NIRAJ KUMAR

BT19CS031

QUESTION 1

Write a C program to read a SIC (Simplified Instructional Computer) assembly language program and create three text (.txt) files. These three files must content labels, instructions, and operands present in the program separately. Assume that the columns of the SIC program are tab-separated.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#define _GNU_SOURCE
#include <assert.h>
#include <stdlib.h>
#include <stdlib.h>
FILE *fpr=NULL,*fpw=NULL,*fps=NULL,*fpi=NULL,*fpo=NULL;
char addr(char ch)
   int n;
n=0; n=ch-'0';
n+=8;
        if(n==8)
return '8';
else if(n==9)
return '9';
else if(n==10)
return 'A';
else if(n==11)
return 'B';
else if(n==12)
return 'C';
else if(n==13)
return 'D';
else if(n==14)
return 'E';
```

```
else if(n==15)
return 'F';
}
   int
main() {
           char label[10],mnemonic[10]
,operand[10],program_name[10],sym_label[30][10],sym_address[30][10],
opcodes[1000][3],opcode[10],str[80],st[10],ch,c,chr[10],text[80][80]
;
int
address=0, line=0, length, x=0, first_address=0, last_address=0, pa=0, obj_
code=0, i=0, j=0, k=0, pos1=0,
           pos2=0, flaq1=0, flaq2=0, p1=0, p2=0, l=0, z=0, n=0, count=0, l1=0, l2=0;
op mnemonic[26][10]={"LDAC", "STAC", "SUBJ", "MULT", "STRL", "DIVD", "ADDA
", "STRCH", "DT",
"JMPEQ", "LOADCH", "DW", "SUBR", "FIXR", "FIX", "JMPLT", "COMP", "CLR", "LOAD
B", "J", "DR", "COMPR", "STREX", "LOADT", "LOADL", "LOADX"};
char
op_opcode[26][10]={"10","1C","58","30","24","34","28","44","F0","40"
,"60","EC",
"4C", "C8", "3C", "39", "38", "C4", "78", "3C", "E8", "D0", "20", "84", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "04", "08", "08", "04", "08", "08", "04", "08", "08", "04", "08", "08", "04", "08", "08", "04", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", "08", 
"};
           /*Adressing*/
          fpr=fopen("new_input.txt","r+");
fpw=fopen("addressed_file.txt","w+");
          fscanf(fpr, "%s%s%X", &program name, &mnemonic, &address);
/*Reading data from input.txt*/
line++;
fprintf(fpw, "%d\t\t%s\t%s\t%.4X\n", line
,program_name,mnemonic,addres
s); /*Writing data into addressed file.txt */
           while(strcmp(mnemonic, "END")!=0)
           {
