

5장 프로시저(1)

책의 라이브러리 사용

5장 전반부: 책의 링크 라이브러리

- 외부 링크 라이브러리 개요
- 라이브러리 프로시저 호출
- 라이브러리 링크
- 라이브러리 프로시저
- 예제

저자 제공 링크 라이브러리

■ 라이브러리 파일

- 어셈블된 프로시저를 포함하고 있는 OBJ 파일들을 모아놓은 파일 (확장자 .LIB)
- 각 OBJ file에는 하나 이상의 procedure가 들어있음
- LIB 명령어를 사용하여 library file을 생성하며 OBJ file들을 LIB파일에 추가하거나, LIB파일에서 삭제 가능

■ 저자 제공 라이브러리 (비표준)

- irvine32.lib 32-bit protected mode용 (make32로 어셈블)
- irvine16.lib 16-bit DOS 용 (make16으로 어셈블)
- 이 라이브러리에서 제공하는 함수에 대한 선언은 다음 파일에 포함됨
INCLUDE irvine32.inc ; protected mode
INCLUDE irvine16.inc ; real address mode

라이브러리 프로시저 호출

■ 프로시저 호출

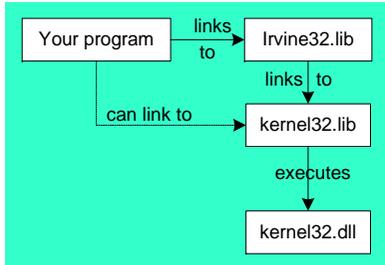
- CALL 명령어 사용
CALL procname
- 일부 procedure는 input argument를 필요로 함
 - argument는 register 또는 stack을 통해서 전달

■ 예: 화면에 1234를 출력

```
INCLUDE Irvine32.inc
.code
main proc
    mov eax,1234h           ; input argument
    call WriteHex          ; show hex number
    call Crlf              ; end of line
main endp
end main
```

라이브러리에 링크하기

- 사용자 프로그램을 library와 함께 링크하여 실행파일 생성
 - link `user.obj irvine32.lib kernel32.lib`
 - link명령어 실행은 `make32.bat`에 포함되어 있음
- `kernel32.lib`
 - MS-Windows OS에서 제공하는 `kernel32.dll`과의 bridge역할



교과서의 링크 라이브러리

Procedure	Description
<code>Clrscr</code>	Clears the console (커서는 왼쪽 상단에 위치)
<code>CrLf</code>	Writes CR-LF (new line)
<code>Delay</code>	Pauses execution for a <i>n</i> msec interval (EAX: delay시간)
<code>DumpMem</code>	Writes a block of memory to standard output in hex.
<code>DumpRegs</code>	Displays the EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP, EFLAGS, and EIP registers in hex and flags (CF,SF,ZF,OF)
<code>GetCommandtail</code>	Copies the program's command-line arguments into an array of bytes. (EDX: buffer주소)
<code>GetMaxXY(32)</code>	Get number of cols, rows in console window buffer (DH: 행, DL: 열)
<code>GetMseconds</code>	Returns # of milliseconds that have elapsed since midnight. (EAX: 결과)
<code>GetTextColor(32)</code>	Returns active foreground/background text colors in console. (AL: bgcolor 4 bits – fgcolor 4 bits)

Book's Libraries (계속)

Procedure	Description
<code>Gotoxy</code>	Locates cursor at row and column on the console. (DL: X좌표, 0-79, DH: Y좌표, 0-24)
<code>IsDigit</code>	Sets Zero flag if AL contains ASCII code for decimal digit
<code>MsgBox</code> <code>MsgBoxAsk</code>	Display popup message boxes (EBX: caption, EDX, msg) (EAX: 결과 6(yes), 7(no))
<code>ParseDecimal32</code>	Converts unsigned integer string to binary (EDX, ECX)
<code>ParseInteger32</code>	Converts signed integer string to binary (EDX, ECX)
<code>Random32</code>	Generates a 32-bit pseudorandom integer (EAX: 결과)
<code>Randomize</code>	Seeds the random number generator. (ECX: seed)
<code>RandomRange</code>	Generates a pseudorandom integer within a specified range (EAX: size입력, 결과 → range는 0 to size-1)
<code>ReadChar</code>	Reads a single character (AL: 결과)
<code>ReadHex</code>	Reads a 32-bit hex integer, terminated by Enter. (EAX: 결과)

Book's Libraries (계속)

