SS LAB

PASS1:

Pass1.c-

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

struct Line {
    char label[10];
    char opcode[10];
    char operand[10];
};

int isOpcodeValid(char *opcode, FILE *fp2) {
    char code[10], mnemonic[10];
    rewind(fp2);

    while (fscanf(fp2, "%s\t%s", code, mnemonic) == 2) {
        if (strcmp(opcode, code) == 0) {
            return 1;
        }
    }
    return 0;
}

int main() {
    struct Line line;
    int start, length;
    unsigned int locctr;

    FILE *fp1, *fp2, *fp3, *fp4;

    fp1 = fopen("input.txt", "r");
    fp2 = fopen("optab.txt", "r");
    fp3 = fopen("sym.txt", "w");
    fp4 = fopen("op.txt", "w");

    if (fp1 == NULL || fp2 == NULL || fp3 == NULL || fp4 == NULL) {
        printf("file not found");
        return 1;
    }

    while (fscanf(fp1, "%s\t%s\t%s", line.label, line.opcode, line.operand) != EOF) {
        if (strcmp(line.opcode, "START") == 0) {
            start = (int)strtol(line.operand, NULL, 16);
            locctr = (unsigned int)start;
            fprintf(fp4, "\t%s\t%s\t%s\n", line.label, line.opcode, line.operand);
        }

        if (strcmp(line.opcode, "START") != 0) {
            fprintf(fp4, "%X\t%s\t%s\t%s\n", locctr, line.label, line.opcode, line.operand);

            if (strcmp(line.label, "**") != 0 && strcmp(line.opcode, "EQU") != 0) {
                fprintf(fp3, "%s\t%X\n", line.label, locctr);
            }

            if (strcmp(line.opcode, "EQU") == 0) {
                fprintf(fp3, "%s\t%s\n", line.label, line.operand);
            }

            if (isOpcodeValid(line.opcode, fp2))
                {
                locctr += 3;
            }
            else if (strcmp(line.opcode, "WORD") == 0)
                {
                locctr += 3;
            } else if (strcmp(line.opcode, "RESW") == 0)
            {
                locctr += 3 * atoi(line.operand);
            }
            else if (strcmp(line.opcode, "RESB") == 0)
                {
                locctr += atoi(line.operand);
            }
            else if (strcmp(line.opcode, "BYTE") == 0) {
                ++locctr;
            }
            else if (strcmp(line.opcode, "ORG") == 0)
                {
                locctr = (int)strtol(line.operand, NULL, 16);
                fprintf(fp4, "%X\t%s\t%s\t%s\n", locctr, line.label, line.opcode, line.operand);

            }
        }
        printf("%X\t%s\t%s\t%s\n", locctr, line.label, line.opcode, line.operand);
    }

    fclose(fp1);
    fclose(fp2);
    fclose(fp3);
    fclose(fp4);

    return 0;
}
```

Input.txt-

```
**      START  2000
**      LDA    FIVE
**      STA    ALPHA
**      ORG    2050
**      LDCH   CHARZ
**      STCH   C1
**      ORG    3000
A       EQU    2000
```

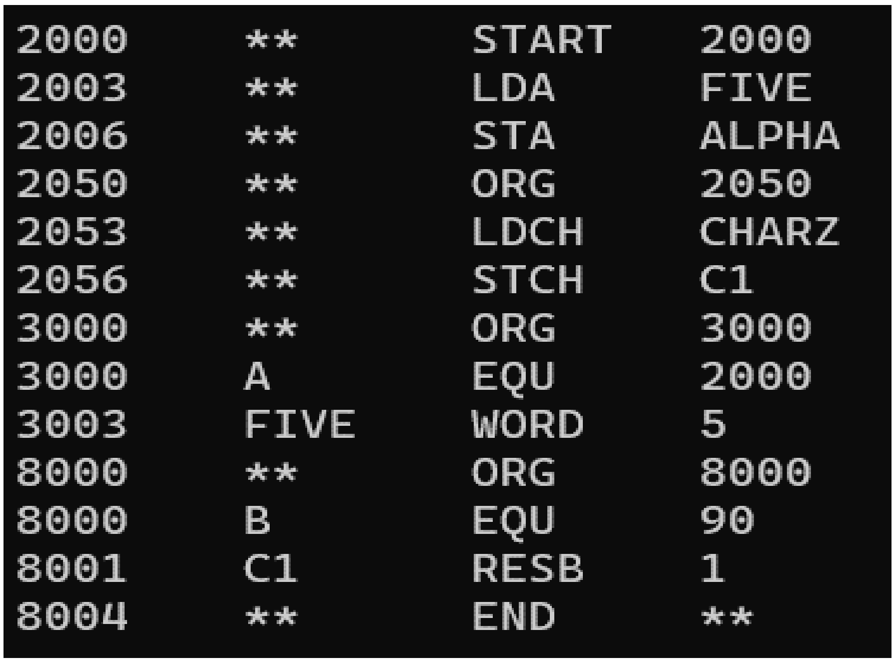
```
FIVE     WORD  5
**       ORG     8000
B        EQU    90
C1       RESB    1
**       END     **
```

Optab.txt-

