```
;-----------------------------------------------------------------------------
; Description: Generates a bit string in EAX that represents members in SetX (a bit-mapped
;              set) that are not members of SetY.
; Author name: Koichi Nakata
; Author email: kanakta595@insite.4cd.edu
; Last modified date: March 28, 2024
; Creation date: March 28, 2024
;-----------------------------------------------------------------------------

INCLUDE Irvine32.inc

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

.data
SetX = 10001010b
SetY = 10000001b

.code
main PROC
        mov  eax, SetX          ; Stores a bit-mapped set of SetX
        xor  eax, SetY          ; Yields members different from EACH OTHER
        and. eax, SetX          ; Filters only members SetX holds

        call WriteBin           ; Displays the value in a bit string

        INVOKE ExitProcess, 0
main ENDP

END main
```

```
;-----------------------------------------------------------------------------
; Description: Implements compound condition AND, using conditional jmp
;                           (val1 > ecx && ecx > edx) ? X = 1 : X = 2
; Author name: Koichi Nakata
; Author email: kanakta595@insite.4cd.edu
; Last modified date: March 28, 2024
; Creation date: March 28, 2024
;-----------------------------------------------------------------------------

INCLUDE Irvine32.inc

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

.data
val1 dword 1212h
X.   dword ?
msg1 byte  "X: ", 0
msg2 byte  "val1: ", 0
msg3 byte  "ECX: ", 0
msg4 byte  "EDX: ", 0

.code
main PROC
        mov  ecx, 1211h
        mov  edx, 1111h

        cmp  val1, ecx          ; Compares val1 and ecx
        jbe  L1                 ; Jumps to L1 if val1 <= ecx
        cmp  ecx, edx           ; Compares ecx and edx
        jbe  L1                 ; Jumps to L1 if ecx <= edx
        mov  X, 1               ; X = 1 if (val1 > ecx) AND (ecx > edx)
```

```
        jmp  next               ; Want to skip L1

L1: mov  X, 2                   ; Else block

next:
        call displayX
        INVOKE ExitProcess, 0

main ENDP


;-----------------------------------------------------------------------------
displayX PROC
; Displays the contents in X, val1, ecx and edx
; Receives: void
; Returns: void
; Remarks: Save the value edx to the stack
;-----------------------------------------------------------------------------
        push edx                ; Need to save the value in edx
        mov  edx, offset msg1
        call WriteString
        mov  eax, X
        call WriteDec
        call Crlf

        mov  edx, offset msg2
        call WriteString
        mov  eax, val1
        call WriteDec
        call Crlf

        mov  edx, offset msg3
        call WriteString
        mov  eax, ecx
        call WriteDec
        call Crlf
```

```
        mov  edx, offset msg4
        call WriteString
        pop  edx                ; Restores the value of edx from the stack
        mov  eax, edx
        call WriteDec
displayX ENDP

END main
```