과목 명: 시스템프로그래밍

담당 교수 명: 김 지 환

<<Assignment 2>>

**서강대학교 컴퓨터학과**

**[학번]20141556**

**[이름]이성문**

목 차

1. 프로그램 개요 3

2. 프로그램 설명 3

2.1 프로그램 흐름도 3

3. 모듈 정의 3

3.1 모듈 이름 : main() 3

3.1.1 기능 3

3.1.2 사용 변수 3

3.2 모듈 이름: display\_instructions() 3

3.2.1 기능 3

3.2.2 사용 변수 3

3.3 모듈이름: get\_data() 4

3.3.1 기능 4

3.3.2 사용변수 4

3.4 모듈이름: display\_report() 4

3.4.1 기능 4

3.4.2 사용변수 4

3.5 모듈이름: continue\_function() 4

3.5.1 기능 4

3.5.2 사용변수 4

4. 전역 변수 정의 4

4.1 long income[100] 4

4.2 int month[100], day[100], year[100] 4

4.3 int ctr 4

4.4 int cont 5

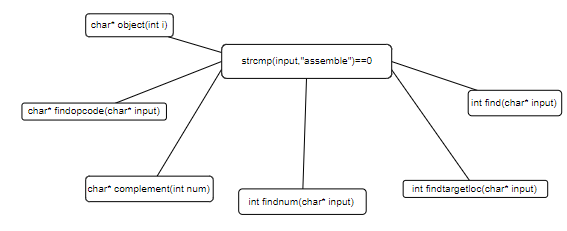
# 프로그램 개요

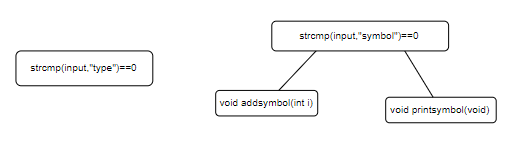
프로젝트 #1에서 구현한 셀(shell)에 assemble 기능을 추가하는 프로그램이다. SIC/XE의 assembly program source 파일을 입력 받아서 object파일을 생성하고, 어셈블리 과정 중 생성된 symbol table과 결과물인 object 파일을 볼 수 있는 기능을 제공한다.

# 프로그램 설명

## 프로그램 흐름도

그림 1> 프로그램 흐름도





중심이 되는 strcmp는 main()에 존재하며 위와 같이 여러 함수들을 호출해 자료구조안에 어셈블 파일 안의 자료들을 저장한다. 위 함수들 parameter중에 int i는 자료구조 전역 구조체 배열의 인덱스를 가리킨다. Input은 보통 mnemonic을 뜻한다. Assemble , type, symbol 모두 main안에 존재한다. 필자는 이 프로그램 작성을 위해 Linked List 자료구조를 사용하였고 알고리즘은 주로 if문과 while,for문을 통해 예외를 처리하며 대체적으로 asm파일 한줄씩 자료구조안에 저장하며 그중 mnemonic과 target이 한 개가 아닐 경우 등등 예외를 처리하며 자료구조안에 저장하고 다 저장됐을 경우 파일 출력을 하면서 object code를 함수를 통해 생성하며 파일 출력을 하였다.

# 모듈 정의

## 모듈 이름 : main()

### 기능

Input과 strcmp함수를 통해 assemble인지 type인지 symbol인지 입력받고 예외는 오류처리한다. assemble에서는 파일 입출력을 통해 2\_5.asm 파일을 입력받아 sou라는 전역변수 자료구조(구조체 배열)안에 반복문을 통해 저장하며 다시 또 반복문을 통해 정해진 규칙에 따라 lst파일과 obj파일을 생성한다. 그리고 type이 입력될 경우 filename에 해당하는 파일을 읽어들여 화면에 출력해준다. 없을 경우 에러메세지를 출력한다. Symbol이 입력될 경우 assemble 명령어가 수행된 후 symbol table에 저장된 symbol들을 정해진 형식에 맞게 출력해준다. Table이 비어있을 경우 오류메세지를 출력해준다.

### 사용 변수

FILE\* q : lst파일을 생성해주는 파일 포인터 변수

FILE\* r : obj파일을 생성해주는 파일 포인터 변수

Typedef struct asmm\* sou : asm 파일에서 자료들을 입력받아 저장하는 구조체 배열이다.

{

Line: lst파일에 새길 라인 넘버이다.

Loc : 그 명령어의 location 값이다

One : symbol 위치의 문자열

Two : Mnemonic 위치의 문자열

Three : Target 위치의 문자열

Four : ‘,’가 존재할경우 두번째 Target의 문자열

Comma : ‘,’가 존재할경우 ‘,’저장

Junk: 명령어가 아닌 주석일 경우 주석 저장

Char\* ob : objectcode 이다

}

Int flag : 오류 체크 플래그이다.

