```
Calc1   PROC STDCALL    USES eax ebx,
        lpBuf:PTR BYTE
    LOCAL dVal:DWORD

    mov esi, lpBuf
    mov dVal, eax
    ;.
    ;|.
    ret
Calc1   ENDP

Calc2   PROC
    push ebp
    mov ebp,esp
    add esp,0FFFFFFFCh; sub esp,4
    push    eax
    push    ebx
    mov esi, [ebp+8]
    mov [ebp-4],eax
    pop ebx
    pop eax
    leave
    ret 4
Calc2   ENDP
```

With stdcall (Windows API) the called program cleans up the stack for the passed parameters, by modifying the RET statement = n*4 where n = # of parameters

```
0x00401105 55 8b ec 83 c4 fc 50 53 8b 75 08 89 45 fc 5b 58 U.....PS.u..E.[X
0x00401115 c9 c2 04 00 55 8b ec 83 c4 fc 50 53 8b 75 08 89 ....U.....PS.u..
```

```
0x00401119 55 8b ec 83 c4 fc 50 53 8b 75 08 89 45 fc 5b 58 U.....PS.u..E.[X
0x00401129 c9 c2 04 00 55 8b ec 53 56 8b 45 08 03 45 0c 5e ....U..SV.E..E.^
```

```
31 ArraySum     PROC uses esi ecx
32     mov eax,0
33 l1A:
34     add eax,[esi]
35     add esi, type dword
36     loop    l1a
37     ret
38 arraySum ENDP
39
40 ArraySum2   Proc
41     push esi
42     push ecx
43     mov eax,0
44 l1B: add eax,[esi]
45     add esi,type dword
46     loop l1B
47     pop ecx
48     pop esi
49     ret
50 ArraySum2 endp
```

Memory(0x4010cf)

```
0x004010CF 56 51 b8 00 00 00 00 03 06 83 c6 04 e2 f9 59 5e VQ............Y^
0x004010DF c3 56 51 b8 00 00 00 00 03 06 83 c6 04 e2 f9 59 .VQ............Y
```

Memory(0x4010e0)

```
0x004010E0 56 51 b8 00 00 00 00 03 06 83 c6 04 e2 f9 59 5e VQ............Y^
0x004010F0 c3 55 8b ec 53 56 8b 45 08 03 45 0c 5e 5b 5d c2 .U..SV.E..E.^[].
```