과목 명: 시스템프로그래밍

담당 교수 명: 김 지 환

<<Assignment 3>>

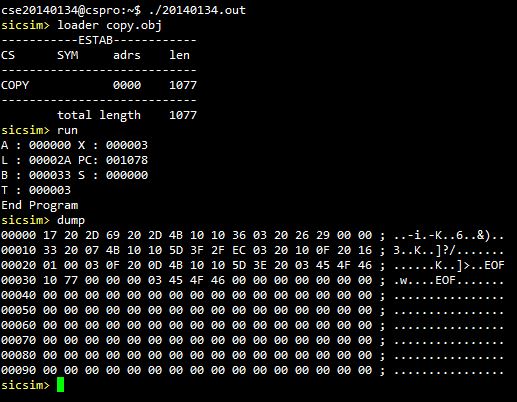
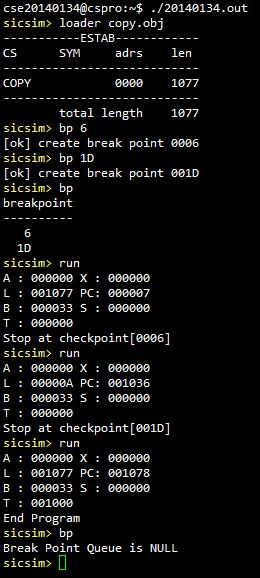
**서강대학교 컴퓨터학과**

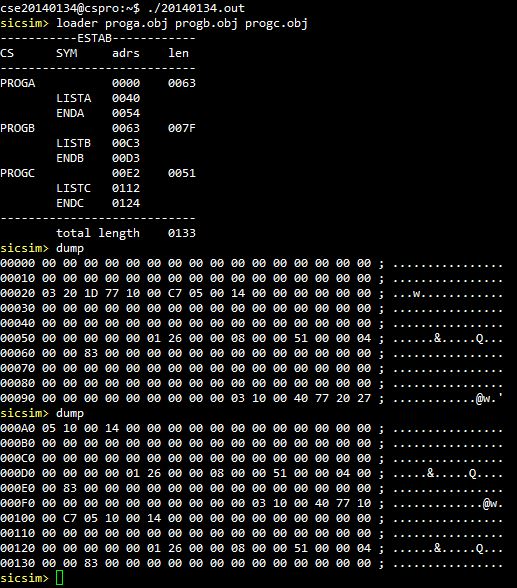
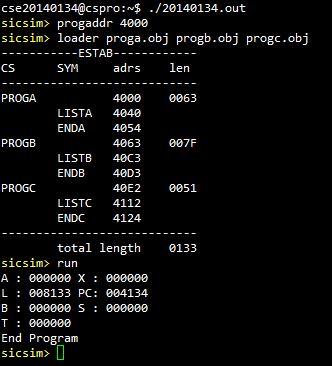
**[학번] 20140134**

**[이름] 조 예린**

# 프로그램 개요

프로젝트 #1, #2 에서 구현한 셀(shell)에 linking과 loading 기능을 추가하는 프로그 램입니다. 프로젝트 #2 에서 구현된 assemble 명령을 통해서 생성된 object 파일을 link시켜 메모리에 올리는 일을 수행합니다.





# 모듈 정의

## 모듈 이름 : put\_ESTAB(char\* cs\_name, char\* sym\_name, int adrs, int len)

### 기능

External Symbol Table에 노드를 추가한다.

### 사용 변수

pESTAB new\_node : 외부 심볼 테이블에 추가될 노드

## 모듈 이름: print\_ESTAB()

### 기능

ESTAB을 출력한다. 만약 외부 심볼 테이블이 비어있으면 “ESTAB is null”을 출력한다.

### 사용 변수

pESTAB new\_es\_p : 외부 심볼 테이블을 돌며 출력을 도울 노드

## 모듈이름: search\_ESTAB(char\* cs\_name, char\* sym\_name)

### 기능

외부 심볼 테이블에서 control section name혹은 symbol name을 검색하고 그 주솟값을 반환한다. 사용자가 control section name을 찾고 싶을 때에는 symbol name에 “\*”을 입력하고, symbol name을 찾고싶을 때에는 control section name에 “\*”을 입력해주어야 한다.

