Pass1

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main()
{
  char ch,label[10],opcode[10],operand[10],code[10],mnemonic[10],read[50];
  int locctr,start,length;
  FILE *fp1,*fp2,*fp3,*fp4,*fp5,*fp6;

  fp1=fopen("input.txt","r");
  fp2=fopen("optab.txt","r");
  fp3=fopen("symtab.txt","w");
  fp4=fopen("infile.txt","w");
  fp5=fopen("length.txt","w");

  fscanf(fp1,"%s\%s\t%s",label,opcode,operand);
  if(strcmp(opcode,"START")==0)
  {
    start = atoi(operand);
    locctr = start;
    fprintf(fp4,"**\t%s\t%s\t%s\n",label,opcode,operand);
    fscanf(fp1,"%s\t%s\t%s",label,opcode,operand);
  }
  else
  {
    locctr = 0;
  }
  while(strcmp(opcode,"END")!=0)
  {
    fprintf(fp4,"%d\t",locctr);
    if(strcmp(label,"**")!=0)
    {
      fprintf(fp3,"%s\t%d\n",label,locctr);
    }
    fscanf(fp2,"%s\t%s",mnemonic,code);
    while(strcmp(mnemonic,"END")!=0)
    {
      if(strcmp(opcode,mnemonic)==0)
      {
        locctr+=3;
        break;
      }
      fscanf(fp2,"%s\t%s",mnemonic,code);
```

```
  }
  if(strcmp(opcode,"WORD")==0)
  {
    locctr+=3;
  }
  else if(strcmp(opcode,"RESW"))
  {
    locctr+=(3*atoi(operand));
  }
  else if(strcmp(opcode,"RESB"))
  {
    locctr+=atoi(operand);
  }
  else if(strcmp(opcode,"BYTE"))
  {
    locctr+=1;
  }
  fprintf(fp4,"%s\t%s\t%s\n",label,opcode,operand);
  fscanf(fp1,"%s\t%s\t%s",label,opcode,operand);

}
fprintf(fp4,"%d\t%s\t%s\t%s\n",locctr,label,opcode,operand);
length = locctr-start;
fprintf(fp5,"%d",length);
fclose(fp1);
fclose(fp2);
fclose(fp3);
fclose(fp4);
printf("The contents of intermediate file are :\n");
fp4=fopen("infile.txt","r");
ch=fgetc(fp4);
while(ch!=EOF)
{
  printf("%c",ch);
  ch=fgetc(fp4);

}
printf("\nThe Contents of SYMTAB are:\n");
fp3=fopen("symtab.txt","r");
ch=fgetc(fp3);
while(ch!=EOF)
{
  printf("%c",ch);
```

```
    ch=fgetc(fp3);
  }
  printf("\nThe length of the program is :\n%d",length);
  fclose(fp4);
  fclose(fp3);
}
```

Absolute loader

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

void main() {
  FILE * fp;
  int i,l, j, staddr1;
  char name[10], line[50], name1[10],staddr[10];
  printf("enter program name for verify:\n");
  scanf("%s", name);
  fp = fopen("input.txt", "r");
  fscanf(fp, "%s", line);
  for (i = 2, j = 0; i < 8, j < 6; i++, j++)
    name1[j] = line[i];
    name1[j] = '\0';
  printf("program name is: %s\n", name1);
  if (strcmp(name, name1) == 0) {
    printf("verification success \n");
    do {
      fscanf(fp, "%s", line);
      if (line[0] == 'T') {
        for (i = 2, j = 0; i < 8, j < 6; i++, j++)
          staddr[j] = line[i];
        staddr[j] = '\0';
        staddr1 = atoi(staddr);
        i = 12;

        while (line[i] != '\0') {
          if (line[i] != '^') {
             printf("%04d \t %c%c\n", staddr1, line[i], line[i + 1]);
            staddr1++;
            i = i + 2;
          } else i++;
        }
      } else if (line[0] = 'E')
        fclose(fp);
```

```
    } while (!feof(fp));
  }
  else
    printf("prgram  name is different verification failed\n");

}
```

