상명대학교 컴퓨터과학과 "EA0003: 시스템소프트웨어" 실습

- 실습 번호: lab05

- 실습 디렉터리: ~/sys\_sw/lab05 이름: 임현

- 실습 날짜: 2016년 5월 26일

- 실습 제목: Subprogram 학번: 201511054

- 실습 내용:

1. (과제준비) 자신의 홈 디렉터리 아래에 과제 디렉터리 (~/sys\_sw/lab05)를 만든다. 이번 과제에서 만들어지는 모든 파일들은 이 디렉터리에 두도록 한다.



1. 다음 두 개의 모듈로 이루어진 어셈블리 프로그램에 대해 물음에 답하시오.

main:

**…**

push dword 3

push dword 2

push dword 1

call subpr

add esp, 12

**…**

subpr:

push ebp

mov ebp, esp

sub esp, 8

;

; ①

;

mov esp, ebp

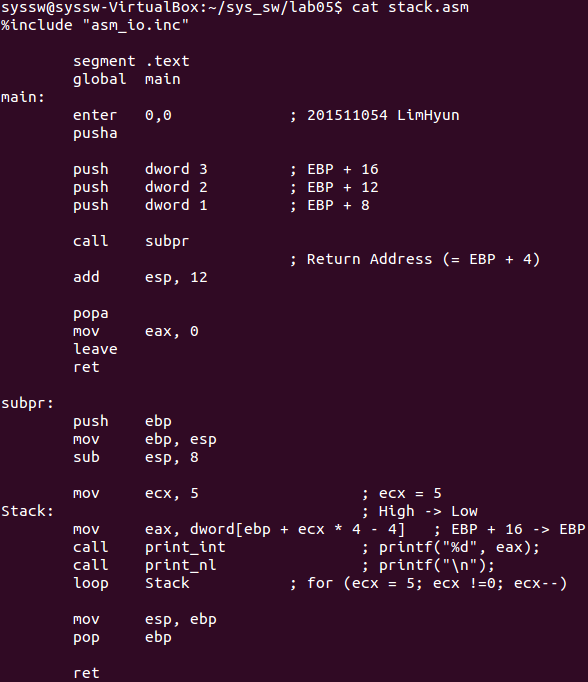
pop ebp

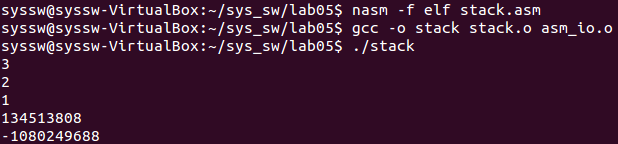
ret

* 1. 위 어셈블리 프로그램의 각 라인에 적절한 설명문을 붙이고, 서브프로그램 subpr 에서 call frame 이 형성된 직후 (① 부분)의 스택의 모습을 그리시오.

|  |  |  |
| --- | --- | --- |
| 3 | EBP + 16 | High |
| 2 | EBP + 12 |  |
| 1 | EBP + 8 |  |
| Return Address | EBP + 4 |  |
| Saved EBP | EBP | Low |

* 1. ① 부분에 call frame의 내용을 출력하는 코드를 추가한 프로그램 stack.asm을 작성하시오. 이 프로그램을 어셈블, 링크하여 실행 파일 stack 을 생성하고, 이를 실행시킨 결과를 화면 캡쳐하여 첨부하시오. 첨부된 출력값이 1.1.의 답과 같은지 확인하시오.





첨부된 출력 값이 1.1의 답과 같습니다.

* 1. 서브프로그램의 시작 시 call frame을 형성하는 부분과 종료 직전 call frame을 해제하는 부분은 각각 단일 명령어인 enter 와 leave 로 대체할 수 있다.
     1. 참고자료 NASM 매뉴얼의 appendix B.4.65 절과 B.4.136절을 참고하여 이 명령어들에 대한 사용 문법, 기능, 동작 과정 등을 조사하여 적으시오.

Enter : 스택의 틀을 만듬.

사용 문법 : ENTER imm, imm ; C8 iw ib [186]

기능 : ENTER는 스택의 틀을 고급 언어의 절차로 구성합니다. 첫 번째 피연산자는 지역 변수를 위한 스택 공간의 양을 할당하고, 두 번째 피연산자는 절차의 수준을 지정해줍니다.

동작 과정 :

ENTER

( = PUSH EBP

MOV EBP, ESP

SUB ESP, operand1 (첫 번째 피연산자) )

Leave : 스택의 틀을 파괴함.

사용 문법 : LEAVE ; C9 [186]

기능 : LEAVE는 ENTER 명령어로부터 만들어진 방식의 스택의 틀을 파괴합니다.