Int length : obj파일 생성시 필요한 최대 길이를 잴 때 쓰는 변수

Int lengthend : 한줄의 길이

Int endindex : 한 줄의 끝의 구조체의 인덱스

## 모듈 이름: find(char\* input)

### 기능

문자열 input을 Hash table에서 찾아서 있다면 1을 리턴, 없으면 0을 리턴해준다.

### 사용 변수

Char\* input : 파라미터로 전달되는 문자열

Table : Hash table , mnemonic들이 저장되있다.

## 모듈이름: object(int i)

### 기능

인덱스 i에 해당하는 구조체 배열 (sou+i)에 저장된 데이터를 대상으로 objectcode를 만들어 리턴해준다. Opcode를 찾고 몇 형식인지 알아내고 비트 n,I,x,b,p,e 값을 알아내고 displace를 계산해 SIC/XE 규칙에 맞게 objectcode를 문자열로 만들어서 리턴해준다.

### 사용변수

Char\* op : Hash table에서 찾아낼 opcode이다

Int num : 그 데이터의 형식 넘버이다.

Int n,I,x,b,p,e : bit 들이다.

Int targetloc : target의 location값이다

Int displace : target과 명령어의 location 차이 값이다

## 모듈이름: findnum(char\* input)

### 기능

문자열 Input을 Hash table에서 찾아 table에 저장된 형식 넘버를 리턴해준다..

### 사용변수

Hnode\* temp : 링크드 리스트를 돌기 위한 임시 변수

## 모듈이름: findopcode(char\* input)

### 기능

문자열 input에 해당되는 opcode를 Hash table에서 찾아내서 리턴해준다.

### 사용변수

Hnode\* temp : 링크드 리스트를 돌기 위한 임시 변수

Temp->op : 리턴할 opcode 문자열

## 모듈이름 : findtargetloc(char\* input)

### 기능

문자열 input이 symbol일 경우 저장된 자료구조로부터 location 값을 알아내 리턴해준다.

### 사용변수

Char\* input2 :만약 target앞에 #이나 @가 붙어있을 경우 제대로된 검색을 위해 저 문자들을 삭제후 검색해주는 새로운 문자열

## 모듈이름 : complement(int num)

### 기능

10진수 num이 음수일 경우 2의 보수로 만들어 16진수로 문자열을 만들어 리턴해준다.

### 사용변수

Char\* result : 리턴할 16진수 문자열

## 모듈이름 : addsymbol(int i)

### 기능

Index I 에 해당하는 구조체 배열을 symbol table에 저장하는 함수이다. 이때 symbol들을 알파벳 순으로 저장한다.

### 사용 변수

Symbolt\* new : 저장할 새로운 구조체이다.

Symbol\* temp,temp2 : 알파벳 순으로 저장하기위한 임시 노드들이다.

## 모듈 이름 : printsymbol(void)

### 기능

Symbol table 에 저장된 데이터들을 순서대로 일정한 형식에 맞게 출력해준다.

### 사용 변수

Symbol\* temp: 데이터들을 출력하기 위해 저장할 임시 노드이다.

# 전역 변수 정의

## Int end

구조체 배열의 마지막 인덱스이다.

## Typedef struct{ int line, int loc, char one[7], char two[6], char three[8], char comma, char four[8], char junk[100], char\* ob}assm, assm\* sou

Asm 파일의 한줄씩 저장할 구조체 배열이다. 자세한 내용은 앞에 나와있다.

## Typedef struct \_SYMBOLT{ char symbol[7], int loc, struct \_SYMBOLT\* link}symbolt , symbol\* stable

Symbol들을 저장한 링크드 리스트 자료구조이다. Symbol은 symbol이며 loc은 그 symbol의 location을 나타낸다.

.

# 코드

///메인 안에 있는 새로 구현한 코드들이다.

else if(strcmp(input,"type")==0)//type일 경우 파일 내용 전체 출력

{

char junk[2]={'\0','\0'};

char file[30];

sscanf(input2,"%s %s %s",input,file,junk);//파일 이름을 뽑아낸다.

if(junk[0]!='\0')//에러체크

{

printf("Error\n");

continue;

}

int k;

int commacount=0;//comma개수를 세주는 변수

for(k=1;k<=strlen(file)-2;k++)

{

if(file[k]=='.')

commacount++;

}

if((commacount!=1||file[0]=='.')||file[strlen(file)-1]=='.')

{

printf("Error\n");

continue;

}

FILE\* p=fopen(file,"r");

if(p==NULL)//그 파일이 없을 때이다.

{

printf("No file\n");

addh(input2);

continue;

}

char c;

while((fscanf(p,"%c",&c)!=EOF))//끝까지 출력해준다.