Procedure	Description
<code>ReadDec</code> <code>ReadInt</code>	Reads a 32-bit unsigned/signed decimal integer, terminated by Enter. (EAX: 결과)
<code>ReadString</code>	Reads a string, terminated by Enter and insert a null terminator. (EDX: buffer주소, ECX: buffer의 크기)
<code>SetTextColor</code>	Sets the foreground and background colors of all subsequent text output to the console. (EAX: bgcolor 4 bits – fgcolor 4 bits)
<code>WaitMsg</code>	Displays message, waits for Enter key to be pressed.
<code>WriteBin</code>	Writes an unsigned 32-bit integer in binary format. (EAX: 정수)
<code>WriteBinB</code>	Writes binary integer in byte/word/doubleword format (EBX: 크기)
<code>WriteChar</code>	Writes a single character. (AL: 문자)
<code>WriteDec</code>	Writes an unsigned 32-bit integer in decimal format. (EAX: 정수)
<code>WriteHex</code>	Writes an unsigned 32-bit integer in hex format. (EAX: 정수)
<code>WriteInt</code>	Writes a signed 32-bit integer in decimal format. (EAX: 정수)
<code>WriteString</code>	Writes a null-terminated string. (EDX: buffer주소)

Example 1

- 화면을 지우고, 500 msec 지연 후, 레지스터와 플래그 출력

```
.code
main proc
    call Clrscr
    mov  eax,500
    call Delay
    call DumpRegs
main endp
end main
```

- 출력

```
EAX=00000613 EBX=00000000 ECX=000000FF EDX=00000000
ESI=00000000 EDI=00000100 EBP=0000091E ESP=000000F6
EIP=00401026 EFL=00000286 CF=0 SF=1 ZF=0 OF=0
```

Example 2

- 널종료 문자열 출력 후, 커서를 다음 줄 앞으로 이동

```
1 .data
   str1 BYTE "Assembly language is easy!",0
   .code
      mov  edx,OFFSET str1
      call WriteString
      call Crlf
```

```
2 ; use embedded CR and LF
   .data
   str1 BYTE "Assembly language is easy!",0Dh,0Ah,0
   .code
      mov  edx,OFFSET str1
      call WriteString
```

Example 3

- unsigned integer를 2진수, 10진수, 16진수로 분리된 줄에 각각 출력함

```
IntVal = 35
.code
    mov  eax,IntVal
    call WriteBin      ; display binary
    call Crlf
    call WriteDec      ; display decimal
    call Crlf
    call WriteHex      ; display hexadecimal
    call Crlf
```

- 출력

```
0000 0000 0000 0000 0000 0000 0010 0011
35
23
```

Example 4

- 문자열 입력하여 버퍼에 저장함

```
.data
fileName BYTE 80 DUP(0)

.code
    mov  edx,OFFSET fileName
    mov  ecx,SIZEOF fileName - 1
    call ReadString
```

- ReadString 프로시저는 널 바이트를 입력 문자열 뒤에 자동적으로 추가함

Example 5

- 0 ~ 99 범위에 있는 10개의 random integer를 생성하여 출력

```
.code
    mov ecx,10           ; loop counter
L1: mov  eax,100         ; ceiling value
    call RandomRange    ; generate random int
    call WriteInt       ; display signed int
    call Crlf          ; goto next display line
    loop L1             ; repeat loop
```

eax

- 50 ~ 49 범위에 있는 10개의 random integer를 생성하여 출력

```
.code
    mov ecx,10           ; loop counter
L1: mov  eax,100         ; ceiling value
    call RandomRange    ; generate random int
    sub  eax,50
    call WriteInt       ; display signed int
    call Crlf          ; goto next display line
    loop L1             ; repeat loop
```

Example 6

- 파란 배경에 노란색 글씨로 널종료 문자열을 출력

- 배경색은 상위 4비트 (16을 곱하여 표시), 전경색(글씨색)은 하위 4비트 (그대로 사용) 에 나타냄.

```
.data
str1 BYTE "Color output is easy!",0
.code
    mov  eax, yellow + (blue * 16)
    call SetTextColor
    mov  edx, OFFSET str1
    call WriteString
    call Crlf
```

Example 7

- 성능 타이밍 - 특정 부분의 소요시간 측정

- 종료시점과 시작시점의 자정이후 경과시간의 차이를 계산

```
...
.code
    ...
    call GetMSeconds
    mov  startTime, eax
    ...
    call GetMSeconds
    sub  eax, startTime    ; Elapsed msec
    mov  edx, offset msg
    call WriteString
    call WriteDec
    call Crlf
```