과목 명: 시스템프로그래밍

담당 교수 명: 김 지 환

<<Assignment 2>>

**서강대학교 컴퓨터학과**

**[학번 : 20141602]**

**[이름 : 황기덕]**

목 차

1. 프로그램 개요 3

2. 프로그램 설명 3

2.1 프로그램 흐름도 3

2.1.1 type 3

2.1.2 assemble 4

2.1.3 symbol 5

3. 모듈 정의

3.1 모듈 이름 : type\_filename(FILE\*) 5

3.1.1 기능 5

3.1.2 사용 변수 5

3.2 모듈 이름: assemble\_file(char\*) 5

3.2.1 기능 5

3.2.2 사용 변수 5

3.3 모듈이름: assemble\_error(FILE\*, FILE\*, FILE\*, int, char\*, int, char\*) 6

3.3.1 기능 6

3.3.2 사용변수 6

3.4 모듈이름: add\_symbol\_table(char\*, int) 7

3.4.1 기능 7

3.4.2 사용변수 7

3.5 모듈이름: find\_symbol\_table(char\*) 7

3.5.1 기능 7

3.5.2 사용변수 7

3.6 모듈이름: free\_symbol\_table() 7

3.6.1 기능 7

3.6.2 사용변수 7

3.7 모듈이름: symbol\_display() 7

3.7.1 기능 7

3.7.2 사용변수 7

4. 전역 변수 정의 8

4.1 symtab \*symbol\_table 8

4.2 modi \*modified, \*modilast 8

# 프로그램 개요

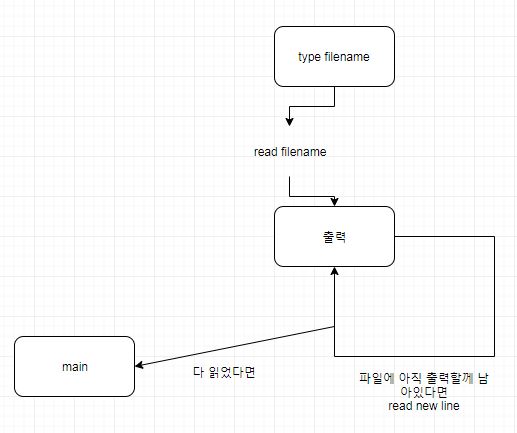
프로젝트 #1에서 구현한 쉘에 assemble 기능을 추가한다.

SIC/XE의 assembly program source 파일을 입력 받아서 object 파일을 생성하고, 어셈블리 과정 중 생성된 symbol table과 결과물인 object 파일을 볼 수 있는 기능을 제공한다.

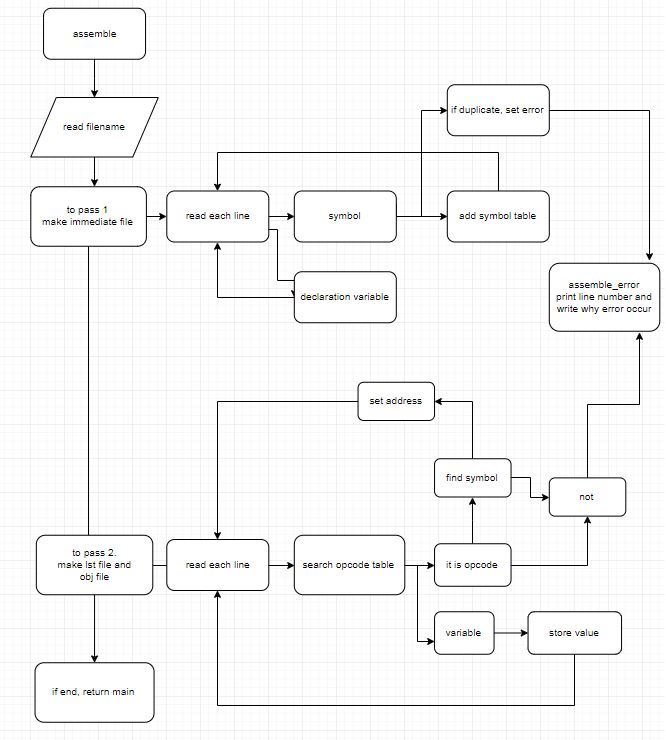
# 프로그램 설명

## 2. 1 프로그램 흐름도

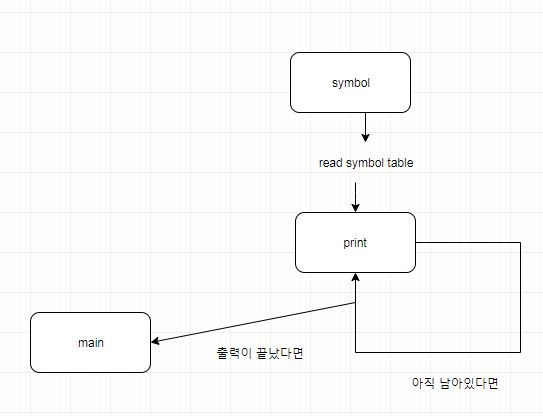
2. 1. 1 type



2.1.2 assemble



2.1.3 symbol



# 모듈 정의

## type\_filename(FILE \*)

## 기능

파일이름을 받아서 그 파일을 열고, 그 파일 내용을 출력해준다.

