

Ex No:9

Date:

IMPLEMENT CODE OPTIMIZATION TECHNIQUES CONSTANT FOLDING

AIM:

To write a C program to implement Constant Folding (Code optimization Technique).

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 <string.h>
#include <ctype.h>
void main() {
    char s[20];
    char flag[20] = "//Constant";
    char result, equal, operator;
    double op1, op2, interrrslt;
    int a, flag2 = 0;
    FILE *fp1, *fp2;
    fp1 = fopen("input.txt", "r");
    fp2 = fopen("output.txt", "w");
    fscanf(fp1, "%s", s);
    while (!feof(fp1)) {
        if (strcmp(s, flag) == 0) {
            flag2 = 1;
        }
        if (flag2 == 1) {
            fscanf(fp1, "%s", s);
            result = s[0];
            equal = s[1];
            if (isdigit(s[2]) && isdigit(s[4])) {
```

Roll Number: 210701103

Name: P. Kanaga Shanmugam

```

if (s[3] == '+' || s[3] == '-' || s[3] == '*' || s[3] == '/') {
    operator = s[3];
    op1 = s[2] - '0';
    op2 = s[4] - '0';
    switch (operator) {
        case '+':
            interrslt = op1 + op2;
            break;
        case '-':
            interrslt = op1 - op2;
            break;
        case '*':
            interrslt = op1 * op2;
            break;
        case '/':
            if (op2 != 0)
                interrslt = op1 / op2;
            else {
                fprintf(fp2, "Division by zero error.\n");
                fclose(fp1);
                fclose(fp2);
                return;
            }
            break;
        default:
            interrslt = 0;
            break;
    }
    fprintf(fp2, "/*Constant Folding*/\n");
    fprintf(fp2, "%c = %.2lf\n", result, interrslt);
    flag2 = 0;
}
} else {
    fprintf(fp2, "Not Optimized\n");
    fprintf(fp2, "%s\n", s);
}
} else {

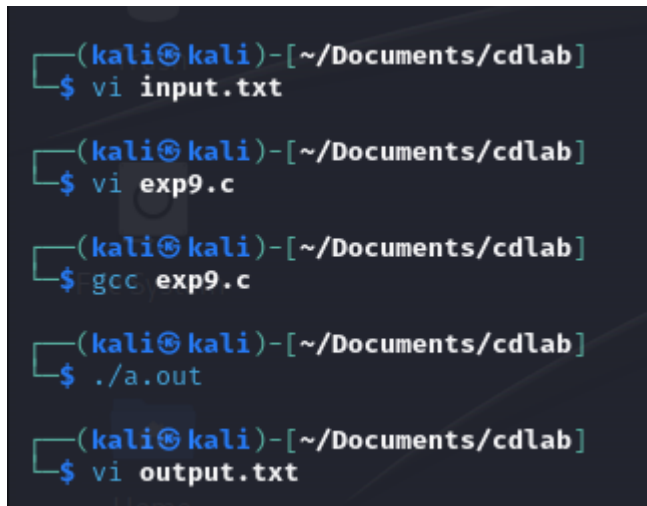
```

```

        fprintf(fp2, "%s\n", s);
    }
    fscanf(fp1, "%s", s);
}
fclose(fp1);
fclose(fp2);
}

```

OUTPUT:



```

(kali㉿kali)-[~/Documents/cdlab]
$ vi input.txt

(kali㉿kali)-[~/Documents/cdlab]
$ vi exp9.c

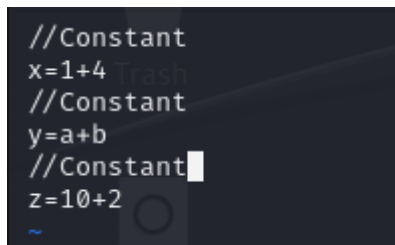
(kali㉿kali)-[~/Documents/cdlab]
$ gcc exp9.c

(kali㉿kali)-[~/Documents/cdlab]
$ ./a.out

(kali㉿kali)-[~/Documents/cdlab]
$ vi output.txt

```

Input.txt:

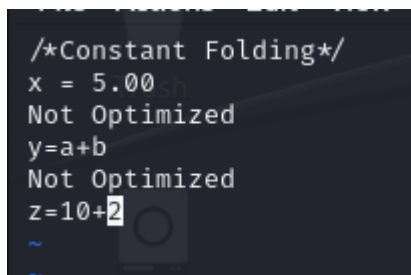


```

//Constant
x=1+4
//Constant
y=a+b
//Constant
z=10+2
~

```

Output.txt:



```

/*Constant Folding*/
x = 5.00
Not Optimized
y=a+b
Not Optimized
z=10+2
~

```

RESULT:

Thus, a C program to implement Constant Folding has been developed.