{

printf("%c",c);

}

printf("\n");

fclose(p);

addh(input2);

}

/////////////////////////////////////////////////////////////////////////////////////////////////////////

else if(strcmp(input,"symbol")==0)//symbol 명령어 입력할때

{

if(strcmp(inputn,"symbol")!=0){//오류 체크

printf("Error\n");

continue;

}

if(stable==NULL)//테이블이 비어있을경우이다.

{

printf("Empty table\n");

addh(input2);

continue;

}

/\*

for(i=0;i<=endd;i++)//저장할 SYMBOL이경우에만 table에 추가해준다.

{

if(((sou+i)->one[0]<'A'||(sou+i)->one[0]>'Z'))

{

continue;

}

else

{

addsymbol(i);//해당 데이터를 저장

}

}\*/

printsymbol();//symbol table을 출력해준다.

//free(sou);

addh(input2);

}

else if(strcmp(input,"assemble")==0)//어셈 파일을 분석해 lst와 obj파일을 생성해준다.

{

char junk[2]={'\0','\0'};

char file[30];

sscanf(input2,"%s %s %s",input,file,junk);//마찬가지로 이름을 뽑아준다.

if(junk[0]!='\0')//에러체크

{

printf("Error\n");

continue;

}

FILE\* p=fopen(file,"r");

if(p==NULL)//에러체크

{

printf("Error\n");

continue;

}

char name[30];//lst파일을 위한 변수

char name2[30];//obj 파일을 위한 변수

sscanf(file,"%[^.]",name);//lst

sscanf(file,"%[^.]",name2);

strcat(name,".lst");

strcat(name2,".obj");

//FILE\* q=fopen(name,"w");

//FILE\* r=fopen(name2,"w");

printf("\toutput file : [%s], [%s]\n",name,name2);

char enter;

int start;

sou=(asmm\*)malloc(sizeof(asmm)\*2);

char starts[31];//어셈 파일의 첫 줄을 받을 문자열이다.

fscanf(p,"%[^\n]%\*c",starts);

sscanf(starts,"%s %s %s",sou->one,sou->two,sou->three);

if(sou->three[0]=='\0')//Start가 존재하지 않을 때이다.

start=0;//0으로 설정

else

sscanf(sou->three,"%d",&start);

if(strcmp(sou->two,"START")!=0&&find(sou->two)==1)//START가 없고 바로 코드가 시작될 경우

{

sou->line=5;

start=0;

sou->loc=start;

}

else{

sou->line=5;

sou->loc=start;

//fprintf(q,"%5d\t%04X %-10s %-8s %-9s\n",sou->line,sou->loc,sou->one,sou->two,sou->three);

}

i=1;

erflag=0;

while(1)

{

if(erflag==1)

break;

sou=(asmm\*)realloc(sou,sizeof(asmm)\*(i+2));//메모리 할당

(sou+i)->one[0]='\0';

(sou+i)->two[0]='\0';

(sou+i)->three[0]='\0';

char temp[10];

char temp2[10];

fscanf(p,"%s",temp2);

if(temp2[0]=='+')//앞에 +가 붙어있을 경우

strcpy(temp,temp2+1);

else

strcpy(temp,temp2);

if(strcmp(temp,".")==0)//빈줄일때, 주석처리된 줄일 때이다.

{

fscanf(p,"%c",&enter);

((sou+i)->one)[0]='.';

if(enter=='\n')

{

strcpy((sou+i)->junk,"\0");

}

else

{

fscanf(p,"%[^\n]%c",(sou+i)->junk,&enter);

}

(sou+i)->line=i\*5+5;

(sou+i)->loc=-1;

//fprintf(q,"%5d\t .%s\n",(sou+i)->line,(sou+i)->junk);

i++;

continue;

}

if(strcmp(temp,"END")==0)//끝 줄일 경우

{

(sou+i)->line=5\*i+5;

(sou+i)->loc=start;//location이 없는경우 -1

strcpy((sou+i)->two,temp);

fscanf(p,"%s",(sou+i)->three);

endd=i;

//fprintf(q,"%5d\t %-8s %-9s",(sou+i)->line,(sou+i)->two,(sou+i)->three);

break;

}

if(strcmp(temp,"BASE")==0)//Base일 경우이다.