Relocation

```
#include <math.h>
#include <stdio.h>
#include <string.h>

char bit[12] = {0};
char zero[10];

void convert(char h[12]);
int hexToDecimal(char acHex[]);
int power(int x, int y);
char* sixDigitConverter(char addr[10]);

int main() {
  char input[100], staddr[10], mask[12], arrOfAddr[10][50], finalAddress[10],
      address[10], line[20];
  int finalAddressinDec, noOfT=0;
  int i, j, k, len;
  FILE *fp1, *fp2;
  fp1 = fopen("relocsic.txt", "r");
  fp2 = fopen("relocsicOUT.txt", "w");
  printf("Enter the starting address : ");
  scanf("%s", address); // ddddddddddddd
  fscanf(fp1, "%s", input);
  // strcpy(line,input);
  //  printf("%s",input);
  char *word = strtok(input, "^");
  i = 0;
  while (word != NULL) {
    if (i == 2) {
      strcpy(staddr, word);
      fprintf(fp2, "%s", sixDigitConverter(address));
    } else {
      fprintf(fp2, "%s", word);
    }
```

```
  if (i != 3) {
    fprintf(fp2, "^");
  }

  i++;
  word = strtok(NULL, "^");
}

while (!feof(fp1)) {
  noOfT++;
  fscanf(fp1, "%s", input);
  char *word = strtok(input, "^");
  i = 0;
  j = 0;
  len = strlen(input);
  if(strcmp(input,"E")){
    fprintf(fp2,"\nT");
  }
  while (word != NULL) {
    if((i==1||i==2)&&(strcmp(input,"E"))){
      fprintf(fp2,"^");
      if(i==1){
        sprintf(finalAddress,"%x",hexToDecimal(address) + hexToDecimal(word));
        fprintf(fp2,"%s",sixDigitConverter(finalAddress));
      }
      else {
        fprintf(fp2,"%s",word);
      }
    }
    if (i == 3&&strcmp(input,"E")) {
      strcpy(mask, word);
      fprintf(fp2,"^");
      fprintf(fp2,"%s",mask);
    }
    if (i > 3) {
      strcpy(arrOfAddr[j], word);
      // printf("%s\n", arrOfAddr[j]);
      j++;
    }
    i++;
    word = strtok(NULL, "^");
  }

  convert(mask); // stored in bit
```

```
    for (k = 0; k < j; k++) {
      if (bit[k] == '1') {
        // use hextodec to add address and locctr
        finalAddressinDec = hexToDecimal(address) + hexToDecimal(arrOfAddr[k]);
        // printf("dec=%d
        // %d\n",hexToDecimal(address),hexToDecimal(arrOfAddr[k]));
        sprintf(finalAddress, "%x", finalAddressinDec);
        fprintf(fp2,"^");
        fprintf(fp2,"%s", sixDigitConverter(finalAddress));
        // finalAddress is the changed
      }
      else {
        fprintf(fp2,"^");
        fprintf(fp2,"%s", sixDigitConverter(arrOfAddr[k]));
      }
    }
  }
  fprintf(fp2,"\nE^");
  fprintf(fp2,"%s", sixDigitConverter(address));
}

char* sixDigitConverter(char addr[10]) {
  int i;
  zero[0]='\0';
  if(strlen(addr)==2)
    return addr;
  for(i=strlen(addr);i<6;i++){
    strcat(zero,"0");
  }
  strcat(zero,addr);
  return zero;
}

void convert(char h[12]) {
  int i, l;
  strcpy(bit, "");
  l = strlen(h);
  for (i = 0; i < l; i++) {
    switch (h[i]) {
    case '0':
      strcat(bit, "0");
      break;
    case '1':
      strcat(bit, "1");
```

```
  break;
case '2':
  strcat(bit, "10");
  break;
case '3':
  strcat(bit, "11");
  break;
case '4':
  strcat(bit, "100");
  break;
case '5':
  strcat(bit, "101");
  break;
case '6':
  strcat(bit, "110");
  break;
case '7':
  strcat(bit, "111");
  break;
case '8':
  strcat(bit, "1000");
  break;
case '9':
  strcat(bit, "1001");
  break;
case 'A':
  strcat(bit, "1010");
  break;
case 'B':
  strcat(bit, "1011");
  break;
case 'C':
  strcat(bit, "1100");
  break;
case 'D':
  strcat(bit, "1101");
  break;
case 'E':
  strcat(bit, "1110");
  break;
case 'F':
  strcat(bit, "1111");
  break;
}
```

```
  }
}

int hexToDecimal(char acHex[]) {
  int len, i, temp;
  int dec = 0;
  len = strlen(acHex);
  for (i = 0; i < len; i++) {
    if (acHex[i] >= '0' && acHex[i] <= '9')
      temp = acHex[i] - '0';
    else if (acHex[i] >= 'a' && acHex[i] <= 'z')
      temp = acHex[i] - 'a' + 10;
    else
      temp = acHex[i] - 'A' + 10;
    dec = dec + temp * (power(16, len - i - 1));
  }
  return dec;
}

int power(int x, int y) {
  int power = 1, i;
  for (i = 1; i <= y; ++i) {
    power = power * x;
  }
  return power;
}
```