만약 외부 심볼 테이블이 비어있다면 “ESTAB is null”을 출력한다.

### 사용변수

Int result\_adrs : 반환할 주솟값을 받는 변수이다.

## 모듈이름: clear\_ESTAB()

### 기능

ESTAB을 비우는 함수로, ESTAB을 가리키는 head node를 받은 es\_p가 차례로 다음 노드를 받으며, es\_p가 NULL이면 종료한다.

### 사용변수

없음

## 모듈이름: progaddr(char\* adrs)

### 기능

Progaddr을 사용자에게 입력을 받아 전역 변수 program\_adrs에 저장한다.

### 사용변수

Char\* ptr : strtol에 사용할 일종의 버퍼 변수

Long val\_sth : strtol에서 결과값을 반환 받을 변수

Int val\_int : val\_sth를 int로 형변환 하여 저장할 변수

## 모듈이름: split(char\* string, int start\_idx, int end\_idx)

### 기능

문자열을 받아 start index에서 end index까지 잘라내어 그 결과값인 문자열을 반환하는 함수로, for loop을 돌며 start\_idx부터 end\_idx까지 strcat을 이용해 한 글자씩 result문자열에 추가해준다.

### 사용변수

Char\* result : 결과값을 저장할 변수

## 모듈이름: str\_to\_int(char\* string, int radix)

### 기능

문자열과 진수(radix)를 입력받아 문자열을 숫자로 바꾸어주는 함수이다. 가령 “1234”문자열의 경우에는 1234를 값으로 가지는 integer 변수를 반환한다.

### 사용변수

Char\* ptr : strtol에 사용할 일종의 버퍼 변수

Long val\_sth : strtol에서 결과값을 반환 받을 변수

Int val\_int : val\_sth를 int로 형변환 하여 저장할 변수

## 모듈이름: loader(char\* file1, char\* file2, char\* file3)

### 기능

File을 최대 3개까지 받아 load를 수행한다.

만약 file1, file2만 입력되었으면 file3에는 미리 “\*”을 넣어두어 segment fault error없이 file3이 없음을 표현한다.

입력된 파일들을 순서대로 읽으며 load와 link를 수행한다. -> “\*”인지 아닌지 확인하며 파일 개수를 세고, 마지막 파일의 경우에는 맨 끝 문자를 ‘\0’으로 바꿔준다. ‘\n’이 입력되어 있어 검색에 용이하지 않기 때문이다.

[pass 1]

우선 헤더 레코드의 경우에는, 사전에 구현해둔 split 함수를 구현해 1 line을 control section name, len, adrs로 나누어 저장해준다. 단, 저장할 때에는 PROGADDR과 total\_len을 더하는 등 linking이 가능하도록 한다.

Define Record의 경우에도 마찬가지로 split 함수를 구현해 1 line의 내용을 필요한 내용에 따라 자른다. 단, 심볼 이름에 공백까지 포함된 경우에는 ESTAB 검색이 경우에 따라 곤란할 수 있으므로 공백을 제거하는 delete\_white\_space()함수를 사용하여 공백을 제거하여준다. Define Record에서 나온 심볼들은 put\_ESTAB()함수를 이용하여 ESTAB에 추가한다.

[pass 2]

헤더 레코드 : pass 1 때와 유사하게 진행된다. Control Section의 이름을 받아 search\_ESTAB() 함수를 통해 해당 control section의 주솟값을 받아온다.

이후는 End Record를 읽을 때까지 while문을 반복한다.

Define Record는 pass 2에서 사용할 일이 없으므로 넘긴다.

Refer Record는 split 함수를 사용하여 해당 섹션에서 사용할 외부 심볼의 참조 기호를 ref\_sym\_table에 저장한다.

Text Record에서는 split함수를 사용하여 주소, 길이, 오브젝트 코드를 받아오고 이를 메모리에 하프바이트씩 저장한다

Modificaion Record에서는 split함수를 사용하여 주소, 길이, 기호(+,-), 심볼 넘버를 받아와 modification을 수행한다. 받은 심볼 넘버를 사용하여 외부 심볼의 주솟값을 받아온다. Get\_from\_mem()함수를 사용하여 주소의 메모리 내용을 받아와 외부 심볼과 연산한 후 insert\_mem()함수를 이용하여 다시 메모리에 추가한다.