{

strcpy((sou+i)->two,temp);

(sou+i)->loc=-1;

fscanf(p,"%s%c",(sou+i)->three,&enter);

}

else if(find(temp)==0)//Hash table에 없을 경우,first일 경우

{

strcpy((sou+i)->one,temp);

fscanf(p,"%s",(sou+i)->two);

if((strcmp((sou+i)->two,"COMPR")==0||strcmp((sou+i)->two,"STCH")==0)||strcmp((sou+i)->two,"LDCH")==0)//콤마가 필요할 경우

{

fscanf(p,"%[^,] %c %s%c",temp,&((sou+i)->comma),(sou+i)->four,&enter);

strcpy((sou+i)->three,rmspace(temp));

}

else//일반적인 음네모닉일때

{

fscanf(p,"%s%c",(sou+i)->three,&enter);

}

//location 설정start로 설정해준다. 일괄적으로 설정해준다.

if(((sou+i)->two)[0]=='+')//+가 앞에 붙어있을 경우 4형식

{

(sou+i)->loc=start;

start+=4;

}

else if(strcmp((sou+i)->two,"BASE")==0)//BASE일 경우

{

(sou+i)->loc=-1;

}

else if(strcmp((sou+i)->two,"RESW")==0)//RESW,RESB,BYTE일 경우 예외 처리

{

(sou+i)->loc=start;

int t;

sscanf((sou+i)->three,"%d",&t);

start+=3\*t;

}

else if(strcmp((sou+i)->two,"RESB")==0)

{

(sou+i)->loc=start;

int t;

sscanf((sou+i)->three,"%d",&t);

start+=t;

}

else if(strcmp((sou+i)->two,"BYTE")==0)

{

(sou+i)->loc=start;

if(((sou+i)->three)[0]=='C')//캐릭터형일 경우

{

int count=0;

count=strlen((sou+i)->three)-3;

start+=count;//문자 개수만큼 location 증가

}

else if(((sou+i)->three)[0]=='X')//16진수일 경우 1 증가

start+=1;

}

else if(find((sou+i)->two)==0)//Hash table안에 존재하지 않을때

{

printf("Line num %d : No mnemonic\n",i+1);

free(sou);

erflag=1;

continue;

}

else//그 외 일반적인 경우

{

(sou+i)->loc=start;

start+=findnum((sou+i)->two);

}

}

else//Hash table에있을 경우 , mnemonic 일 경우

{

strcpy((sou+i)->two,temp2);

if((strcmp((sou+i)->two,"COMPR")==0||strcmp((sou+i)->two,"STCH")==0)||strcmp((sou+i)->two,"LDCH")==0)//콤마가 필요할 경우

{

fscanf(p,"%[^,] %c %s%c",temp,&(sou+i)->comma,(sou+i)->four,&enter);

strcpy((sou+i)->three,rmspace(temp));

}

else if(strcmp((sou+i)->two,"RSUB")==0)

{

(sou+i)->line=5\*i+5;

}

else//일반적인 음네모닉

{

fscanf(p,"%s%c",(sou+i)->three,&enter);

}

//location 설정

if(((sou+i)->two)[0]=='+')//+가 앞에 붙어있을 경우 4형식

{

(sou+i)->loc=start;

start+=4;

}

else if(strcmp((sou+i)->two,"BASE")==0)//위의 경우와 같다.

{

(sou+i)->loc=-1;

}

else if(strcmp((sou+i)->two,"RESW")==0)

{

(sou+i)->loc=start;

int t;

sscanf((sou+i)->three,"%d",&t);

start+=3\*t;

}

else if(strcmp((sou+i)->two,"RESB")==0)

{

(sou+i)->loc=start;

int t;

sscanf((sou+i)->three,"%d",&t);

start+=t;

}

else if(strcmp((sou+i)->two,"BYTE")==0)

{

(sou+i)->loc=start;

start+=1;

}

else//그 외 일반적인 경우

{

(sou+i)->loc=start;

start+=findnum((sou+i)->two);

}

}

(sou+i)->line=5\*i+5;

i++;

}

//여기서부턴 SYMBOL table에 없는 타겟이 어셈 파일에 있을경우 오류를 발생하는 역할을 한다.

//SYMBOL TABLE을 비워야한다.

for(i=0;i<=endd;i++)

{

char\* temp3;

temp3=(char\*)malloc(sizeof(char)\*16);//Target 앞에 #이나 @가 붙을 수 있으므로 temp를 이용해 제대로된 스트링을 구한다.

if(strcmp((sou+i)->two,"BYTE")==0||strcmp((sou+i)->two,"RESW")==0)//예외 처리이다.

continue;

if(strcmp((sou+i)->two,"CLEAR")==0||strcmp((sou+i)->two,"STCH")==0)

continue;

if(strcmp((sou+i)->two,"COMPR")==0||strcmp((sou+i)->two,"TIXR")==0)

continue;

if(strcmp((sou+i)->two,"END")==0)

continue;

if((sou+i)->three[0]=='@'||(sou+i)->three[0]=='#')

temp3=(sou+i)->three+1;

else

strcpy(temp3,(sou+i)->three);

if(temp3[0]<='Z'&&temp3[0]>='A')

