|  |  |
| --- | --- |
| **Ex.no.6** | **PASS ONE OF DIRECT LINKING LOADER** |
| 24.10.20 |

### Header

#include <stdio.h>

#include <string.h>

#include <stdio.h>

FILE \*fp1, \*fp2;

typedef struct ESTAB{

char csname[10];

char address[7];

char length[5];

char symname[10];

}ESTAB;

void externalSymbolTableGeneration(char[],char [],char[]);

void displayTable(char []);

### Implementation

#include "head.h"

void externalSymbolTableGeneration(char fname[],char fname1[], char csaddr[]){

ESTAB es;

int i,len,j;

int add,csadd;

char temp[100];

char address[7],csname[10],length[5],sname[10],c,rtype[2];

fp1=fopen(fname,"r");

fp2=fopen(fname1,"a");

while(!feof(fp1)){

fscanf(fp1,"%[^\n]",temp);

c=fgetc(fp1);

if(feof(fp1)) break;

len=strlen(temp);

rtype[0]=temp[0];

rtype[1]='\0';

if(strcmp(rtype,"H")==0){

i=2;

j=0;

while(temp[i]!='^'&&i<len){

csname[j]=temp[i];

i++;

j++;

}

csname[j]='\0';

i=i+1;

j=0;

while(temp[i]!='^'&&i<len){

address[j]=temp[i];

i++;

j++;

}

address[j]='\0';

i=i+1;

j=0;

while(i<len)

{

length[j]=temp[i];

i++;

j++;

}

length[j]='\0';

strcpy(es.csname,csname);

strcpy(es.symname,"\0");

strcpy(es.length,length);

sscanf(csaddr,"%04x",&csadd);

sscanf(address,"%06x",&add);

add=add+csadd;

sprintf(address,"%04X",add);

strcpy(es.address,address);

strcpy(temp,"\0");

fwrite(&es,sizeof(es),1,fp2);

}

if(strcmp(rtype,"D")==0)

{

i=2;

j=0;

while(temp[i]!='^'&&i<len)

{

sname[j]=temp[i];

i++;

j++;

}

sname[j]='\0';

i=i+1;

j=0;

while(i<len){

address[j]=temp[i];

j++;

i++;

}

address[j]='\0';

strcpy(es.csname,"\0");

strcpy(es.length,"\0");

strcpy(es.symname,sname);

sscanf(csaddr,"%04x",&csadd);

sscanf(address,"%06x",&add);

add=add+csadd;

sprintf(address,"%04X",add);

strcpy(es.address,address);

strcpy(temp,"\0");

fwrite(&es,sizeof(es),1,fp2);

}

}

sscanf(csaddr,"%04x",&csadd);

sscanf(length,"%04x",&add);

csadd=csadd+add;

sprintf(csaddr,"%04x",csadd);

fclose(fp1);

fclose(fp2);

}

void displayTable (char fname[]){

ESTAB es;

fp1=fopen(fname,"r");

printf("\nControlSection\tSymname\tAddress\tLength\n");

while(!feof(fp1)){

fread(&es,sizeof(es),1,fp1);

if(feof(fp1)) break;

printf("\n%s\t\t%s\t%s\t%s\n",es.csname,es.symname,es.address,es.length);

}

fclose(fp1);

}

### Application

#include "head.h"

main(int argc, char \*argv[]){

int i;

char pgmaddr[5],csaddr[5];

printf("Enter the program address\n");

scanf("%s",pgmaddr);

strcpy(csaddr,pgmaddr);

i=2;

fp1=fopen(argv[1],"w");

fclose(fp1);

while(i<argc){

externalSymbolTableGeneration(argv[i],argv[1],csaddr);

i++;

}

displayTable(argv[1]);

}

### Input file 1

H^PROGA^000000^0057

D^LISTA^0040

D^ENDA^0054

R^LISTB^ENDb^LISTC^ENDC

T^0020^0A^03201D^77300004^050014

T^000054^03^000014

M^000024^05^+LISTB

M^000054^06^+LISTC

E^000020

### Input file 2

H^PROGB^000000^0073

D^LISTB^000060

D^ENDB^000070

R^LISTA^ENDA^LISTC^ENDC

T^000036^0A^03100000^772027^05100000

T^000070^03^000000

M^000037^05^+LISTA

M^00003E^05^+ENDA

M^00003E^05^-LISTA

M^000070^06^-LISTA

M^000070^06^+LISTC

E^000036

### Input file 3

H^PROGC^000000^0045

D^LISTC^000030

D^ENDC^0000042

R^LISTA^ENDA^LISTB^ENDB

T^000042^03^000030

M^000019^05^+LISTA

M^00001D^05^+LISTB

M^000021^05^-LISTA

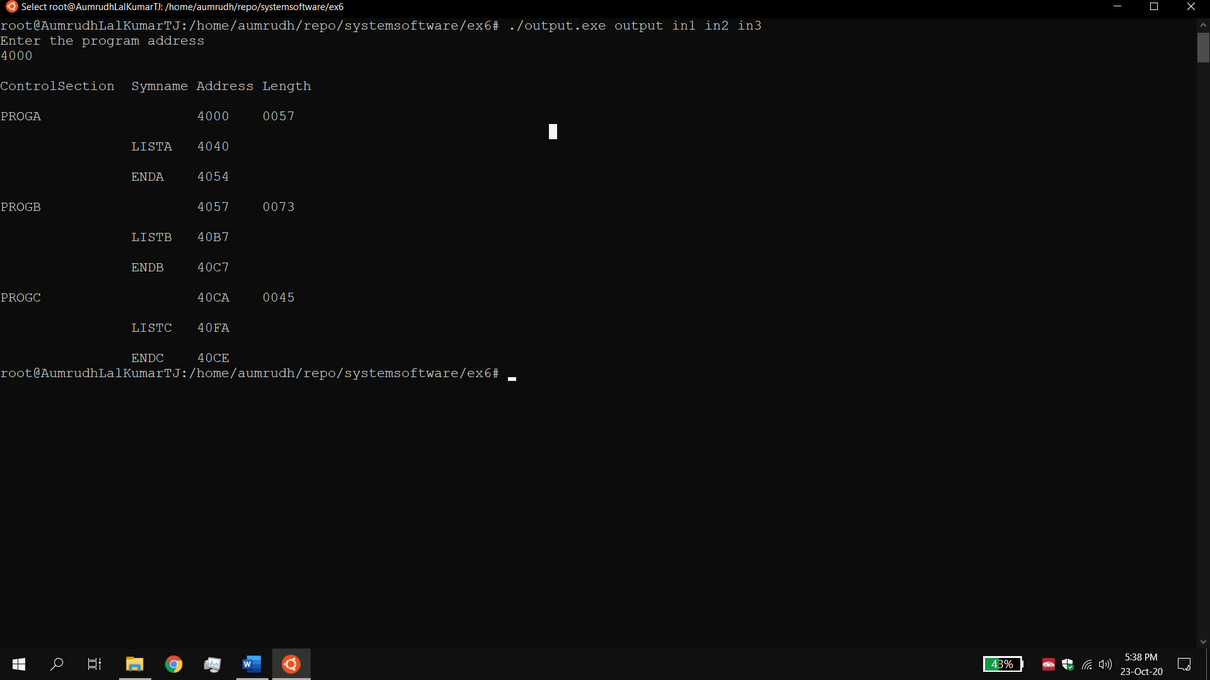
M^000042^06^+ENDA

M^000042^06^-LISTA

M^000042^06^+PROGC

E^000018

## Output



### Result

The pass 1 of linking loader which produces load map as output when given input programs is done successfully.