialized one for reading the C program from the file and one for writing the converted program with constant folding
- The file is read and checked if there are any digits or operands present.
- If there is, then the evaluations are to be computed in switch case and stored.
- Copy the stored data to another file. □ Print the copied data file.

### PROGRAM:

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

#define MAX 100

typedef struct {
    char lhs[10];
    char op1[10];
    char op[5];
    char op2[10];
} Instruction;

int is_copy_instruction(Instruction *ins) {    return
strcmp(ins->op, "=") == 0 && strlen(ins->op2) == 0;
}

void copy_propagation(Instruction ins[], int count) {
    for (int i = 0; i < count; i++) {
        if (is_copy_instruction(&ins[i])) {
            char from[10], to[10];
            strcpy(to, ins[i].lhs);
```

MANOJ RL 220701161

```
        strcpy(from, ins[i].op1);

        for (int j = i + 1; j < count; j++)
        {
            if (strcmp(ins[j].op1, to) ==
0)
                strcpy(ins[j].op1, from);
            if (strcmp(ins[j].op2, to) == 0)
                strcpy(ins[j].op2, from);
        }
    }
}
```

```
int main() {
    FILE *fin = fopen("input.txt", "r");
    if (!fin) {        printf("Error
opening input.txt\n");
        return 1;
    }
}
```

```
    Instruction ins[MAX];
    int count = 0;
```

```
    char line[100];    while (fgets(line,
sizeof(line), fin)) {
        // Skip blank lines
        if (strlen(line) <= 1) continue;
```

```
        Instruction temp;
        temp.op[0] = '\0';
        temp.op2[0] = '\0';
```

```
        int tokens = sscanf(line, "%s = %s %s %s", temp.lhs, temp.op1, temp.op, temp.op2);
```

```
        if (tokens == 2) {
            // It's a copy statement like: a =
b
            strcpy(temp.op, "=");
            temp.op2[0] = '\0';        } else if
(tokens != 4) {        printf("Invalid
line: %s\n", line);
            continue;
        }
    }
```

```
        ins[count++] = temp;
    }
}
```

MANOJ RL 220701161

```
fclose(fin);

// Perform copy propagation
copy_propagation(ins, count);

// Print optimized code
printf("\nOptimized Code (Copy Propagation Only):\n\n"); for (int i =
0; i < count; i++) { if (strcmp(ins[i].op, "=") == 0 &&
strlen(ins[i].op2) == 0) printf("%s = %s\n", ins[i].lhs, ins[i].op1);
else printf("%s = %s %s %s\n", ins[i].lhs, ins[i].op1, ins[i].op,
ins[i].op2);
}

return 0;
}
```

OUTPUT :

```
Enter statements (e.g., a = b or c = a + d). Enter 'END' to finish:
A=B+C+D
C=B+S+k
END

Optimized code:
A = B+C+D
C = B+S+k
```

Implementation	
----------------	--

MANOJ RL 220701161

Output/Signature	
------------------	--

RESULT:

Thus the above to implement code optimization techniques for copy propagation is executed successfully.