{

int k;

int flag=0;//오류 체크 플래그

for(k=0;k<=endd;k++)

{

if((sou+k)->one[0]>='A'&&(sou+k)->one[0]<='Z')

{

if(strcmp((sou+k)->one,temp3)==0)//있으면 1

{

flag=1;break;

}

}

}

if(flag==0)//오류 발생시

{

printf("Line num %d has invalid target name\n",i+1);

return 0;

}

}

}

if(erflag==1)

{

sou=NULL;

continue;

}

FILE\* q=fopen(name,"w");

FILE\* r=fopen(name2,"w");

if(strcmp(sou->two,"START")==0)//START일경우

{

fprintf(q,"%5d\t%04X %-10s %-8s %-9s\n",sou->line,sou->loc,sou->one,sou->two,sou->three);

}

else//Start로 시작하지 않을 경우 첫줄만 똑같이 규칙으로 출력해준다.

{

sou->ob=object(0);

fprintf(q,"%5d\t",sou->line);

if(sou->one[0]=='.')

fprintf(q," . %s\n",sou->junk);

else{

if(strcmp(sou->two,"BASE")!=0)

fprintf(q,"%04X ",sou->loc);

else

fprintf(q," ");

if(sou->one[0]!='\0')

fprintf(q,"%-10s",sou->one);

else

fprintf(q," ");

if(sou->two[0]=='+')

fprintf(q,"%-9s",sou->two);

else

fprintf(q," %-8s",sou->two);

if(sou->comma==',')

{

char dum[20];

strcpy(dum,sou->three);

strcat(dum,",");

strcat(dum,sou->four);

fprintf(q," %-9s",dum);

}

else

{

if(sou->three[0]=='#'||sou->three[0]=='@')

fprintf(q,"%-10s",sou->three);

else if(sou->three[0]=='\0')

fprintf(q," ");

else

fprintf(q," %-9s",sou->three);

}

if(sou->ob==NULL)

fprintf(q,"\n");

else

fprintf(q," %s\n",sou->ob);

}

}

for(i=1;i<=endd;i++)//여기서 lst파일에 써주는 역할을 한다.

{

fprintf(q,"%5d\t",(sou+i)->line);//라인 출력

if(((sou+i)->one)[0]=='.')//주석일 경우

{

fprintf(q," . %s\n",(sou+i)->junk);

}

else if(strcmp((sou+i)->two,"END")==0)//END 어셈 파일의 끝일 경우

{

fprintf(q,"\t %-8s %-9s",(sou+i)->two,(sou+i)->three);

endd = i;

break;

}

else

{

if(strcmp((sou+i)->two,"BASE")!=0)//location을 출력하지 않는다

{

fprintf(q,"%04X ",(sou+i)->loc);

}

else

{

fprintf(q," ");

}

if(((sou+i)->one)[0]!='\0')//멀쩡할 경우

{

fprintf(q,"%-10s",(sou+i)->one);

}

else

fprintf(q," ");

if(((sou+i)->two)[0]=='+')//+가 붙어져있을 경우

fprintf(q,"%-9s",(sou+i)->two);

else

fprintf(q," %-8s",(sou+i)->two);

if((sou+i)->comma==',')//콤마가 있을경우

{

char dum[20];

strcpy(dum,(sou+i)->three);

strcat(dum,",");

strcat(dum,(sou+i)->four);

fprintf(q," %-9s",dum);

}

else//일반적인 경우

{

if(((sou+i)->three)[0]=='#'||((sou+i)->three)[0]=='@')

fprintf(q,"%-10s",(sou+i)->three);

else if(((sou+i)->three)[0]=='\0')

fprintf(q," ");

else

fprintf(q," %-9s",(sou+i)->three);

}

(sou+i)->ob=object(i);//오브젝 코드 생성

if((sou+i)->ob==NULL)//오브젝 코드가 존재하지않을때

fprintf(q,"\n");

else

fprintf(q," %s\n",(sou+i)->ob);

}

}

/////////////////////////////////////여기부터 obj 파일 쓰기

if(strcmp(sou->two,"START")==0)//Start가 있을 때 없을 때 구별

{

i=1;

fprintf(r,"H%-6s%06X%06X\n",sou->one,sou->loc,(((sou+endd)->loc)-(sou->loc)));

}

else

{

fprintf(r,"H SICXE%06X%06X\n",sou->loc,(((sou+endd)->loc)-(sou->loc)));

i=0;

}

int k;

while(1)//Text Record

{

fprintf(r,"T%06X",(sou+i)->loc);

int endindex;//한줄 끝 코드의 인덱스

int lengthend;//한줄 길이

int length=0;

for(k=i;;k++)

{

if(strcmp((sou+k)->two,"BASE")==0||(sou+k)->one[0]=='.')//예외

continue;

else if((k==endd||length>=58)||(strcmp((sou+k)->two,"RESW")==0||strcmp((sou+k)->two,"RESB")==0))//이런 경우들에 obj코드 한줄의 끝을 설정한다.