                      if(strcmp(label,".")!=0) /*For checking comments*/
                                 fscanf(fpr, "%s%s%s", &Label, &mnemonic, &operand);
                                 line++;
```

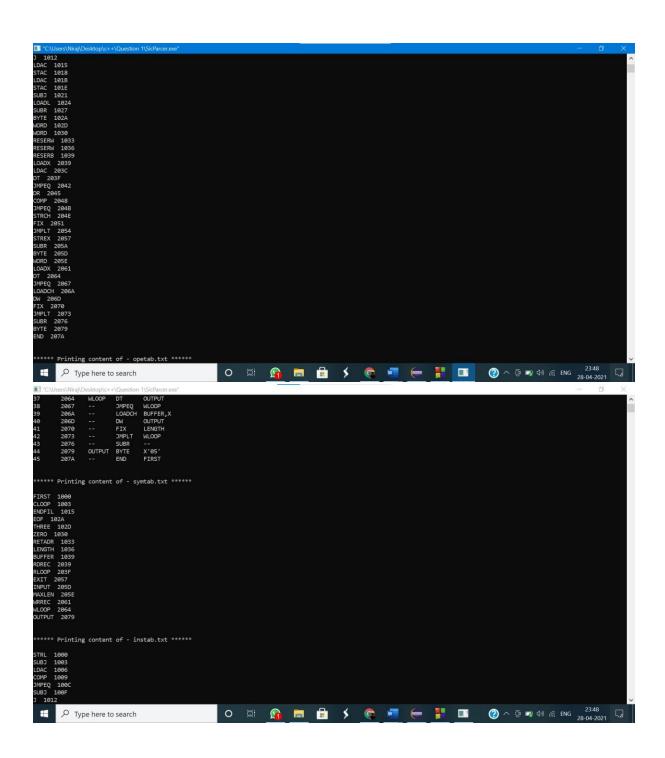
```
fprintf(fpw, "%d\t%.4X\t%s\t%s\t", line, address, label, mnemonic, ope
rand); /*Writing data into addressed file.txt */
            /*checking conditions for various mnemonics*/
            if(strcmp(mnemonic, "BYTE")==0)
                              Length=0;
length=strlen(operand);
                                   /*excluding C'' ,X'' */
                Length-=3;
                if(operand[0]=='C')
address+=length;
                else
{
                    length=(length/2);
address+=length;
}
            else if(strcmp(mnemonic, "WORD")==0)
                address+=3;
            else if(strcmp(mnemonic, "RESERW")==0)
                x=atoi(operand); /*converting character into
integer */
                address+=(x*3); /*1 WORD = 3 BYTES */
            else if(strcmp(mnemonic, "RESERB")==0)
                x=atoi(operand);
                                   /*change RESERB's string
mnemonic into integer --> x */
                                               address+=x;
}
              else
                address+=3;
else
fprintf(fpw, "\
n");
fclose(fpr);
fclose(fpw);
    /*Creating Symtab*/
    fpw=fopen("addressed_file.txt","r+");
fps=fopen("symtab.txt","w+");
                                 fpi=fopen("instab.txt","w+");
fpo=fopen("opetab.txt","w+");
```

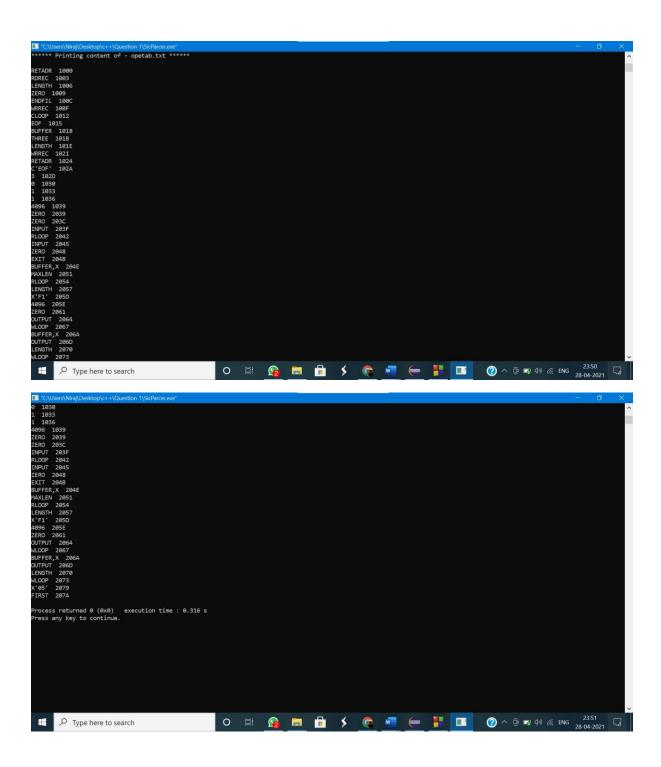
```
fscanf(fpw, "%d%s%s%s", &line, &program_name, &mnemonic, &operand);
    while(strcmp(mnemonic, "END")!=0)
fscanf(fpw, "%d%X%s%s%s", &Line, &address, &Label, &mnemonic, &operand);
        if(line==2)
            first_address=address; /*Will be used during object
file creation */
        if(strcmp(label, "--")!=0)
            fprintf(fps, "%s %X\n", label, address);
j++;
        if(strcmp(mnemonic,"--")!=0)
            fprintf(fpi, "%s %X\n", mnemonic, address);
        if(strcmp(operand, "--")!=0)
            fprintf(fpo, "%s %X\n", operand, address);
}
    last_address=address; /*Will be used during object file