```
LDA     03
STA     0f
LDCH    53
STCH    57
END     *
```

Sym.txt and Op.txt are created

Terminal O/P-



```
2000    **      START   2000
2003    **      LDA     FIVE
2006    **      STA     ALPHA
2050    **      ORG     2050
2053    **      LDCH    CHARZ
2056    **      STCH    C1
3000    **      ORG     3000
3000    A       EQU     2000
3003    FIVE    WORD    5
8000    **      ORG     8000
8000    B       EQU     90
8001    C1      RESB    1
8004    **      END     **
```

PASS2:

Pass2.c-

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

char label[50], opcode[50], operand[50];
char symbol[50];
char value[10];
char mnemonic[50];
char operand_address[5];
char opcode_address[10];
char locctr[10];
int length = 0;
char text_record[100];
char object_code[20];
char integer[20];
```

```
int cur_length = 0;
int is_last = 1;
int starting_address;
int STARTING_ADDR = 0;
int i = 0;


void get_length()
{
   FILE *fp4 = fopen("length.txt", "r");
   if(fp4== NULL)
      printf("Error Opening length.txt\n");

   fscanf(fp4, "%d", &length);
}

int check_indexed()
{
   int is_indexed = 0;
   char *p = strtok (operand, " ,");
   char *array[3];
   int len = 0;

   while (p != NULL)
   {
      array[len++] = p;
      p = strtok (NULL, " ,");
   }

   if((len == 2) && (strcmp(array[1], "X") == 0)){
      strcpy(operand, array[0]);
      is_indexed = 1;
   }

   return is_indexed;
}

int search_symtab()
{
            FILE *fp5=fopen("symtab.txt","r");
   if(fp5== NULL)
      printf("Error Opening symtab.txt\n");

   int found = 0;
   strcpy(operand_address, "0000");
   while(!feof(fp5))
   {
            fscanf(fp5,"%s\t%s",symbol,value);
      if(strcmp(symbol,operand)==0)
      {
            strcpy(operand_address, value);
          found = 1;
            break;
      }
   }
```

```
    fclose(fp5);
    if(!found)
        printf("%s --- Error! - undefined symbol\n", operand);
    return found;
}

int search_optab()
{
            FILE *fp6=fopen("optab.txt","r");
    if(fp6== NULL)
        printf("Error Opening optab.txt\n");

            int found = 0;
    strcpy(opcode_address, "0");
    while(!feof(fp6))
    {
            fscanf(fp6,"%s\t%s",mnemonic,value);
        if(strcmp(mnemonic, opcode)==0)
        {
            strcpy(opcode_address, value);
            found = 1;
            break;
        }
    }

    fclose(fp6);
    return found;
}

void pass2()
{
    FILE *fp1;
    fp1 = fopen("intermediate.txt", "r");
    FILE *fp2 = fopen("output.txt", "w");
    FILE *fp3 = fopen("object_program.txt", "w");

    if(fp1== NULL)
        printf("Error Opening intermediate.txt\n");
    if(fp2== NULL)
        printf("Error Opening output.txt\n");
    if(fp3== NULL)
        printf("Error Opening object program.txt\n");

    char delimit[]=" \t\r\n";
    int start;
    char line[100];
    size_t len = 100 * sizeof(char);

    while ((fgets(&line, &len, fp1)) != NULL)
    {
        int len = 0;
        strcpy(label, " ");
        strcpy(opcode, " ");
        strcpy(operand, " ");
        char *p = strtok (line, delimit);
        char *array[5];
```

```
strcpy(object_code, "");

while (p != NULL)
{
   array[len++] = p;
   p = strtok (NULL, delimit);
}
if(len == 1)
{
   strcpy(opcode, array[0]);
}
else if(len == 2)
{
   strcpy(locctr, array[0]);
   strcpy(opcode, array[1]);
}
else if(len == 3)
{
   strcpy(locctr, array[0]);
   strcpy(opcode, array[1]);
   strcpy(operand, array[2]);
}
else if(len == 4)
{
   strcpy(locctr, array[0]);
   strcpy(label, array[1]);
   strcpy(opcode, array[2]);
   strcpy(operand, array[3]);
}
if(strcmp(opcode, "END")==0)
   break;

if(strcmp(opcode, "START")==0)
{
   fprintf(fp2, "%s\t%s\t%s\t%s\n", locctr, label, opcode, operand);
   STARTING_ADDR = starting_address = (int)strtol(operand, NULL, 16);
   get_length();
   for(i = 0; i <= 6 - strlen(label); i++)
      strcat(label, " ");
   fprintf(fp3,"H^%s^%06x^%06x\n", label, starting_address, length);
   fprintf(fp3, "T^");
   fprintf(fp3, "%06x^", starting_address);

   continue;
}
if((!strcmp(label, " ")==0) || (!strcmp(opcode, " ")==0) || (!strcmp(operand, " ")==0))
{
   if(search_optab())
   {
      if(!(strcmp(operand, " ") == 0))
      {
         int is_indexed = check_indexed();

         search_symtab();

         if(is_indexed)
```

```
            {
                strcat(operand, ", X");
                int num = (int)strtol(operand_address, NULL, 16);
                num = num | (1 << 15);
                sprintf(operand_address, "%04x", num);
            }
        }
        else
            strcpy(operand_address, "0000");

        strcpy(object_code, strcat(opcode_address, operand_address));
        fprintf(fp2, "%s\t%s\t%s\t%s\t%s\n", locctr, label, opcode, operand, object_code);

        cur_length = (int)strtol(locctr, NULL, 16) - starting_address;
    }
    else if((strcmp(opcode, "BYTE") == 0) || (strcmp(opcode, "WORD") == 0))
    {
        if(strcmp(opcode, "WORD") == 0)
        {
            strcpy(object_code, "");
            sprintf(integer, "%06x", atoi(operand));
            strcpy(object_code, integer);
            fprintf(fp2, "%s\t%s\t%s\t%s\t%s\n", locctr, label, opcode, operand, object_code);
        }
        else
        {
            fprintf(fp2, "%s\t%s\t%s\t%s\t", locctr, label, opcode, operand);

            strcpy(object_code, "");
            if(operand[0] == 'C' || operand[0] == 'c')
            {
                for(i = 2; i < strlen(operand) - 1; i++){
                    sprintf(integer, "%x", operand[i]);
                    strcat(object_code, integer);
                }
                fprintf(fp2, "%s\n", object_code);
            }
            else
            {
                for(i = 2; i < strlen(operand) - 1; i++)
                {
                    sprintf(integer, "%c", operand[i]);
                    strcat(object_code, integer);
                }
                fprintf(fp2, "%s\n", object_code);
            }

        }
    }
    else
    {
        fprintf(fp2, "%s\t%s\t%s\t%s\n", locctr, label, opcode, operand);
    }
    if(((int)strtol(locctr, NULL, 16) - starting_address) < 30)
    {
        if(!(strcmp(object_code, "")) == 0)
```

```
                {
                    strcat(text_record, "^");
                    strcat(text_record, object_code);
                }
                else if(is_last)
                {
                    cur_length = (int)strtol(locctr, NULL, 16) - starting_address;
                    is_last = 0;
                }
            }
            else
            {
                cur_length = (int)strtol(locctr, NULL, 16) - starting_address;
                fprintf(fp3, "%02x%s\n", cur_length, text_record);
                strcpy(text_record, "^");
                strcat(text_record, object_code);
                starting_address = (int)strtol(locctr, NULL, 16);
                fprintf(fp3, "T^");
                fprintf(fp3, "%06x^", starting_address);
                is_last = 1;
            }
        }
    }
    fprintf(fp3, "%02x%s\n", cur_length, text_record);
    starting_address = (int)strtol(locctr, NULL, 16);

    // End record
    fprintf(fp3, "E^%06x\n", STARTING_ADDR);

    fclose(fp1);
    fclose(fp2);
    fclose(fp3);

    printf("Completed Pass 2\n");
}

void show_output()
{
    FILE *fp8 = fopen("output.txt", "r");
    char locctr[50];

    if(fp8== NULL)
        printf("Error Opening output.txt\n");

    printf("\n---------Output File---------\n");

    char line[100];
    size_t len = 100 * sizeof(char);

    while ((fgets(&line, &len, fp8)) != NULL)
        printf("%s", line);

    fclose(fp8);
}

int main()
```

```
{
        pass2();
    show_output();

        return 0;
}
```