{

endindex=k-1;//그전까지 인덱스를 한줄 끝으로 설정

lengthend=length;

break;

}

else

length+=strlen((sou+k)->ob);//그전까지 길이를 더한다.

}

fprintf(r,"%02X",lengthend/2);//코드 길이

for(k=i;k<=endindex;k++)

{

if(strcmp((sou+k)->two,"BASE")==0||(sou+k)->one[0]=='.')

continue;

fprintf(r,"%s",(sou+k)->ob);

}

i=k;

if(i>=endd)//브레이크

break;

fprintf(r,"\n");

while(1)//오브젝 코드가 있는 라인까지 인덱스 증가

{

if((sou+i)->ob!=NULL)

break;

if((sou+i)->ob==NULL)

i++;

}

}

fprintf(r,"\n");

for(i=1;i<=endd;i++)//Modify Record

{

if((sou+i)->two[0]=='+'&&((sou+i)->three[0]!='#'))

{

fprintf(r,"M0000%02X05\n",(sou+i)->loc+1);

}

}

fprintf(r,"E%06X",sou->loc);

for(i=0;i<=endd;i++)//symboltable에 저장

{

if(strcmp((sou+i)->two,"START")==0)

continue;

else if(((sou+i)->one[0]<'A'||(sou+i)->one[0]>'Z'))

continue;

else

addsymbol(i);

}

free(sou);

fclose(p);

fclose(q);

fclose(r);

addh(input2);

}

else//올바르지않은 입력일 경우 에러

{

printf("Error\n");

}

}

deletem();//메모리 해제

return 0;

}

///////////////////여기까지가 메인 속 코드

여기서부터 지역함수들이다

int find(char\* input)//Hash table에서 input을 찾아주는 함수

{

int i;

if(input[0]=='+')

input=input+1;

for(i=0;i<20;i++)

{

if(input[0]==(table[i]->ins)[0])

{

hnode\* temp;

temp=table[i];

while(temp!=NULL&&(strcmp(temp->ins,input)!=0))

temp=temp->link;

if(temp==NULL)

{

return 0;

}

return 1;

}

}

return 0;

}

char\* findopcode(char\* input)//opcode를 찾아 리턴해준다.

{

int i;

for(i=0;i<20;i++)

{

if(input[0]==(table[i]->ins)[0])

{

hnode\* temp;

temp=table[i];

while(temp!=NULL&&(strcmp(temp->ins,input)!=0))

temp=temp->link;

if(temp==NULL)

{

printf("No data\n");

return NULL;

}

return temp->op;

}

}

return NULL;

}

int findnum(char\* input)//Hash table에서 input의 형식 넘버를 리턴해준다.

{

int i;

for(i=0;i<20;i++)

{

if(input[0]==(table[i]->ins)[0])

{

hnode\* temp;

temp=table[i];

while(temp!=NULL&&(strcmp(temp->ins,input)!=0))

temp=temp->link;

if(temp==NULL)

return 0;

return temp->num;

}

}

return 0;

}

char\* object(int i)//Object Code를 리턴해준다.

{

char\* ob=NULL;

char\* op;

int num;

if(strcmp((sou+i)->two,"BASE")==0)//예외이다.

return NULL;

if(strcmp((sou+i)->two,"BYTE")==0)

{

if((sou+i)->three[0]=='C')//예외 처리 캐릭터일경우

{

ob=(char\*)malloc(sizeof(char)\*2);

for(int k=2;k<=strlen((sou+i)->three)-2;k++)

{

if((sou+i)->three[k]==39)

break;

else{

ob=(char\*)realloc(ob,sizeof(char)\*(2\*(k+1)));

ob[2\*(k-2)]=to16((int)((sou+i)->three[k]))[0];

ob[2\*(k-2)+1]=to16((int)((sou+i)->three[k]))[1];

}

}

}

else if((sou+i)->three[0]=='X')//16진수 일경우

{

ob=(char\*)malloc(sizeof(char)\*3);

for(int k=2;k<=strlen((sou+i)->three)-2;k++)

{

if((sou+i)->three[k]==39)

break;

else

{

ob=(char\*)realloc(ob,sizeof(char)\*((k)));

ob[(k-2)]=(sou+i)->three[k];

}

}

}

return ob;

}

else if(strcmp((sou+i)->two,"RESW")==0||strcmp((sou+i)->two,"RESB")==0)//예외 처리

return NULL;

op=(char\*)malloc(sizeof(char)\*3);

if((sou+i)->two[0]=='+'){//+가 붙어있을 경우이다.

