25/11/2022 PASS1 of two pass Assembler

Aim: to implement the first pass of two pass Assembler

program:

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main(){
    char opcode[10], operand[10], label[10], code[10], mnemonic[10];
    int locctr, start, length;
    FILE *fp1, *fp2, *fp3, *fp4, *fp5;
    fp1 = fopen("input.txt","r");
    fp3 = fopen("optab.txt","r");
    fp2 = fopen("inter.txt","w");
    fp4 = fopen("symtab.txt","w");
    fp5 = fopen("length.txt","w");
fscanf(fp1,"%s %s %s",label,opcode,operand);
    if(strcmp(opcode,"START")==0){
        start = atoi(operand)
        locctr = start;
        fprintf(fp4,"\t%s\t%s\t%s\n",label,operand,opcode);
        fscanf(fp1,"%s %s %s",label,opcode,operand);
    else
        locctr = 0;
    while(strcmp(opcode,"END")!=0){
        fprintf(fp4,"%d\t",locctr);
        if(strcmp(label,"*")!=0)
            fprintf(fp2,"%s\t%d\n",label,locctr);
            fscanf(fp2,"%s %s",code,mnemonic);
    if(strcmp(opcode,code)==0){
```

```
            locctr += 3
            break;
        }
    fscanf(fp2, "%s %s %s", mnemonic); }
    if (strcmp(opcode, "WORD") == 0)
            locctr += 3
    if (strcmp(opcode, "RESW") == 0)
            locctr += (3 * atoi(operand));
    if (strcmp(opcode, "Byte") == 0)
            locctr += atoi(operand);
    if (strcmp(opcode, "RESB") == 0)
            ++locctr;
    fprintf(fp4, "%s\t%s\t%s\t%s\n", label, opcode, operand);
    fscanf(fp1, "%s%s%s", label, opcode, operand); }
    fprintf(fin, "%s\t%s\t%s\t%s\n", label, opcode, operand);
    length = locctr - start
    fprintf(fp5, "%d", length);
    fclose(fp1);
    fclose(fp2);
    fclose(fp3);
    fclose(fp4);
    fclose(fp5); }
```

Result:

program is executed and o/p is verified successfully.

Algorithm

```
begin
    read the first i/p line
    if opcode 'START' then
    begin
      save operand as starting address
      initialize locctr to starting address
      write line to intermediate file
      read next line
      end if
    else
      init locctr to 0
      while opcode ≠ END
        begin
      if this is not comment line
      begin
        search symtab for label.
          if found
              set error flag
          else
              insert (label, locctr) into symtab)
          search OPTAB for opcode.
          if found then
          add 3 to locctr
        else if opcode = 'WORD' then
            add 3 to locctr
        else if opcode = RESW then
            add 3* # operand to locctr
        else if opcode = RESB then
                add, #[operand] to locctr
        else if opcode = RESB then
                add, #[operand] to locctr
        else if opcode = 'Byte' then
            begin
                find length of constant in byte
                 add length to locctr
              end if
```

```
else
    Set error flag
end
write line to Intermediate file
read next I/P line
end
write last line to Intermediate file
Save(locctr.start) if locctr of program
end
```

input.text

```
COPY    START   2000
*       LDA     FIVE
*       STA     ALPHA
*       LDCH    CHAR
*       STCH    C1
*       MOV     A,B
ALPHA   RESW    1
FIVE    WORD    5
CHAR    BYTE    C
TEST    WORD    6
C1      RESW    1
*       END     * *
```

optab.txt

```
ADD 18
SUB 1C
LDA 00
LDB 68
LDX 54
STB 78
STX L0
STCH 0C
LDCH 50
MUL 20
MOV 1D
START *
END *
```

Symtab.txt

```
ALPHA   2015
FIVE    2018
CHAR    2021
TEST    2022
C1      2025
```

inter.txt

	COPY	START	2000
2000	*	LDA	FIVE
2003	*	STA	ALPHA
2006	*	LDCH	CHAR2
2009	*	STCH	C1
2012	*	MOV	A,B
2015	ALPHA	RESW	1
2018	FIVE	WORD	15
2021	CHAR	BYTE	C'EOF'
2022	TEST	WORD	6
2025	C1	RESW	1
*		END	**

length.txt

25