Intermediate.txt-

```
** 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'EOF'
2019 C1 RESB 1
2020 ** END **
```

Length.txt-

```
25
```

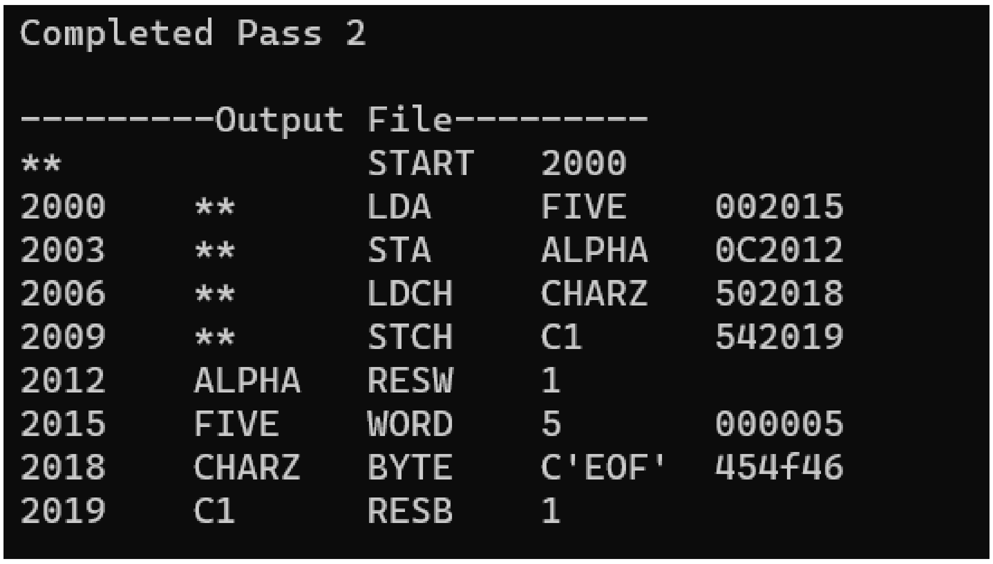
Optab.txt-

```
LDA 00
STA 0C
LDCH 50
STCH 54
END *
```