3.1.2 사용 변수

### char tmp

## assemble\_file(char \*)

## 기능

## asm 파일을 읽어들여서 lst 파일과 obj 파일을 만들어준다

## 사용 변수

### int line\_number = 0; // 줄 수를 저장 int LOCCTR = 0; // location counter을 저장 int ex\_LOCCTR = 0; // 이전 LOCCTR을 저장 int next\_LOCCTR = 0; // 이 다음 LOCCTR을 저장 int base\_address; // base address를 저장 int i, j, k; // 반복문을 위해 선언 int opcode\_number; // opcode의 숫자를 나타냄 int variable\_number; // asm에서 변수 선언 시, 어떤 숫자를 저장할지 int format4\_flag = 0; // 포멧 4이면 저장 int operand\_address; // operand 심볼의 LOCCTR을 저장 int start\_address, end\_address; // asm 시작부분의 주소와 끝부분의 주소 저장 int obj\_newlineflag = 0; // obj 코드에서 줄이 일정 수 넘어가면 다음 줄로 넘어가라고 저장 int obj\_linelen; // 현재 obj 파일 라인에서 몇 바이트를 썻는지 저장 unsigned int disp; // obj code에 저장할 disp를 저장 unsigned int A\_ad, B\_ad, C\_ad, D\_ad; // ABCD -> object code의 각각의 바이트를 저장 int numberflag, labelflag, commaflag, xflag, formatflag;

### //각종 플래그 char line\_input\_by\_asm[100]; // 파일에서 읽어들이는 것 저장 char output\_at\_nextfile[100]; // 다음 파일(pass1 -> immediate, pass2 - > lst file) 에 저장. char \*front\_filename; // filename 저장 char \*tmp\_input[6]; // 각 opcode, symbol 등을 위한 임시저장장소 char hash\_type[6]; // opcode의 type 확인 char output\_at\_objfile[100]; // obj file에 저장할 라인 저장 char tmpchar; // char 저장할 임시변수 char asm\_code\_name[10]; // assemble 코드 이름 modi \*modistep, \*moditmp; // obj 파일 만들 시, modify 할 라인을 저장하기 위한 링크드리스트

### FILE \*fp\_asm, \*fp\_lst, \*fp\_obj // 각각의 파일포인터

### 

## assemble\_error(FILE \*, FILE \*, FILE \*, int , char \*, int, char \*)

## 기능

## assemble 함수를 진행하다가 무언가 오류가 발생하면 열었던 파일들을 모두 닫고 그 파일을 제거하며 라인 넘버와 에러가 생성된 이유를 출력해준다.

## 사용 변수

없음

### 

## add\_symbol\_table(char \*, int)

## 기능

## assemble 하는 도중 새로운 symbol이 있으면 링크드 리스트에 정렬된 상태로 추가해준다

## 사용 변수

### symtab \*tmp, \*tmpstep, \*exstep // symbol 링크드리스트에 추가하기 위한 변수

## find\_symbol\_table(char \*)

## 기능

## asm 파일에서 읽어들인 symbol 중, 이전에 이미 나온 심볼인지 아닌지를 확인해 있으면 그 LOCCTR을, 아니면 -1을 반환해준다

## 사용 변수

### symbol \*find // 검색을 위한 임시변수

int num // LOCCTR 혹은 찾기 실패를 반환해줄 변수

## free\_symbol\_table()

## 기능

## 새로운 파일 어셈블 시, symbol 링크드 리스트를 free해주기 위해 사용한다.

## 사용 변수

### symtab \*symbolstep, \*nextsymbolstep // free를 위한 임시변수

## free\_symbol\_table()

## 기능

## symbol 명령어 입력시, symbol을 출력해준다,

## 사용 변수

### symtab \*step // 출력을 위한 변수.

# 전역 변수 정의

# symtab \*symbol\_table

어셈블 할 시 있는 심볼들을 저장하기 위한 링크드 리스트.

# modi \*modified, \*modilast

오브젝트 코드를 생성 시, 필요한 modified 하는 것을 링크드 리스트로 저장하기 위한 변수