num=findnum(((sou+i)->two)+1);

strcpy(op,findopcode(((sou+i)->two)+1));

}

else{

num=findnum((sou+i)->two);//형식

strcpy(op,findopcode((sou+i)->two));

}

int N=1;//비트들 초기화

int I=1;

int x=0;

int b=0;

int p=0;

int e=0;

if((sou+i)->three[0]=='#')//immediate addressing

{

N=0;

I=1;

}

else if((sou+i)->three[0]=='@')//indirect addressing

{

N=1;

I=0;

}

if((sou+i)->two[0]=='+'){//4형식임을 알려준다.

num=4;

e=1;

}

if((sou+i)->comma==','&&(strcmp((sou+i)->two,"STCH")==0||strcmp((sou+i)->two,"LDCH")==0))

{

x=1;

b=1;

}

ob=(char\*)malloc(sizeof(char)\*9);

int targetloc=findtargetloc((sou+i)->three);//target address이다.

int displace;//타켓과 현재 위치의 주소차이이다.

if((sou+i+1)->loc==-1)

displace=targetloc-(sou+i+2)->loc;

else

displace=targetloc-(sou+i+1)->loc;

int temp=x\*8+b\*4+p\*2+e\*1;

char\* disp;

disp=(char\*)malloc(sizeof(char)\*3);

//음수면 2의 보수로 변환

if(strcmp((sou+i)->two,"RSUB")==0)//예외

{

ob[0]=op[0];

ob[1]=op[1]+2\*N+I;

ob[2]='0';

ob[3]='0';

ob[4]='0';

ob[5]='0';

ob[6]='\0';

return ob;

}

if(strcmp((sou+i)->two,"TIXR")==0)//예외

{

ob[0]=op[0];

ob[1]=op[1];

ob[2]='5';

ob[3]='0';

ob[4]='\0';

return ob;

}

if(num==3)//3형식일때

{

ob[0]=op[0];

ob[1]=op[1]+2\*N+I;

if(op[1]<='9'&&ob[1]>'9'){

ob[1]+=7;

}

else if(ob[1]>'F')//opcode에 ni값을 더했을 경우 자릿수가 넘어갈때이다.

{

ob[1]=ob[1]-'F'-1+'0';

ob[0]++;

if(ob[0]==58)

ob[0]='A';

}

if(strcmp((sou+i)->two,"STCH")==0||strcmp((sou+i)->two,"LDCH")==0)//이경우엔

{

int k;

for(k=1;k<endd;k++)

{

if(strcmp((sou+k)->two,"BASE")==0)

break;

}

displace=targetloc-findtargetloc((sou+k)->three);

ob[2]=temp+'0';

if(ob[2]>'9')

ob[2]+=7;

ob[3]='0';ob[4]='0';ob[5]='0';

for(k=strlen(to16(displace))-1;k>=0;k--)

{

ob[k-strlen(to16(displace))+6]=to16(displace)[k];//뒤에서부터 채워준다. 나머지는 0으로 채워준다.

}

ob[6]='\0';

return ob;

}

if((sou+i)->three[0]=='#')//immediate addressing 일 때

{

char dump[10];

dump[0]='\0';

char ctarget;

int target;

sscanf((sou+i)->three,"%\*c%c%s",&ctarget,dump);

if(ctarget>='0'&&ctarget<='9'){

sscanf((sou+i)->three,"%\*c%d",&target);

//숫자만 있을경우

ob[2]='0';

ob[3]='0';ob[4]='0';ob[5]='0';

for(int k=strlen(to16(target))-1;k>=0;k--)

{

ob[k-strlen(to16(target))+6]=to16(target)[k];//위와 마찬가징다.

}

ob[6]='\0';

return ob;

}

}

if(displace>=-2048&&displace<=2047)//PC relative일때이다.

p=1;

else//Base relative일 때이다.

b=1;

if(displace<0&&b!=1){

temp=x\*8+b\*4+p\*2+e\*1;

ob[2]=temp+'0';//

ob[3]=complement(displace)[1];

ob[4]=complement(displace)[2];

ob[5]=complement(displace)[3];

ob[6]='\0';

return ob;

}

if(b==1)//Base relative일때

{

temp=x\*8+b\*4+p\*2+e\*1;

int k;

for(k=1;k<endd;k++)

{

if(strcmp((sou+k)->two,"BASE")==0)

break;

}

displace=targetloc-findtargetloc((sou+k)->three);

ob[2]=temp+'0';

if(ob[2]>'9')

ob[2]+=7;

ob[3]='0';ob[4]='0';ob[5]='0';

for(k=strlen(to16(displace))-1;k>=0;k--)

{

ob[k-strlen(to16(displace))+6]=to16(displace)[k];

}

ob[6]='\0';

return ob;

}

if(strlen(to16(displace))==1)//disp가 한자리 일때이다.