Symtab.txt-

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

Output-
Output.txt and object program.txt are created.
Terminal O/P-



```
Completed Pass 2

---------Output File---------
**              START   2000
2000    **      LDA     FIVE    002015
2003    **      STA     ALPHA   0C2012
2006    **      LDCH    CHARZ   502018
2009    **      STCH    C1      542019
2012    ALPHA   RESW    1
2015    FIVE    WORD    5       000005
2018    CHARZ   BYTE    C'EOF'  454f46
2019    C1      RESB    1
```

ABSOLUTE LOADER-

Absolute.c-

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main()
{
   FILE *f1,*fp2;
   f1=fopen("object1.txt","r+");
   fp2=fopen("output2.txt","r+");
   char buffer[1000];
   char b[10],c[10],d[10],e[10];
   char a[10]="H";
   char temp[100];
   unsigned long temp1,temp2,temp3;
   int i;
   int len;
   int j;
   char z[10]="E";
   printf("memory locations object codes");
   fprintf(fp2, "Memory values  \t\t\t  contents");
   printf("\n");
   for(i=0;i<=5;i++)
   {
      fscanf(f1,"%s",buffer);
      if(strcmp(buffer,a)== 0)
      {
         fscanf(f1,"%s %s %s",b,c,d);
         temp1=strtoul(c, NULL, 16);
      }

      else
      {
         fscanf(f1,"%s %s",b,d);
         temp2=strtoul(d, NULL, 16);
         len=temp2/3;
         temp3=temp1-3096;
         printf("%d",temp3);
         fprintf(fp2, "\n");
         fprintf(fp2, "%d", temp3);
         for(j=0;j<len;j++)
         {
            fscanf(f1,"%s",c);
            printf("\t");
            printf("%s",c);
            fprintf(fp2, "\t%s", c);
            printf("\t");
         }
         fprintf(fp2, "\n");
         printf("\n");
         temp1=temp1+temp2;
      }

   }
}
```

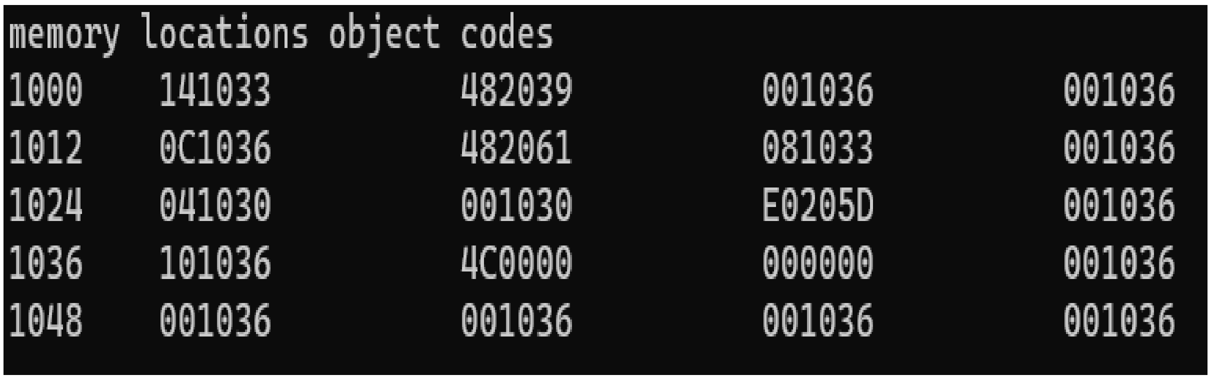
Object1.txt-

```
H COPY 001000 00107A
T 001000 0C 141033 482039 001036 001036
T 00101E 0C 0C1036 482061 081033 001036
T 001047 0C 041030 001030 E0205D 001036
T 001077 0C 101036 4C0000 000000 001036
```

Output.txt-

Create this text doc and keep.

Terminal O/P-