모든 pass가 끝난 후에는 ESTAB의 내용을 clear해준다.

### 사용변수

Int len : 파일 이름의 길이를 받는 변수이다.

Int PROGADDR : program\_adrs 전역변수의 값을 받는다.

Int CSADDR : PROGADDR을 받는다.

Char Header[255] : 헤더 레코드를 받는 변수

Char\* Control\_Section : split()함수의 결과를 반환받을 변수

Int cs\_adrs : 주솟값을 반환받을 변수

Char record[255] : 한 줄씩 레코드 내용을 받아들일 변수

Char ref\_sym\_table[MAX\_REF\_NUM][20] : 각 섹션마다의 외부 심볼 번호를 저장해서 linking을 진행하도록 한다

Char\* original\_mem : get\_from\_mem을 사용하여 메모리의 내용을받아와 저장할 변수

Int mem\_to\_int : 메모리 값을 int로 변환하여 저장할 변수

Char\* new\_result : 외부 심볼과의 연산 이후 값을 저장하는 변수

## 모듈이름: hex\_to\_ascii(int hex)

### 기능

16진수값을 받아 알맞은 아스키코드 값으로 변환해주는 함수로, 소문자 대문자에 따라 다른 값을 더해준다. 해당 함수는 새로 추가되긴 하였으나 실제 함수에서 쓰이진 않았다.

### 사용변수

없음

## 모듈이름: get\_from\_mem(int m\_adrs, int m\_len)

### 기능

주솟값과 길이를 입력받아 해당 길이만큼 메모리에서 내용을 가져온다

가령 n 주솟값에서 5개의 글자만 받는다면, 메모리의 내용은 half byte로 저장되어 있으므로 이를 고려하여 12 34 56 에서 2 34 56만 떼와 반환해줄 수 있도록 한다.

### 사용변수

Int original\_len : 입력된 길이값을 받아 저장한다.

Char\* result : 메모리의 내용을 받아올 문자열 변수

Int dif\_of\_len : 원래 문자열의 길이와 수정된 길이의 차이를 계산하여 저장한다. 5글자가 들어온 경우 등에 사용된다.

## 모듈이름: put\_zero\_str(char\* string, int len)

### 기능

문자열과 문자열이 되어야 할 길이를 받아 비는 공간이 있다면 앞에 0을 추가해주고 넘는 공간이 있다면 앞부분을 잘라준다.

### 사용변수

char\* result : 결과 문자열을 저장하여 반환해줄 문자열 변수

int string\_len : 원래 문자의 길이

int dif\_of\_lens : len과 string\_len의 차이값을 받아 저장해 연산에 사용한다

## 모듈이름: two\_complement(int minus\_hex)

### 기능

5글자인 음수 hex가 입력되면 올바른 10진수 값으로 반환한다

### 사용변수

Int result : 결과를 받아 저장하고 반환하는 변수

## 모듈이름: print\_bp()

### 기능

사용자가 입력한 중단점들을 출력한다. 아무것도 없을 경우 Break Point Queue is NULL 을 출력한다.

### 사용변수

없음

## 모듈이름: run()

### 기능

메모리에 로드된 값을 1바이트씩 가져와 실행시킨다.

명령어를 아직 받지 않은경우에는 해당 1바이트를 명령어로 간주하고 계산한다. 이후 명령어를 전역변수 now\_seeing\_operation에 저장한다.

명령어를 받은 경우에는 get\_operation()함수에서 명령어의 형태에 따라 추가해준 STEP에 맞추어 메모리를 받는다. STEP이 1까지 간 경우엔 excute\_operation()을 이용해 명령어의 내용을 수행한다.

PC가 맨 끝에 도달하면 탈출하고, 레지스터 값을 출력한다.

### 사용변수

Int STOP : break point를 받고 만약 break point가 현재 PC보다 적다면 PROGRAM\_END값을 대신 받는다

Int memory : PC값의 바이트를 받아온다.

