Practical 9
Compiler Construction
19BCE248

Aim:- To implement an assembly code generator

Code:

main. 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);
          fprintf(fp2,"\nMOV %s,R0",result);
        }

        if(strcmp(op,"-")==0)
        {
          fprintf(fp2,"\nMOV R0,%s",arg1);
          fprintf(fp2,"\nSUB R0,%s",arg2);
          fprintf(fp2,"\nMOV %s,R0",result);
        }

        if(strcmp(op,"/")==0)
        {
          fprintf(fp2,"\nMOV R0,%s",arg1);
          fprintf(fp2,"\nDIV R0,%s",arg2);
          fprintf(fp2,"\nMOV %s,R0",result);
        }

        if(strcmp(op,"=")==0)
        {
          fprintf(fp2,"\nMOV R0,%s",arg1);
          fprintf(fp2,"\nMOV %s,R0",result);
        }
    }
    fclose(fp1);
    fclose(fp2);
    getch();
}
```

Output:

Input.txt:

```
+ x y t1
* z w t2
- t2 t1 t
= t ? a
```

Output.txt:

```
MOV R0,x
ADD R0,y
MOV t1,R0
MOV R0,z
MUL R0,w
MOV t2,R0
MOV R0,t2
SUB R0,t1
MOV t,R0
MOV R0,t
MOV a,R0
```