```
memory locations object codes
1000    141033          482039          001036          001036
1012    0C1036          482061          081033          001036
1024    041030          001030          E0205D          001036
1036    101036          4C0000          000000          001036
1048    001036          001036          001036          001036
```

Relocatable_loader-

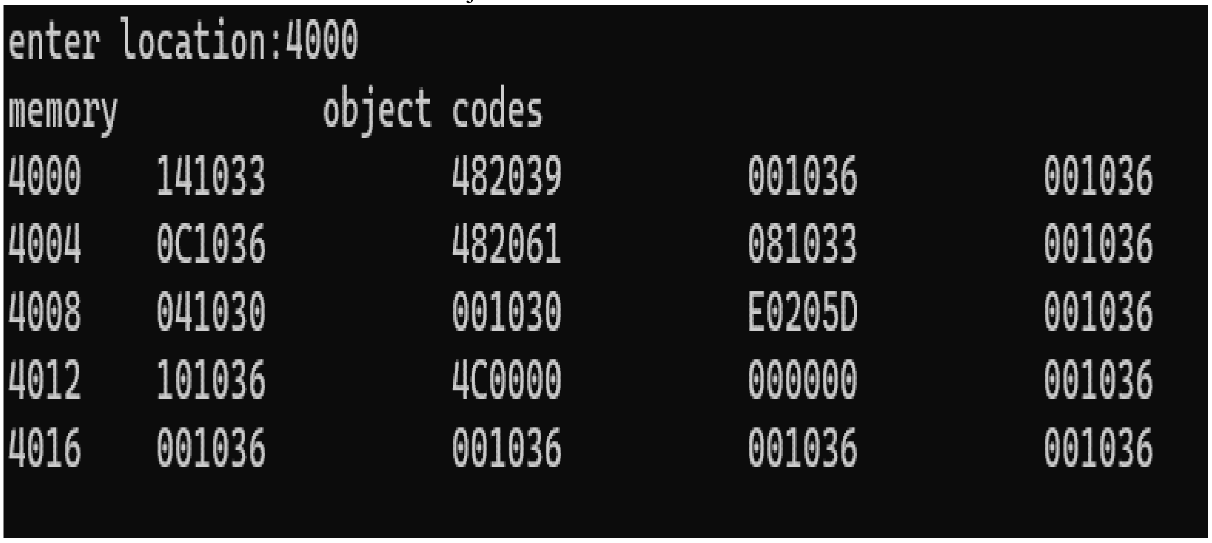
Relocatable.c-

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
void main()
{
   FILE *f1,*fp2;
   f1=fopen("object1.txt","r+");
   fp2=fopen("output2.txt","r+");
   char buffer[1000];
   char b[10],c[10],d[10],e[10];
   char a[10]="H";
   char temp[100];
   unsigned long temp1,temp2,temp3;
   int i;
   int len;
   int j;
   char z[10]="E";
   printf("enter location:");
   scanf("%d",&temp1);
   printf("memory \t\t object codes");
   fprintf(fp2, "Memory values  \t\t\t  contens");
   printf("\n");
   for(i=0;i<=5;i++)
   {
      fscanf(f1,"%s",buffer);
      if(strcmp(buffer,a)== 0)
      {
            fscanf(f1,"%s %s %s",b,c,d);
        }

        else
        {
            fscanf(f1,"%s %s",b,d);
            temp2=strtoul(d, NULL, 16);
            len=temp2/3;
            printf("%d",temp1);
            fprintf(fp2, "\n");
            fprintf(fp2, "%d", temp1);
            for(j=0;j<len;j++)
            {
                fscanf(f1,"%s",c);
                printf("\t");
                printf("%s",c);
                fprintf(fp2, "\t%s", c);
                printf("\t");
            }
            fprintf(fp2, "\n");
            printf("\n");
            temp1=temp1+len;
        }

    }
}
```

Object1.txt-

```
H COPY 001000 00107A
T 001000 0C 141033 482039 001036 001036
T 00101E 0C 0C1036 482061 081033 001036
T 001047 0C 041030 001030 E0205D 001036
T 001077 0C 101036 4C0000 000000 001036
```

Create the output2.txt txt doc -
 Terminal O/P-
Enter the desired location to relocate the object code.



```
enter location:4000
memory          object codes
4000    141033          482039          001036          001036
4004    0C1036          482061          081033          001036
4008    041030          001030          E0205D          001036
4012    101036          4C0000          000000          001036
4016    001036          001036          001036          001036
```

Loader pass1:

Loader pass1.c:
```
#include<stdio.h>
#include<string.h>
struct estab
{
  char csname[10];
  char extsym[10];
  int address;
  int length;
}es[20];
void main()
{
	char input[10],name[10],symbol[10],ch;  int count=0,progaddr,csaddr,add,len;
	FILE *fp1,*fp2;
	fp1=fopen("input1.txt","r");
	fp2=fopen("ESTAB.txt","w");
	printf("\n\nEnter the address where the program has to be loaded : ");
	scanf("%x",&progaddr); // TAKING THE PROGRAM ADDRESS FROM THE USER,GENERALLY
IT IS DONE BY THE OS
	csaddr=progaddr;
	fscanf(fp1,"%s",input);
	while(strcmp(input,"END")!=0)
	{
		if(strcmp(input,"H")==0)
		{
			fscanf(fp1,"%s",name);
			strcpy(es[count].csname,name);
			strcpy(es[count].extsym,"  ");
			fscanf(fp1,"%x",&add);
			es[count].address=add+csaddr;
			fscanf(fp1,"%x",&len);
			es[count].length=len;
			fprintf(fp2,"%s ** %x %x\n",es[count].csname,es[count].address,es[count].length);
			count++;
		}
		else if(strcmp(input,"D")==0)
		{
			fscanf(fp1,"%s",input);
			while(strcmp(input,"R")!=0)
			{
				strcpy(es[count].csname,"  ");
				strcpy(es[count].extsym,input);
				fscanf(fp1,"%x",&add);
				es[count].address=add+csaddr;
				es[count].length=0;
				fprintf(fp2,"** %s %x\n",es[count].extsym,es[count].address);
				count++;
				fscanf(fp1,"%s",input);
			}
			csaddr=csaddr+len;
		}
		else if(strcmp(input,"T")==0)
		{
			while(strcmp(input,"E")!=0)
```

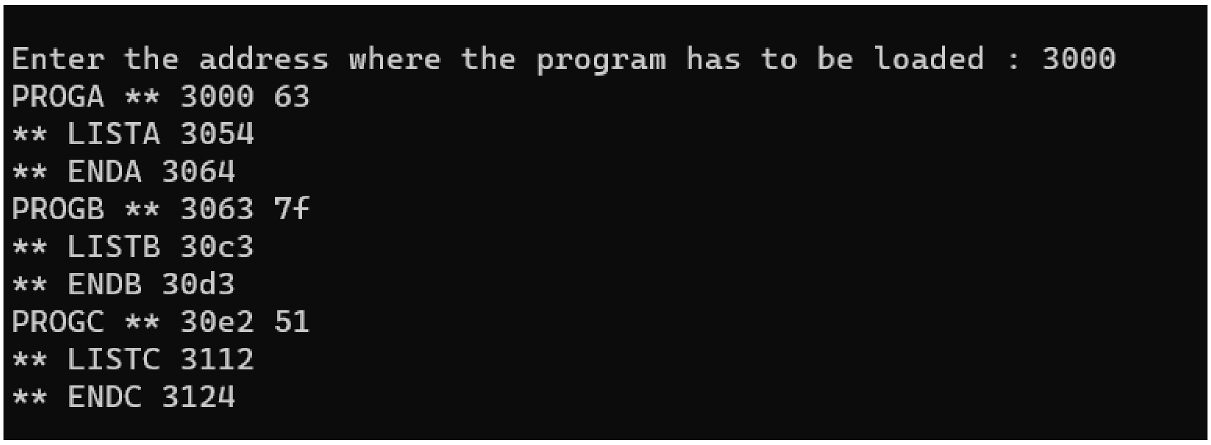
```
                    fscanf(fp1,"%s",input);
               }
               fscanf(fp1,"%s",input);
          }
  fclose(fp1);
  fclose(fp2);
  fp2=fopen("ESTAB.txt","r");
  ch=fgetc(fp2);
  while(ch!=EOF)
  {
   printf("%c",ch);
   ch=fgetc(fp2);
  }
  fclose(fp2);
}
```

Input.txt:

```
H PROGA 000000 000063
D LISTA 000054 ENDA 000064
R LISTB ENDB LISTC ENDC
T 000020 0A 03201D 77100004 050014
T 000054 0F 100014 000008 004051 000004 100000
M 000024 05 +LISTA
M 000054 06 +LISTC
M 000060 06 +LISTB
M 000060 06 -LISTA
E 000020

H PROGB 000000 00007F
D LISTB 000060 ENDB 000070
R LISTA LISTC ENDY
T 000036 0B 03100000 772027 05100000
T 000070 0F 100000 000008 004051 000004 100060
M 000037 05 +LISTA
M 00003E 05 -LISTA
M 000070 06 -LISTA
M 000070 06 +LISTC
M 00007C 06 +PROGB
M 00007C 06 -LISTA
E 000000

H PROGC 000000 0000051
D LISTC 000030 ENDC 000042
R LISTA LISTB ENDB
T 000018 0C 03100000 77100004 05100000
T 000042 0F 100030 000008 004051 000004 100000
M 00001D 05 +LISTB
M 000021 05 -LISTA
M 000042 06 -LISTA
M 000042 06 +PROGC
M 00004E 06 +LISTB
M 00004E 06 -LISTA
E
END
```

ESTAB: create ESTAB.txt txt doc

Terminal O/P:
This is stored in ESTAB txt file.



```
Enter the address where the program has to be loaded : 3000
PROGA ** 3000 63
** LISTA 3054
** ENDA 3064
PROGB ** 3063 7f
** LISTB 30c3
** ENDB 30d3
PROGC ** 30e2 51
** LISTC 3112
** ENDC 3124
```

