```
; Listing generated by Microsoft (R) Optimizing Compiler Version 19.12.25834.0

        TITLE   c:\K-State\cis450\Programs\MemoryTest\VC_x86_Exercise\VC_x86_Exercise.c
        .686P
        .XMM
        include listing.inc
        .model  flat

INCLUDELIB LIBCMT
INCLUDELIB OLDNAMES

PUBLIC  _x
_DATA   SEGMENT
_x      DD      02H
_DATA   ENDS
PUBLIC  _f
PUBLIC  _main
_BSS    SEGMENT
_y      DD      01H DUP (?)
_BSS    ENDS
; Function compile flags: /Odtp
; File c:\k-state\cis450\programs\memorytest\vc_x86_exercise\vc_x86_exercise.c
_TEXT   SEGMENT
_m1$ = -8                                               ; size = 4
_s1$ = -4                                               ; size = 2
_argc$ = 8                                              ; size = 4
_argv$ = 12                                             ; size = 4
_envp$ = 16                                             ; size = 4
_main   PROC
  00000 55              push    ebp
  00001 8b ec           mov     ebp, esp
  00003 83 ec 08        sub     esp, 8
  00006 8b 45 08        mov     eax, DWORD PTR _argc$[ebp]
  00009 89 45 f8        mov     DWORD PTR _m1$[ebp], eax        ; MM: m1 is 4 bytes (int)
  0000c b9 22 00 00 00  mov     ecx, 34                         ; 00000022H
  00011 66 89 4d fc     mov     WORD PTR _s1$[ebp], cx          ; MM: s1 is 2 bytes (signed or unsigned: not known)
  00015 0f bf 55 fc     movsx   edx, WORD PTR _s1$[ebp]         ; MM: s1 is signed  => short s1;
  00019 52              push    edx
  0001a 8b 45 f8        mov     eax, DWORD PTR _m1$[ebp]
  0001d 50              push    eax
  0001e e8 00 00 00 00  call    _f
  00023 83 c4 08        add     esp, 8
  00026 33 c0           xor     eax, eax
  00028 8b e5           mov     esp, ebp
  0002a 5d              pop     ebp
  0002b c3              ret     0
_main   ENDP
_TEXT   ENDS
; Function compile flags: /Odtp
; File c:\k-state\cis450\programs\memorytest\vc_x86_exercise\vc_x86_exercise.c
_TEXT   SEGMENT
_i1$ = -12                                              ; size = 4
_s2$ = -8                                               ; size = 2
_s1$ = -4                                               ; size = 2
_p1$ = 8                                                ; size = 4
_p2$ = 12                                               ; size = 4
_f      PROC
  00000 55              push    ebp
  00001 8b ec           mov     ebp, esp
  00003 83 ec 0c        sub     esp, 12                         ; 0000000cH
  00006 8b 45 08        mov     eax, DWORD PTR _p1$[ebp]        ; MM: p1 is 4 bytes (int)
  00009 03 45 0c        add     eax, DWORD PTR _p2$[ebp]        ; MM: p2 is 4 bytes (int)
  0000c 89 45 f4        mov     DWORD PTR _i1$[ebp], eax        ; MM: i1 is 4 bytes (int)
  0000f 8b 4d f4        mov     ecx, DWORD PTR _i1$[ebp]
  00012 03 4d 08        add     ecx, DWORD PTR _p1$[ebp]
  00015 66 89 4d fc     mov     WORD PTR _s1$[ebp], cx          ; MM: s1 is 2 bytes (signed/unsingned not known)
```



```
  00019 8b 55 f4        mov     edx, DWORD PTR _i1$[ebp]
  0001c 03 55 0c        add     edx, DWORD PTR _p2$[ebp]
  0001f 66 89 55 f8     mov     WORD PTR _s2$[ebp], dx          ; MM: s2 is 2 bytes (signed/unsigned not known)
  00023 a1 00 00 00 00  mov     eax, DWORD PTR _x
  00028 83 e8 01        sub     eax, 1
  0002b a3 00 00 00 00  mov     DWORD PTR _x, eax
  00030 83 3d 00 00 00
        00 00           cmp     DWORD PTR _x, 0
  00037 7e 17           jle     SHORT $LN2@f
  00039 0f b7 4d f8     movzx   ecx, WORD PTR _s2$[ebp]         ; MM: s2 is unsigned short
  0003d 51              push    ecx
  0003e 0f bf 55 fc     movsx   edx, WORD PTR _s1$[ebp]         ; MM: s1 is signed short
  00042 52              push    edx
  00043 e8 00 00 00 00  call    _f
  00048 83 c4 08        add     esp, 8
  0004b a3 00 00 00 00  mov     DWORD PTR _y, eax
$LN2@f:
  00050 0f bf 45 fc     movsx   eax, WORD PTR _s1$[ebp]
  00054 03 45 f4        add     eax, DWORD PTR _i1$[ebp]
  00057 0f b7 4d f8     movzx   ecx, WORD PTR _s2$[ebp]
  0005b 03 05 00 00 00
        00              add     eax, DWORD PTR _y
  00061 03 c1           add     eax, ecx
  00063 8b e5           mov     esp, ebp
  00065 5d              pop     ebp
  00066 c3              ret     0
_f      ENDP
_TEXT   ENDS
END
```