Macro

```
#include<stdio.h>
//#include<conio.h>
#include<ctype.h>
#include<string.h>
int m=0,i,j,flag=0;
char c,*s1,*s2,*s3,*s4,str[50]="",str1[50]="";
char mac[10][10];
void main()
{
    FILE * fpm=fopen("macro.txt","r");
    FILE * fpi=fopen("minput.txt","r");
    FILE * fpo=fopen("moutput.txt","w");
    //clrscr();
    while(!feof(fpm))
    {
        fgets(str,50,fpm);
```

```
    s1=strtok(str,"");
    s2=strtok(NULL,"");
    if(strcmp(s1,"MACRO")==0)
    {
        strcpy(mac[m],s2);
        m++;
    }
    s1=s2=NULL;
}
fgets(str,50,fpi);
while(!feof(fpi))
{
    flag=0;
    strcpy(str1,str);
    for(i=0;i<m;i++)
    {
        if(strcmp(str1,mac[i])==0)
        {
            rewind(fpm);
            while(!feof(fpm))
            {
                fgets(str,50,fpm);
                s2=strtok(str,"");
                s3=strtok(NULL,"");
                if(strcmp(s2,"MACRO")==0&&strcmp(s3,str1))
                {
                    fgets(str,50,fpm);
                    strncpy(s4,str,4);
                    s4[4]='\0';
                    while(strcmp(s4,"MEND")!=0);
                    {
                        fprintf(fpo,"%s",str);
                        printf("\n####%s",str);
                        fgets(str,50,fpm);
                        strncpy(s4,str,4);
                        s4[4]='\0';
                    }
                }
            }
            flag=1;
            break;
        }
    }
    if(flag==0)
```

```
        {
            fprintf(fpo,"%s",str);
            printf("%s",str);
        }
        fgets(str,50,fpi);
    }
    fclose(fpm);
    fclose(fpi);
    fclose(fpo);
}
```