creation */ fclose(fpr);
                                fclose(fpw);
                                                 fclose(fps);
fclose(fpi);
                fclose(fpo);
    printf("\n\n****** Printing content of - addressed_file.txt
*****\n\n");
    fpw=fopen("addressed_file.txt","r");
    char _ch=fgetc(fpw);
while(!feof(fpw))
    {
        printf("%c",_ch);
       _ch=fgetc(fpw);
    }
        printf("\n\n****** Printing content of - symtab.txt
*****\n\n");
    fps=fopen("symtab.txt","r");
    _ch=fgetc(fps);
while(!feof(fps))
    {
        printf("%c",_ch);
       _ch=fgetc(fps);
    }
        printf("\n\n****** Printing content of - instab.txt
*****\n\n");
```

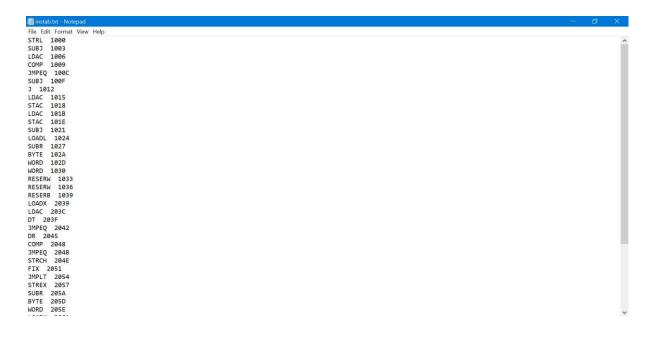
```
fpi=fopen("instab.txt", "r");
     _ch=fgetc(fpi);
while(!feof(fpi))
    {
        printf("%c",_ch);
        _ch=fgetc(fpi);
    }
        printf("\n\n****** Printing content of - opetab.txt
*****\n\n");
    fpo=fopen("opetab.txt", "r");
    _ch=fgetc(fpo);
while(!feof(fpo))
    {
        printf("%c",_ch);
        _ch=fgetc(fpo);
    }
fclose(fpr);
fclose(fpw);
fclose(fps);
fclose(fpi);
fclose(fpo);
return 0;
```

OUTPUT –

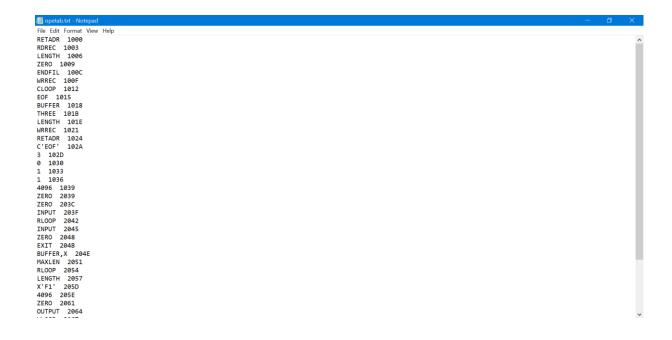
```
Comparison (Comparison (Compar
```



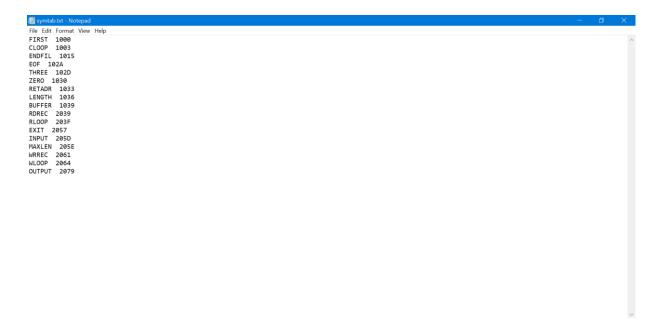


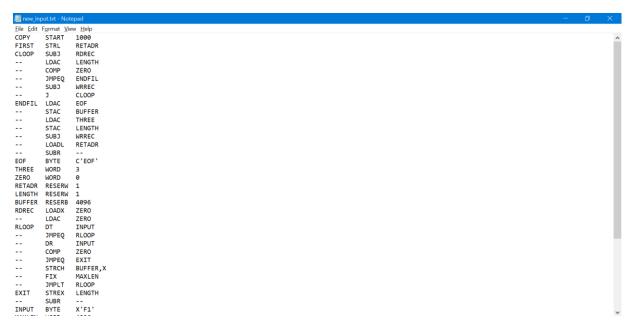


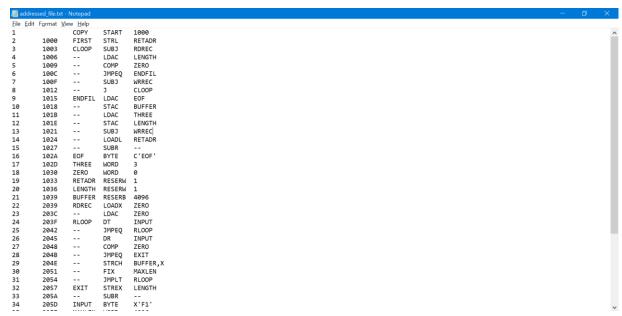












📕 add	lressed_file.txt	- Notepad		
Eile Eo	dit F <u>o</u> rmat \	iew <u>H</u> elp		
13	1021		SUBJ	WRREC
14	1024		LOADL	RETADR
15	1027		SUBR	
16	102A	EOF	BYTE	C'EOF'
17	102D	THREE	WORD	3
18	1030	ZERO	WORD	0
19	1033	RETADR	RESERW	1
20	1036	LENGTH	RESERW	1
21	1039	BUFFER	RESERB	4096
22	2039	RDREC	LOADX	ZERO
23	203C		LDAC	ZERO
24	203F	RLOOP	DT	INPUT
25	2042		JMPEQ	RLOOP
26	2045		DR	INPUT
27	2048		COMP	ZERO
28	204B		JMPEQ	EXIT
29	204E		STRCH	BUFFER,X
30	2051		FIX	MAXLEN
31	2054		JMPLT	RLOOP
32	2057	EXIT	STREX	LENGTH
33	205A		SUBR	
34	205D	INPUT	BYTE	X'F1'
35	205E	MAXLEN	WORD	4096
36	2061	WRREC	LOADX	ZERO
37	2064	WLOOP	DT	OUTPUT
38	2067		JMPEQ	WLOOP
39	206A		LOADCH	BUFFER,X
40	206D		DW	OUTPUT
41	2070		FIX	LENGTH
42	2073		JMPLT	WLOOP
43	2076		SUBR	
44	2079	OUTPUT	BYTE	X'05'
45	207A		END	FIRST