Int mem\_buf : PC값의 메모리 내용을 받아온다

Char\* buffer : mem\_buf 내용을 sprintf를 사용한 문자열로 반환받을 변수

## 모듈이름: bp(char\* input\_adrs)

### 기능

사용자가 입력한 중단점을 저장한다.

### 사용변수

Int adrs : 사용자가 입력한 adrs값을 integer로 바꾸어 저장한다

## 모듈이름: clear\_bp()

### 기능

Break point queue를 정리한다.

### 사용변수

pBP new\_node : POP()을 통해 노드를 받아가며 삭제를 도울 변수

## 모듈이름: POP()

### 기능

Break point queue에서 노드를 꺼낸다

### 사용변수

pBP temp\_rare : 큐의 맨 뒤를 가리키는 rare를 받아올 변수

## 모듈이름: get\_operation(int opni)

### 기능

Run()에서 현재 받은 operation을 받아, 0,1,2,3(ni값)을 빼가며 해당 opcode가 있는지 탐색한다. for문이 끝나면 전역변수 now\_seeing\_operation에 해당 operation이 입력되고 전역변수 ni에 ni값이 저장된다. 이때 명령어들의 format과 operand 개수(STEP)도 저장해주도록 한다.

### 사용변수

int temp\_opni : Run()에서 넘겨준 operation을 저장

int temp\_ni : ni값을 저장한다

char operation[20] : temp\_opni을 sprintf를 통해 문자열로 만든 것을 저장하는 변수

## 모듈이름: excute\_operation()

### 기능

Operation을 수행한다. 각각의 명령어에 따라 모두 다르게 작동하며, ni값이나 format에 따라 다르게 작동한다. 단, format에 따라 사전에 현재 인자가 저장되어 있는 now\_seeing\_operand를 알맞게 잘라 operand에 저장해준다.

### 사용변수

Char\* operand : format에 맞추어 잘린 operand를 저장할 변수

Char\* xbpe : xbpe를 저장하는 변수로 get\_xbpe()를 통해 4자리 2진수로 저장된다

Char\* buf\_xbpe : xbpe bit를 저장한다

## 모듈이름: get\_xbpe(char\* buffer)

### 기능

Xbpe 비트를 받아 4자리 2진수로 반환해준다

### 사용변수

Char\* result : 결과를 저장해 반환해줄 변수

int binary, sum, i : 2진수로의 변환 과정에 사용되는 변수

int val : get\_xbpe가 입력받은 값을 integer로 바꾸어 저장하는 변수

## 모듈이름: insert\_mem(int adrs, int value, int len)

### 기능

메모리에 입력값에 따라 내용을 넣는 함수이다.

### 사용변수

Char\* buffer : 입력받은 value를 문자열로 바꾸어 저장할 때 사용되는 변수

Int t : edit에서 adrs를 1씩 올려줄 때 사용할 일종의 인덱스 변수

Int I : 문자열을 하프바이트로 바꾸어 다시 저장해야하므로 이를 위해 사용하는 일종의 인덱스 변수

# 전역 변수 정의

## DEVICE\_STATEMENT : I/O 명령어를 처리할 때 사용하는 define된 변수. 1(ready)로 정의되어있다.

## DEVICE\_INPUT : RD 명령어를 처리할 때 사용하는 INPUT을 대체할 값. 무한루프를 돌지 않기 위해 0으로 저장해두었다.

## program\_adrs : 프로그램 주솟값을 저장하는 전역변수

## total\_len : 프로그램의 총 길이를 저장하는 전역변수

## PROGRAM\_END : 프로그램의 맨 끝을 저장하는 전역변수

## A,X,L,PC,B,S,T,SW : 레지스터 값을 저장할 전역변수

## now\_seeing\_operation, now\_seeing\_operand : run() 과정에서 현재 받아둔 명령어와 인자를 저장하는 전역변수

## ni : ni값을 저장하는 전역변수

## format : format을 저장하는 전역변수

## STEP : run에서 받아야할 오퍼랜드 바이스 개수를 저장하는 전역변수

# 코드

/\*포함되는 파일\*/

#include<stdio.h>

/\*정의되는 상수\*/

#define MAX 100

#define YES 1

#define NO 0

/\*변수\*/

long income[MAX];

int month[MAX], day[MAX], year[MAX];

int cont;

int ctr;

