ABHINAV RANJAN
RA1911003010003
CSE A1 SECTION
SRMIST , KTR

COMPILER DESIGN LAB

EXP 10 - INTERMEDIATE CODE GENERATOR

24-03-2002

EXP 10 – INTERMEDIATE CODE GENERATOR

AIM

To implement a C program for intermediate code generation.

REQUIREMENTS

1. Knowledge of the concepts of Compiler design and intermediate code generation
2. Online GDB compiler to run and execute code.

THEORY

In the analysis – synthesis model of a compiler, the front end of a compiler translates a source program into an independent intermediate code, then the back end of the compiler uses this intermediate code to generate the target code which can be understood by a machine.

Benefits

* enhanced portability of the code
* retargeting is facilitated
* easier to apply source code modification to improve the performance of source code by optimising the intermediate code.

ALGORITHM

1. Start
2. Get address code sequence
3. Determine current location of 3 using address [for 1st operand]
4. If current location not already exist generate move (B, 0).
5. Update address of A [for 2nd operand]
6. If current value of B and C) is null, exit.
7. If they generate operator C) A, 3 ADPR.
8. Store the move instruction in memory.
9. Stop.

SCREENSHOTS :

main.c | input.txt | output.txt

```c
#include<stdio.h>
#include<conio.h>
#include<string.h>
char op[2],arg1[5],arg2[5],result[5];
void main()
{
  FILE *fp1,*fp2;
  fp1=fopen("input.txt","r");
  fp2=fopen("output.txt","w");
  while(!feof(fp1))
  {

    fscanf(fp1,"%s%s%s%s",op,arg1,arg2,result);
    if(strcmp(op,"+")==0)
    {
      fprintf(fp2,"\nMOV R0,%s",arg1);
      fprintf(fp2,"\nADD R0,%s",arg2);
      fprintf(fp2,"\nMOV %s,R0",result);
    }
    if(strcmp(op,"*")==0)
    {
      fprintf(fp2,"\nMOV R0,%s",arg1);
      fprintf(fp2,"\nMUL R0,%s",arg2);
```

input

INPUT FILE :

main.c | input.txt | output.txt

```
+ a b t1
* c d t2
- t1 t2 t
= t ? x
```

OUTPUT FILE:

Run Debug Stop Share Save { } Beautify

main.c input.txt output.txt

```
MOV R0,a
ADD R0,b
MOV t1,R0
MOV R0,c
MUL R0,d
MOV t2,R0
MOV R0,t1
SUB R0,t2
MOV t,R0
MOV R0,t
MOV x,R0
```

input

```
...Program finished with exit code 0
Press ENTER to exit console.
```

RESULT

Hence the c program for intermediate code generator was implemented successfully.