Experiment – 10

Aim: Implement a Machine Code for a given Intermediate Code

Program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int label[20];
int no = 0;

int check_label(int k) {
    int i;
    for (i = 0; i < no; i++) {
        if (k == label[i])
            return 1;
    }
    return 0;
}

int main() {
    FILE *fp1, *fp2;
    char fname[10], op[10], ch;
    char operand1[8], operand2[8], result[8];
    int i = 0, j = 0;
    printf("\n Enter filename of the intermediate code");
    scanf("%s", &fname);
    fp1 = fopen(fname, "r");
    fp2 = fopen("target.txt", "w");
```

```
if (fp1 == NULL || fp2 == NULL) {
    printf("\n Error opening the file");
    exit(0);
}
while (!feof(fp1)) {
    fprintf(fp2, "\n");
    fscanf(fp1, "%s", op);
    i++;
    if (check_label(i))
        fprintf(fp2, "\nlabel#%d", i);
    if (strcmp(op, "print") == 0) {
        fscanf(fp1, "%s", result);
        fprintf(fp2, "\n\t OUT %s", result);
    }
    if (strcmp(op, "goto") == 0) {
        fscanf(fp1, "%s %s", operand1, operand2);
        fprintf(fp2, "\n\t JMP %s,label#%s", operand1, operand2);
        label[no++] = atoi(operand2);
    }
    if (strcmp(op, "[]=") == 0) {
        fscanf(fp1, "%s %s %s", operand1, operand2, result);
        fprintf(fp2, "\n\t STORE %s[%s],%s", operand1, operand2, result);
    }
    if (strcmp(op, "uminus") == 0) {
        fscanf(fp1, "%s %s", operand1, result);
        fprintf(fp2, "\n\t LOAD -%s,R1", operand1);
        fprintf(fp2, "\n\t STORE R1,%s", result);
    }
```

```
switch (op[0]) {
case '*':
    fscanf(fp1, "%s %s %s", operand1, operand2, result);
    fprintf(fp2, "\n \t LOAD%s,R0", operand1);
    fprintf(fp2, "\n \t LOAD%s,R1", operand2);
    fprintf(fp2, "\n \t MUL R1,R0");
    fprintf(fp2, "\n \t STORE R0,%s", result);
    break;
case '+':
    fscanf(fp1, "%s %s%s", operand1, operand2, result);
    fprintf(fp2, "\n \t LOAD %s,R0", operand1);
    fprintf(fp2, "\n \t LOAD %s,R1", operand2);
    fprintf(fp2, "\n \t ADD R1,R0");
    fprintf(fp2, "\n \t STORE R0,%s", result);
    break;
case '-':
    fscanf(fp1, "%s %s %s", operand1, operand2, result);
    fprintf(fp2, "\n\t LOAD %s,R0", operand1);
    fprintf(fp2, "\n \t LOAD %s,R1", operand2);
    fprintf(fp2, "\n \t SUB R1,R0");
    fprintf(fp2, "\n \t STORE R0,%s", result);
    break;
case '/':
    fscanf(fp1, "%s %s s", operand1, operand2, result);
    fprintf(fp2, "\n \t LOAD %s,R0", operand1);
    fprintf(fp2, "\n \t LOAD %s,R1", operand2);
    fprintf(fp2, "\n \t DIV R1,R0");
    fprintf(fp2, "\n \t STORE R0,%s", result);
```

```
            break;
        case '%':
            fscanf(fp1, "%s %s %s", operand1, operand2, result);
            fprintf(fp2, "\n \t LOAD %s,R0", operand1);
            fprintf(fp2, "\n \t LOAD %s,R1", operand2);
            fprintf(fp2, "\n \t DIV R1,R0");
            fprintf(fp2, "\n \t STORE R0,%s", result);
            break;
        case '=':
            fscanf(fp1, "%s %s", operand1, result);
            fprintf(fp2, "\n\t STORE %s %s", operand1, result);
            break;
        case '>':
            j++;
            fscanf(fp1, "%s %s %s", operand1, operand2, result);
            fprintf(fp2, "\n \t LOAD %s,R0", operand1);
            fprintf(fp2, "\n\t JGT %s,label#%s", operand2, result);
            label[no++] = atoi(result);
            break;
        case '<':
            fscanf(fp1, "%s %s %s", operand1, operand2, result);
            fprintf(fp2, "\n \t LOAD %s,R0", operand1);
            fprintf(fp2, "\n\t JLT %s,label#%d", operand2, result);
            label[no++] = atoi(result);
            break;
        }
    }
    fclose(fp2);
```

```
    fclose(fp1);
    fp2 = fopen("target.txt", "r");
    if (fp2 == NULL) {
        printf("Error opening the file\n");
        exit(0);
    }
    do {
        ch = fgetc(fp2);
        printf("%c", ch);
    } while (ch != EOF);
    fclose(fp1);
    return 0;
}
```

Input File: intput.txt

```
=t1 2
[]=a 0 1
[]=a 1 2
[]=a 2 3
*t1 6 t2
+a[2] t2 t3
-a[2] t1 t2
/t3 t2 t2
uminus t2 t2
print t2
goto t2 t3
=t3 99
uminus 25 t2
*t2 t3 t3
```

uminus t1 t1

+t1 t3 t4

print t4

Output: target.txt

main.c | input.txt | target.txt

```
1     STORE 2 []=a
2
3    LOAD
4     LOADt2,R1
5     MUL R1,R0
6     STORE R0,+a[2]
7
8     LOAD t1,R0
9     LOAD t2,R1
10    SUB R1,R0
11    STORE R0,/t3
12
13    LOAD -t2,R1
14    STORE R1,t2
15
16    OUT t2
17
18    JMP t2,label#t3
19
20    STORE 99 uminus
21
22   LOAD
23    LOADt3,R1
24    MUL R1,R0
25    STORE R0,uminus
26
27    LOAD t3,R0
28    LOAD t4,R1
29    ADD R1,R0
30    STORE R0,print
31
```