```
COMMENT !
Description: Sums all the gaps between successive array elements, using a loop and
indexed addressing
Author name: Koichi Nakata
Author email: kanakta595@insite.4cd.edu
Last modified date: February 29, 2024
Creation date: February 29, 2024
!

INCLUDE Irvine32.inc

.386
.model flat, stdcall
.stack 4096
ExitProcess PROTO, dwExitCode: dword

.data
array dword 0, 2, 5, 9, 10

.code
main proc
        mov esi, type array                     ; Index pointing to the second
element
        mov ecx, lengthof array         ; Counter operand
        dec ecx                                         ; Want to iterate from the
second element
        mov eax, array                          ; Assign the first element (eax will
be prev)
        mov ebx, 0                                      ; ebx = sum

L1:
        add ebx, array[esi]                     ; Anyway add the current value to
sum
        sub ebx, eax                            ; Subtract the previous element from
sum

        mov eax, ebx                            ; Update prev value
        add esi, type array                     ; Increment the index

        loop L1

        CALL DumpRegs

        INVOKE ExitProcess, 0
main endp
end main
```

```
Description: Copies all the elements from an unsigned word array into an unsigned
dword array
Author name: Koichi Nakata
Author email: kanakta595@insite.4cd.edu
Last modified date: February 29, 2024
Creation date: February 29, 2024
!

INCLUDE Irvine32.inc

.386
.model flat, stdcall
.stack 4096
ExitProcess PROTO, dwExitCode: dword

.data
origin word 11h, 22h, 33h, 44h, 55h
target dword 5 DUP(?)                           ; Prepare dword empty array

.code
main proc
        mov esi, 0                              ; Indexing operand for the original
array
        mov edi, 0                              ; Indexing operand for the target
array
        mov ecx, lengthof origin                ; Counter operand

L1:
        movzx eax, origin[esi]                  ; Zero extension is necessary
(small->big)
        mov target[edi], eax

        add esi, type origin                    ; Increment byte is different
        add edi, type target

        loop L1

        CALL DumpRegs

        INVOKE ExitProcess, 0
main endp
end main
```