```
INCLUDE Irvine32.inc
.data
key BYTE "ABXmv#7"
text BYTE 50 DUP(0)      ;輸入緩衝區
byteCount DWORD ?
start BYTE "please key in the plain text: ",0
after BYTE "Cipher text: ",0
decrypt BYTE "Decrypted: ",0
```

等等會用到的 data 資料~~

```
.code
main PROC

    mov edx,OFFSET start     ;將字串移至edx暫存器中
    call WriteString         ;顯示在edx暫存器中的字串
    mov edx,OFFSET text      ;緩衝區的位移
    mov ecx,(SIZEOF text)-1      ;指定所輸入字元的最大數量
    call ReadString          ;輸入字串
    mov byteCount,eax        ;存放所鍵入的字元數量
```

輸入字串流程!!!從 edx 開始放入 ecx 放 buffer 區能放的最大數量

並且會將字元數量放在 eax!!

```
    mov edx,0                ;紀錄已經處理幾個字元
    mov esi,OFFSET text
    mov ebx,OFFSET key
```

用 edx 紀錄 esi 放 buffer 位置 ebx 放 key 位置

```
mov ebx,OFFSET key
mov ecx,LENGTHOF key     ;有幾少key就輪流做幾次
L1:
    mov al,[esi]
    mov ah,[ebx]
    XOR al,ah            ;原本al,key:ah,做XOR
    mov [esi],al         ;把結果存回
    inc esi
    inc ebx
    inc edx
    cmp edx,byteCount    ;當全做完則跳出
    je next
    loop L1
mov ecx,LENGTHOF key     ;key做完卻還沒結束,重設ecx
mov ebx,OFFSET key       ;重設ebx回key首端
jmp L1                   ;回去繼續
```

以多少 key 為準，有多少就做多少(放在 ecx)，做完就不管其他的直接跳出，沒做完就再重設 key 相關初始值(位置:ebx, 迴圈次數:ecx)

```
next:
    mov edx,OFFSET after
    call WriteString
    mov edx,OFFSET text
    call WriteString
```

最後印出來~

```
    call Crlf

    mov esi,OFFSET text
    mov ebx,OFFSET key
    mov edx,0
    XOR eax,eax
    mov ecx,LENGTHOF key
    L2:
        mov al,[esi]
        mov ah,[ebx]
        XOR al,ah
        mov [esi],al
        inc esi
        inc ebx
        inc edx
        cmp edx,byteCount
        je next2
        loop L2
    mov ecx,LENGTHOF key
    mov ebx,OFFSET key
    jmp L2

    next2:
        mov edx,OFFSET decrypt
        call WriteString
        mov edx,OFFSET text
        call WriteString

exit
```

重複操作~