Loader pass2:

Loader pass2.c:

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct exttable
{
  char cextsym[20], extsym[20];
  int address,length;
}estab[20];

struct objectcode
{
 unsigned char code[15];
 int add;
}obcode[500];

void main()
{
 char temp[10];
 FILE *fp1,*fp2,*fp3;
 int i,j,x,y,pstart,exeloc,start,textloc,loc,textlen,length,location,st,s;
 int n=0,num=0,inc=0,count=0,record=0,mloc[30],mlen[30];
 signed long int newadd;
 char operation,lbl[10],input[10],label[50][10],opr[30],ch,*add1,address[10];
 fp1=fopen("input1.txt","r");
 fp2=fopen("ESTAB.txt","r");
 fp3=fopen("OUTPUT2.txt","w");
 while(!feof(fp2))
 {
             fscanf(fp2,"%s %s %x %x", estab[num].cextsym, estab[num].extsym, &estab[num].address,
                                   &estab[num].length);
             num++;
 }
 exeloc=estab[0].address;
 loc=exeloc;
 start=loc;
```

```
st=start;
while(!feof(fp1))
{
        fscanf(fp1,"%s",input);
        if(strcmp(input,"H")==0)
        {
                fscanf(fp1,"%s",input);
                for(i=0;i<num;i++)
                if(strcmp(input,estab[i].cextsym)==0)
                {
                        pstart=estab[i].address;
                        break;
                }
                while(strcmp(input,"T")!=0)
                        fscanf(fp1,"%s",input);
        }
        do
        {

                if(strcmp(input,"T")==0)
                {
                        fscanf(fp1,"%x",&textloc);
                        textloc=textloc+pstart;
                        for(i=0;i<(textloc-loc);i++)
                        {
                                strcpy(obcode[inc].code,"..");
                                obcode[inc++].add=start++;
                        }
                        fscanf(fp1,"%x",&textlen);
                        loc=textloc+textlen;
                }
                else if(strcmp(input,"M")==0)
                {
                        fscanf(fp1,"%x",&mloc[record]);
                        mloc[record]=mloc[record]+pstart;
                        fscanf(fp1,"%x",&mlen[record]);
                        fscanf(fp1,"%s",label[record++]);
                }
                else
                {
                        length=strlen(input);
                        x=0;
                        for(i=0;i<length;i++)
                        {
                                obcode[inc].code[x++]=input[i];
                                if(x>1)
                                {
                                        obcode[inc++].add=start++;
                                        x=0;
                                }
                        }
                }
                fscanf(fp1,"%s",input);
        }while(strcmp(input,"E")!=0);
        if(strcmp(input,"E")==0)
        fscanf(fp1,"%s",input);
```

```
}
            for(n=0;n<record;n++)
            {
            operation=label[n][0];
            length=strlen(label[n]);
            for(i=1;i<length;i++)
            {
                        lbl[i-1]=label[n][i];
            }
            lbl[length-1]='\0';
            length=0;
            strcpy(address,"\0");
            location=mloc[n]-exeloc;
            loc=location;
            count=0;
            while(length<mlen[n])
            {
                        strcat(address,obcode[location++].code);
                        count++;
                        length+=2;
            }
            for(i=0;i<num;i++)
            {
                        if(strcmp(lbl,estab[i].cextsym)==0)
                        break;
                        if(strcmp(lbl,estab[i].extsym)==0)
                        break;
            }
            switch(operation)
            {
                        case '+':
                                                newadd=strtol(address,&add1,16)+(long int)estab[i].address;
                                                break;
                        case '-':
                                                newadd=strtol(address,&add1,16)-(long int)estab[i].address;
                                                break;
            }
 ltoa(newadd,address,16);
 x=0; y=0;
 while(count>0)
 {
  obcode[loc].code[x++]=address[y++];
  if(x>1)
  {
                        x=0; loc++;
                        count--;
  }
 }
 }
count=0;
n=0;
s=st-16;
fprintf(fp3,"%x\t",s);
for(i=1;i<=16;i++)
{
 fprintf(fp3,"xx");
 if(i==4||i==8||i==12)
 {
  fprintf(fp3,"\t");
 }
}
fprintf(fp3,"\n\n%x\t",obcode[0].add);
for(i=0;i<inc;i++)
{
            fprintf(fp3,"%s",obcode[i].code);
            n++;
            if(n>3)
            {
                        fprintf(fp3,"\t");
                        n=0;
      count++;
            }
            if(count>3)
            {
                        fprintf(fp3,"\n\n%x\t",obcode[i+1].add);
                        count=0;
            }
 }
fclose(fp1);
fclose(fp2);
fclose(fp3);
printf("\n\t*** PASS TWO OF A LINKING LOADER ***\n");
printf("\nThe contents of the output file :");
printf("\n---------------------------------------------------------------");
printf("\nAddress\t\t\t\tContents");
printf("\n---------------------------------------------------------------\n");
fp3=fopen("OUTPUT2.txt","r");
ch=fgetc(fp3);
while(ch!=EOF)
{
 printf("%c",ch);
 ch=fgetc(fp3);
}
fclose(fp3);
}
```

Input1.txt:

```
H PROGA 000000 000063
D LISTA 000054 ENDA 000064
R LISTB ENDB LISTC ENDC
T 000020 0A 03201D 77100004 050014
T 000054 0F 100014 000008 004051 000004 100000
M 000024 05 +LISTA
M 000054 06 +LISTC
M 000060 06 +LISTB
M 000060 06 -LISTA
E 000020

H PROGB 000000 00007F
D LISTB 000060 ENDB 000070
R LISTA LISTC ENDY
T 000036 0B 03100000 772027 05100000
```

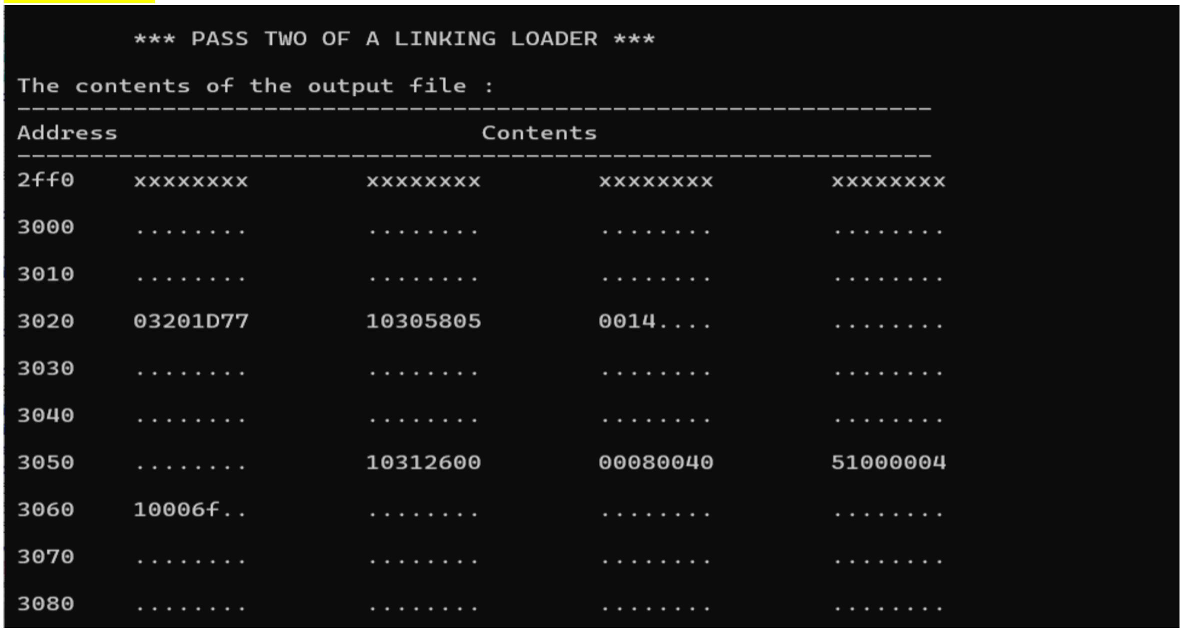
T 000070 0F 100000 000008 004051 000004 100060
M 000037 05 +LISTA
M 00003E 05 -LISTA
M 000070 06 -LISTA
M 000070 06 +LISTC
M 00007C 06 +PROGB
M 00007C 06 -LISTA
E 000000

H PROGC 000000 0000051
D LISTC 000030 ENDC 000042
R LISTA LISTB ENDB
T 000018 0C 03100000 77100004 05100000
T 000042 0F 100030 000008 004051 000004 100000
M 00001D 05 +LISTB
M 000021 05 -LISTA
M 000042 06 -LISTA
M 000042 06 +PROGC
M 00004E 06 +LISTB
M 00004E 06 -LISTA
E
END

ESTAB: This is generated by loader pass1.

PROGA ** 3000 63
** LISTA 3054
** ENDA 3064
PROGB ** 3063 7f
** LISTB 30c3
** ENDB 30d3
PROGC ** 30e2 51
** LISTC 3112
** ENDC 3124

OUTPUT2.txt: Create this txt
Terminal O/P:


```
        *** PASS TWO OF A LINKING LOADER ***

The contents of the output file :
---------------------------------------------------------------
Address                         Contents
---------------------------------------------------------------
2ff0    xxxxxxxx        xxxxxxxx        xxxxxxxx        xxxxxxxx

3000    ........        ........        ........        ........

3010    ........        ........        ........        ........

3020    03201D77        10305805        0014....        ........

3030    ........        ........        ........        ........

3040    ........        ........        ........        ........

3050    ........        10312600        00080040        51000004

3060    10006f..        ........        ........        ........

3070    ........        ........        ........        ........

3080    ........        ........        ........        ........
```