# Ch08 short Answer + Algorithm workbench

## Short Answer

1. Pop, ret
2. EAX레지스터
3. 프로시저 끝에서 ret이 정리한다
4. Offset = 고정된 주소값 / lea = 현재 레지스터 값들로 동적인 주소까지 계산
5. 4바이트
6. 1 : 가변 인자 함수 지원 / 2: 함수 포인터 유연성
7. False
8. True
9. Treu

## Algorithm Workbench

1. 스택 모양

```
ESP -> [old EBP]
       [RetrunAddress]
       [30h]
       [20h]
       [10h]
```

2. .

```
AddThree Proto
        a : Dword
        b : DWORD
        C : DWORD
.code
AddThree PROC
        a : DWORD
        b : DWORD
        c : DWORD

        mov eax, a
        mov ebx, b
        mov ecx, c
        ret 12
```

3.

```
MyProc PROC
        LOCAL pArray : PTR DWORD
MyProc ENDP
```

4.

```
MyProc PROC
        LOCAL buffer[20]:BYTE
MyProc ENDP
```

5.

```
MyProc PROC
   LOCAL pwArray:PTR WORD
   ; ...
MyProc ENDP
```

6.
```
MyProc PROC
   LOCAL myByte:SBYTE
   ; ...
MyProc ENDP
```

7.
```
MyProc PROC
   LOCAL myArray[20]:DWORD
   ; ...
MyProc ENDP
```

8.
```
SetTextColor PROTO colorAttr:BYTE

SetColor PROC,
   forecolor:BYTE,
   backcolor:BYTE

   ; colorAttr = (backcolor << 4) | forecolor
   movzx eax, backcolor   ; EAX = backcolor (0~15)
   shl   eax, 4           ; EAX = backcolor * 16
   movzx ecx, forecolor   ; ECX = forecolor (0~15)
   or    eax, ecx         ; EAX = (back << 4) | fore

   INVOKE SetTextColor, al  ; 하위 바이트만 사용

   ret 8  ; 파라미터 2개 * 4바이트(STDCALL 가정이면)

SetColor ENDP
```

9.
```
SetTextColor PROTO colorAttr:BYTE
WriteChar    PROTO

WriteColorChar PROC USES eax ecx edx,
   ch:BYTE,
   forecolor:BYTE,
   backcolor:BYTE

   ; 1) 색 속성 만들기
   movzx eax, backcolor
   shl   eax, 4
   movzx ecx, forecolor
   or    eax, ecx
   INVOKE SetTextColor, al

   ; 2) 한 글자 출력
   movzx edx, ch
   mov   al, dl       ; AL에 문자 값
   call  WriteChar

   ret 12             ; 3개 * 4바이트

WriteColorChar ENDP
```

10.
```
DumpMem PROTO,
    pData:PTR BYTE,
    count:DWORD,
    unitSize:DWORD

DumpMemory PROC USES esi,
    pArray:PTR BYTE,
    count:DWORD,
    unitSize:DWORD

    INVOKE DumpMem, pArray, count, unitSize

    ret 12      ; 3개 * 4바이트

DumpMemory ENDP
```

11.
```
; 프로토타입 선언
MultArray PROTO,
    pArray1:PTR DWORD,
    pArray2:PTR DWORD,
    count:DWORD

; 실제 프로시저 선언
MultArray PROC,
    pArray1:PTR DWORD,
    pArray2:PTR DWORD,
    count:DWORD

    ; 여기에서 둘 다 사용해서 곱셈/연산을 한다고 가정
    ; 예: pArray1[i] *= pArray2[i] 같은 작업
    ; ...
    ret 12      ; 파라미터 3개 * 4바이트

MultArray ENDP
```