{

disp[0]='0';

disp[1]=to16(displace)[0];

}

else

disp=to16(displace);

temp=x\*8+b\*4+p\*2+e\*1;

ob[2]=temp+'0';

ob[3]='0';

ob[4]=disp[0];

ob[5]=disp[1];

ob[6]='\0';

}

else if(num==2)//2 포멧일 때

{

ob[0]=op[0];

ob[1]=op[1];

if(strcmp((sou+i)->two,"COMPR")==0)

{

if(strcmp((sou+i)->three,"A")==0)

ob[2]='0';

else if(strcmp((sou+i)->three,"S")==0)

ob[2]='4';

else if(strcmp((sou+i)->three,"X")==0)

ob[2]='1';

else if(strcmp((sou+i)->three,"L")==0)

ob[2]='2';

else

ob[2]='8';

if(strcmp((sou+i)->four,"A")==0)

ob[3]='0';

else if(strcmp((sou+i)->four,"S")==0)

ob[3]='4';

else if(strcmp((sou+i)->four,"X")==0)

ob[3]='1';

else if(strcmp((sou+i)->four,"L")==0)

ob[3]='2';

else

ob[3]='8';

ob[4]='\0';

return ob;

}

if(strcmp((sou+i)->three,"X")==0)

{

ob[2]='1';

ob[3]='0';

}

else if(strcmp((sou+i)->three,"A")==0)

{

ob[2]='0';

ob[3]='0';

}

else if(strcmp((sou+i)->three,"S")==0)

{

ob[2]='4';

ob[3]='0';

}

ob[4]='\0';

return ob;

}

else if(num==4)//4형식일때이다.

{

temp=x\*8+b\*4+p\*2+e\*1;

if(strcmp((sou+i)->two+1,"JSUB")==0||strcmp((sou+i)->two,"+JSUB")==0)

{

ob[0]=op[0];

ob[1]=op[1];

ob[2]=temp+'0';

ob[3]='0';

ob[4]='\0';

strcat(ob,to16(targetloc));

return ob;

}

if((sou+i)->three[0]=='#')//네모닉 앞에 +가 붙고 target 앞에 #이 붙을 때

{

ob[0]=op[0]+N\*2;

ob[1]=op[1]+I;

ob[2]=temp+'0';

ob[3]='0';

int tar;

sscanf((sou+i)->three,"%\*c%d",&tar);

ob[4]='0';

ob[5]='0';

ob[6]='0';

ob[7]='0';

for(int k=strlen(to16(tar))-1;k>=0;k--)

{

ob[8+k-strlen(to16(tar))]=to16(tar)[k];

}

ob[8]='\0';

return ob;

}

}

return ob;

}

int findtargetloc(char\* input)//input에 해당하는 니모닉의 location을 리턴해준다.

{

int i=0;

char\* input2;

if(input[0]=='#'||input[0]=='@')

input2=input+1;

else

input2=input;

for(i=1;i<=endd;i++)

{

if(strcmp((sou+i)->one,input2)==0)

{

return (sou+i)->loc;

}

}

return -1;

}

char\* complement(int num)//음수를 2의 보술 바꿔주는 함수

{

char\* result;

result=(char\*)malloc(sizeof(char)\*5);

result=to16(65535+num+1);

return result;

}

void addsymbol(int i)//i번째 심볼을 테이블에 저장

{

symbolt\* new;

new=(symbolt\*)malloc(sizeof(symbolt));

strcpy(new->symbol,(sou+i)->one);

new->loc=(sou+i)->loc;

new->link=NULL;

if (stable==NULL)

{

stable=new;

return;

}

else

{

symbolt\* temp;

symbolt\* temp2;

temp=stable;

temp2=NULL;

while(strcmp(temp->symbol,new->symbol)>=0&&temp!=NULL)//알파벳 순으로 저장되는 알고리즘

{

temp2=temp;

temp=temp->link;

if(temp==NULL)

break;

}

if(temp2==NULL)

{

new->link=temp;

stable=new;

}

else{

temp2->link=new;

new->link=temp;

}

}

}

void printsymbol(void)//symbol 테이블을 출력해준다.

{

symbolt\* temp;

temp=stable;

while(temp!=NULL)//NULL일 경우 break한다

{

if(temp->link==NULL)//마지막줄을 \n 생략

printf("\t%s\t%04X",temp->symbol,temp->loc);

else

printf("\t%s\t%04X\n",temp->symbol,temp->loc);

temp=temp->link;

}

}

//여기까지가 코드 끝