/\*함수 원형\*/

void main(void);

int display\_instructions(void);

void get\_data(void);

void display\_report(void);

int continue\_funtion(void);

/\*프로그램 시작\*/

void main(void){

cont = display\_instructions();

if (cont == YES){

get\_data();

display\_report();

}else

printf("\nProgram Aborted by User!!\n\n");

}

/\*------------------------------------------------------------------------------------\*/

/\*함수 : display\_intructions\*/

/\*목적 : 프로그램 사용에 대한 정보를 나타내고, 중단하고 싶으면 0을 계속하고 싶으면 1을 입력할것을 사용자에게 요청한다.\*/

/\*리턴값 : YES - 1

NO - 0\*/

/\*------------------------------------------------------------------------------------\*/

int display\_instructions(void){

printf("\n\n");

printf("\nThis program enables you to enter up to 99 people\'s");

printf("\nincomes and birthdays. It then p;rints the incomes by");

printf("\nmonth along with the overall income and overall average.");

cont = continue\_funtion();

return(cont);

}

/\*------------------------------------------------------------------------------------\*/

/\*함수 : get\_data()\*/

/\*목적 : 사용자로부터 데이터를 입력받는다.

100명의 사용자가 입력되거나 사용자가 월에대해 0을 입력할때까지 계속해서 데이터를 입력받는다\*/

/\*리턴값 : 없음\*/

/\*------------------------------------------------------------------------------------\*/

void get\_data(void){

for(cont = YES, ctr = 0; ctr < MAX && cont == YES; ctr++){

printf("\nEnter information for Person %d.", ctr + 1 );

printf("\n\tenter Birthday: ");

do{

printf("\n\tMonth(0 - 12) : ");

scanf("%d", &month[ctr]);

}while(month[ctr] < 0 || month[ctr] > 12);

do{

printf("\n\tday(0 - 31) : ");

scanf("%d", &day[ctr]);

}while(day[ctr] < 0 || day[ctr] > 31);

do{

printf("\n\tYear(0 - 2006) : ");

scanf("%d", &year[ctr]);

}while(year[ctr] < 0 || year[ctr] > 2006);

printf("\nEnter Yearly Income (whole dollars): ");

scanf("%d", &income[ctr]);

cont = continue\_funtion();

}

}

/\*------------------------------------------------------------------------------------\*/

/\*함수 : display\_report\*/

/\*목적 : 화면에 보고서를 나타낸다.\*/

/\*리턴값 : 없음\*/

/\*------------------------------------------------------------------------------------\*/

void display\_report(){

int x, y;

long grand\_total = 0, month\_total = 0;

printf("\n\n");

printf("\n SALARY SUMMARY");

printf("\n ==============");

for(x = 0; x <= 12; x++){

month\_total = 0;

for(y = 0; y <ctr; y++){

if(month[y] == x)

month\_total += income[y];

}

printf("\n\nTotal for month %d is %ld",x, month\_total);

grand\_total += month\_total;

}

printf("\n\nReport totals : ");

printf("\n\nTotal income is %d", grand\_total);

printf("\n\n\* \* \* End of Report \* \* \*\n");

}

/\*------------------------------------------------------------------------------------\*/

/\*함수 : continue\_funtion()\*/

/\*목적 : 사용자가 계속 진행하고 싶은지의 여부를 묻는다.\*/

/\*리턴값 : YES - 계속하고 싶은 경우

NO - 중단하고 싶은 경우\*/

/\*------------------------------------------------------------------------------------\*/

int continue\_funtion(void){

int ask;

printf("\n\nDo you wish to continue? (0 = NO/ 1 = YES) : ");

scanf( "%d", &ask);

while(ask < 0 || ask >1){

printf("\n\n%d is invalid! ", cont);

printf("\n\nPlease enter 0 to Quit or 1 to continue: ");

scanf("%d", &ask);

}

if(ask == 0)

return(NO);

else

return(YES);

}