/* Implement PASS ONE of a Two pass assembler */

Aim:

To write a C program to implement PASS ONE of a two pass assembler

Algorithm:

Open the files fp1 and fp4 in read mode and fp2 and fp3 in write mode
Read the source program
If the opcode read in the source program is START, the variable location counter is initialized with the operand value.
Else the location counter is initialized to 0.
The source program is read line by line until the reach of opcode END.
Check whether the opcode read is present in the operation code table.
If the opcode is present, then the location counter is incremented by 3.
If the opcode read is WORD, the location counter is incremented by3.
If the opcode read is RESW, the operand value is multiplied by 3 and then the location counter is incremented.
If the opcode read is RESB, the location counter value is incremented by operand value.
If the opcode read is BYTE, the location counter is auto incremented.
The length of the source program is found using the location counter value.

INPUT FILES

INPUT.DAT

```
** START 2000
** LDA FIVE
** STA ALPHA
** LDCH CHARZ
** STCH C1
ALPHA RESW 1
FIVE WORD 5
CHARZ BYTE C'Z'
C1 RESB 1
** END **
```

OPTAB.DAT

```
START
LDA
STA
LDCH
STCH
END
```

// **Source Code program in c pass one of a two pass assembler.**

```
# include <stdio.h>
# include <conio.h>
# include <string.h>
void main()
{
char opcode[10],mnemonic[3],operand[10],label[10],code[10];
int locctr,start,length;
FILE *fp1,*fp2,*fp3,*fp4;
clrscr();
fp1=fopen("input.dat","r");
fp2=fopen("symtab.dat","w");
fp3=fopen("out.dat","w");
fp4=fopen("optab.dat","r");
fscanf(fp1,"%s%s%s",label,opcode,operand);
if(strcmp(opcode,"START")==0)
{
start=atoi(operand);
locctr=start;
fprintf(fp3,"\t%s\t%s\t%s\n",label,opcode,operand);
fscanf(fp1,"%s%s%s",label,opcode,operand);
}
else
locctr=0;
while(strcmp(opcode,"END")!=0)
{
fprintf(fp3,"%d\t",locctr);
if(strcmp(label,"**")!=0)
fprintf(fp2,"%s\t%d\n",label,locctr);
rewind(fp4);
fscanf(fp4,"%s",code);
while(strcmp(code,"END")!=0)
{
if(strcmp(opcode,code)==0)
{
locctr+=3;
break;
}
fscanf(fp4,"%s",code);
}
if(strcmp(opcode,"WORD")==0)
locctr+=3;
else if(strcmp(opcode,"RESW")==0)
locctr+=(3*(atoi(operand)));
else if(strcmp(opcode,"RESB")==0)
```

```
locctr+=(atoi(operand));
else if(strcmp(opcode,"BYTE")==0)
++locctr;
fprintf(fp3,"%s\t%s\t%s\n",label,opcode,operand);
fscanf(fp1,"%s%s%s",label,opcode,operand);
}
fprintf(fp3,"%d\t%s\t%s\t\%s\n",locctr,label,opcode,operand);
length=locctr-start;
printf("The length of the program is %d",length);
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
getch();
}
```

OUTPUT FILES

OUT.DAT

```
** START 2000
2000 ** LDA FIVE
2003 ** STA ALPHA
2006 ** LDCH CHARZ
2009 ** STCH C1
2012 ALPHA RESW 1
2015 FIVE WORD 5
2018 CHARZ BYTE C'Z'
2019 C1 RESB 1
2020 ** END **
```

SYMTAB.DAT

```
ALPHA 2012
FIVE 2015
CHARZ 2018
C1 2019
```

Send me your feedback on this Article

Website Address: Pass one of a two pass assembler