기초적인 어셈블리어

# 기본 형태 : 명령어(opcode) + 인자(operand)

예 : push 0x123 = 0x123이라는 인자를 스택에 push(명령어)하라

예 : mov eax, 1 = eax 레지스터에 1을 복사하라

PUSH

스택에 값을 저장한다.

POP

스택에서 값을 빼 낸다.

MOV

값 혹은 주소가 가르키는 곳의 값을 가져옴.

LEA

주소 그 자체를 가져온다.

MOV : int a = \*ptr, LEA : int\* b = ptr

ADD

dest에 src만큼 값을 더한다.

SUB

dest에 src만큼 값을 뺀다.

INT

인터럽트(예외상황에 대한 일 처리)를 일으킨다.

INT3(0xCC)로 리버싱 시 멈춤기능 사용

MOV EAX, EBX 가 있다면

dest(목적지) : EAX

src(출발지) : EBX

CALL ([맨 윗 그림 참고하기](http://cr3denza.blogspot.com/2015/03/caller-callee.html))

함수(Callee)를 호출한다. (C 언어의 함수를 생각하면 쉽다.)

RET(Return)을 만나면 원래 함수(Caller)로 되돌아간다.

INC, DEC

i++, i--라고 생각하면 된다. 즉 증가와 감소

AND, OR, XOR

[비트연산](https://ko.wikipedia.org/wiki/%EB%B9%84%ED%8A%B8_%EC%97%B0%EC%82%B0)을 한다.

NOP

아무것도 하지 않는다.

CMP(Compare)

비교한다.

JMP 혹은 J\*

JMP는 무조건적인 점프, J\*(J 뒤에 다양한 문자들이 올 수 있음)은 조건에 해당하면 점프한다.