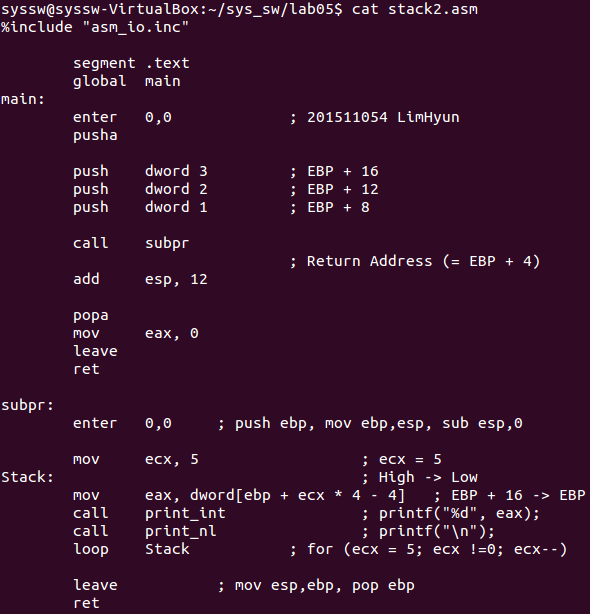
동작 과정 :

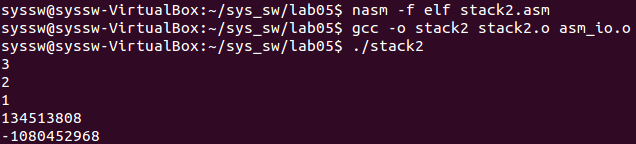
LEAVE

( = MOV ESP, EBP

POP EBP )

* + 1. 위 1.2에서 작성한 프로그램 stack.asm 에서 해당 부분을 enter 와 leave 명령어로 대체한 프로그램 stack2.asm을 작성하시오. 이 프로그램을 어셈블, 링크하여 실행 파일 stack2를 생성하고, 이를 실행시킨 결과를 화면 캡쳐, 첨부하여 동일하게 동작하는지 확인하시오. 또한 stack.asm 과 stack2.asm 두 프로그램 중 어느 것이 더 효율적인지 생각해보고 아래에 의견을 적으시오.





stack.asm 과 stack2.asm 두 프로그램 중 더 효율적인 것은 stack2.asm 이라고 생각합니다. ENTER 명령어가 고급 언어라서 컴퓨터의 관점에서 본다면 비효율적이겠지만 하드웨어 성능이 좋아진 만큼 사용자가 사용하기 쉬운 ENTER 명령어가 더 효율적이라고 생각합니다.

1. 다음 조건을 만족하는 어셈블리 프로그램 sum.asm을 작성하시오. 이 프로그램을 어셈블, 링크하여 실행 파일 sum을 생성하고, 이를 실행시킨 결과를 화면 캡쳐하여 첨부하시오.

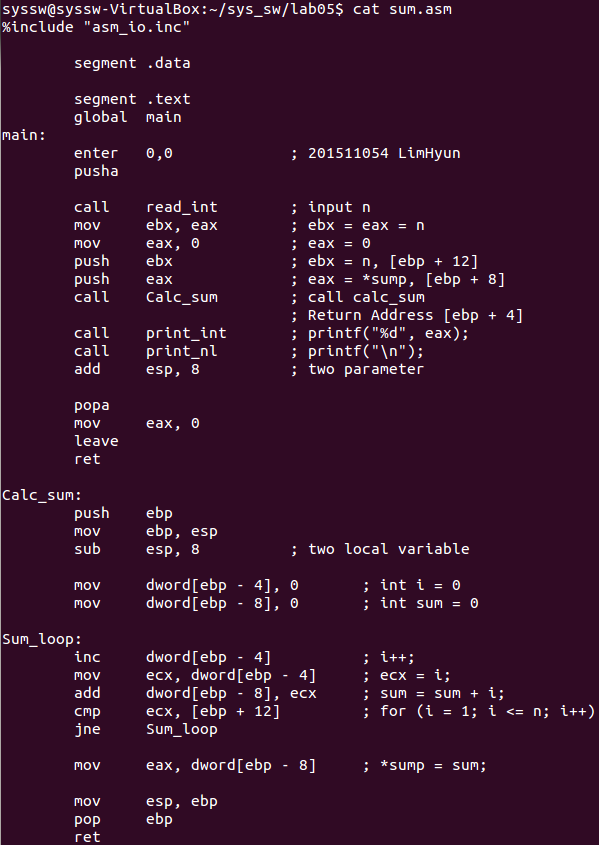
* 프로그램 sum 은 1 ~ n 의 정수들의 합을 계산하여 출력하는 프로그램이다. Sum 프로그램은 main 과 calc\_sum 두 개의 모듈로 구성된다.
* main 모듈은 (1) 사용자로부터 숫자 n 을 입력 받기, (2) 서브프로그램 calc\_sum 호출, (3) 결과값 출력 의 순서로 수행된다.
* calc\_sum 은 1 ~ n 의 덧셈을 계산하는 모듈이다. calc\_sum 은 main 에 의해 호출되며, 이 때 n 값과 합이 저장될 변수의 주소를 파라미터로 넘겨받는다. calc\_sum 은 수행 중에 두 개의 지역 변수를 사용하도록 한다 (아래 C 함수 참조)

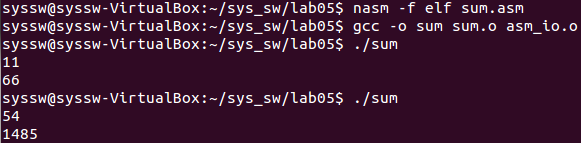
void calc\_sum( int n, int \* sump)  
{  
    int i, sum= 0;

    for (i=1; i<=n; i++){  
        sum = sum + i;

\*sump = sum;  
}

* calc\_sum 호출 시의 calling convention은 반드시 수업 시간에 다룬 내용을 따라야 한다.





끝.