Pass2

```
#include<stdio.h>
#include<string.h>
#include<math.h>
#include<stdlib.h>
int power(int x,int y)
{
        int power = 1, i;
        for (i = 1; i <= y; ++i)
        {
                power = power * x;
        }
        return power;
}
int hexToDecimal(char acHex[]){
        int len,i,temp;
        int dec=0;
        len=strlen(acHex);
        for(i=0;i<len;i++){
                if(acHex[i]>='0'&&acHex[i]<='9')
                        temp=acHex[i]-'0';
                else if(acHex[i]>='a'&&acHex[i]<='z')
                        temp=acHex[i]-'a'+10;
                else
                        temp=acHex[i]-'A'+10;
                dec=dec+temp*(power(16,len-i-1));
        }
        return dec;
}
int j=-1,m=70,n=70,k=0;
char T[100][110];
```

```
void newT(char loc[]){
	if(j>=0){
		char lenT[10];
		int lth=(n-9)/2;
		sprintf(lenT,"%x",lth);
		T[j][k]='\0';
		T[j][9]=strlen(lenT)==1?'0':lenT[0];
		T[j][10]=lenT[1];
	}
	j++;
	T[j][0]='T';
	T[j][1]='^';
	k=2;
	for(int l=strlen(loc);l<6;l++)
		T[j][k++]='0';
	T[j][k]='\0';
	strcat(T[j],loc);
	T[j][8]=T[j][11]='^';
	T[j][9]=T[j][10]=' ';
	m=n=9;k=12;
}
void secondpass(){
	char label[30],opcode[30],operand[30],loc[30],prgmname[30];
	char opc[30],mnemo[30],sym[30],addr[30],byte[30];
	char startAddress[30];
	int l=0,temp,length;
	int locctr,i;
	FILE *fp,*f1,*f2;
	fp=fopen("intermediate.txt","r");
	while(!feof(fp)){
		fscanf(fp,"%s%s%s%s",loc,label,opcode,operand);
		if(strcmp(opcode,"START")==0){
			strcpy(prgmname,label);
			strcpy(startAddress,operand);
			locctr=hexToDecimal(operand);
		}
		else{
			if(strcmp(opcode,"RESW")==0){
				locctr=hexToDecimal(loc)+atoi(operand)*3;
				if(m+(6*atoi(operand))>69){
					m=70;
				}
				else{
					for(l=0;l<6*atoi(operand);l++)
```

```
                        T[j][k++]=' ';
                T[j][k++]='^';
                m+=6*atoi(operand);
                n=m;
        }
}
else if (strcmp(opcode,"RESB")==0){
        locctr=hexToDecimal(loc)+atoi(operand);
        if(m+(2*atoi(operand))>69){
                m=70;
        }
        else{
                for(l=0;l<2*atoi(operand);l++)
                        T[j][k++]=' ';
                T[j][k++]='^';
                m+=2*atoi(operand);
                n=m;
        }
}
else if(strcmp(opcode,"WORD")==0){
        locctr=hexToDecimal(loc)+3;
        if(m+6>69)
                newT(loc);
        T[j][k]='\0';
        for(l=strlen(operand);l<6;l++)
                strcat(T[j]," ");
        strcat(T[j],operand);
        k+=6;
        m+=6;
        n=m;
        T[j][k++]='^';
}
else if(strcmp(opcode,"BYTE")==0){
        locctr=hexToDecimal(loc)+1;
        if(m+2>69)
                newT(loc);
        T[j][k]='\0';
        for(l=strlen(operand);l<2;l++)
                strcat(T[j]," ");
        strcat(T[j],operand);
        k+=2;
        m+=2;
        n=m;
        T[j][k++]='^';
```

```
                }
                else{
                        f1=fopen("optab.txt","r");
                        if(m+6>69)
                                newT(loc);
                        T[j][k]='\0';
                        while(!feof(f1)){
                                fscanf(f1,"%s%s",opc,mnemo);
                                if(strcmp(opcode,opc)==0){
                                        strcat(T[j],mnemo);
                                        break;
                                }
                        }
                        fclose(f1);
                        if(strcmp("**",operand)==0)
                                strcat(T[j],"0000");
                        else {
                                f2=fopen("symtab.txt","r");
                                while(!feof(f2)){
                                        fscanf(f2,"%s%s",sym,addr);
                                        if(strcmp(operand,sym)==0){
                                                strcat(T[j],addr);
                                                break;
                                        }
                                }
                                fclose(f2);
                        }
                        locctr=hexToDecimal(loc)+3;
                        k+=6;
                        m+=6;
                        n=m;
                        T[j][k++]='^';
                }
        }
}
fclose(fp);
char lenT[10];
sprintf(lenT,"%x",n);
T[j][k]='\0';
T[j][9]=strlen(lenT)==1?'0':lenT[0];
T[j][10]=lenT[1];
char leng[30];
length=locctr-hexToDecimal(startAddress);
sprintf(leng,"%x",length);
```

```
        fp=fopen("record.txt","w");
        fprintf(fp,"H^%s",prgmname);
        for(l=strlen(prgmname);l<6;l++)
                fprintf(fp," ");
        fprintf(fp,"^");
        for(l=strlen(startAddress);l<6;l++)
                fprintf(fp,"0");
        fprintf(fp,"%s^",startAddress);
        for(l=strlen(leng);l<6;l++)
                fprintf(fp,"0");
        fprintf(fp,"%s\n",leng);

        for(l=0;l<=j;l++)
                fprintf(fp,"%s\n",T[l]);

        fprintf(fp,"E^");
        for(l=strlen(startAddress);l<6;l++)
                fprintf(fp,"0");
        fprintf(fp,"%s\n",startAddress);
}
void main(){
        secondpass();
}
```