

● 程式碼與執行流程與記憶體狀態截圖

```

C:\Users\m0966\Downloads\windbg\windbg\hell

.data
    result BYTE 81 DUP(?)

.code
hi PROC uses ecx
    mov ecx,9
    mov eax,1
L:
    mul ebx
    mov result[esi],al
    mov edx,10
    sub edx,ecx
    mov eax,edx
    inc esi
    inc eax
    Loop L
    ret
hi ENDP

main PROC
    mov eax,0
    mov esi,0
    mov ebx,1
    mov ecx,9
L1:
    CALL hi
    inc ebx
    Loop L1
    exit
main ENDP
END main

```

| Memory(&result) | | | | | | | | | | |
|-----------------|----|----|----|----|----|----|----|----|----|-----------|
| 0x00403000 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | |
| 0x00403009 | 02 | 04 | 06 | 08 | 0a | 0c | 0e | 10 | 12 | |
| 0x00403012 | 03 | 06 | 09 | 0c | 0f | 12 | 15 | 18 | 1b | |
| 0x0040301B | 04 | 08 | 0c | 10 | 14 | 18 | 1c | 20 | 24 | |
| 0x00403024 | 05 | 0a | 0f | 14 | 19 | 1e | 23 | 28 | 2d |#(|
| 0x0040302D | 06 | 0c | 12 | 18 | 1e | 24 | 2a | 30 | 36 |\$*0 |
| 0x00403036 | 07 | 0e | 15 | 1c | 23 | 2a | 31 | 38 | 3f |#*18 |
| 0x0040303F | 08 | 10 | 18 | 20 | 28 | 30 | 38 | 40 | 48 | ... (08@ |
| 0x00403048 | 09 | 12 | 1b | 24 | 2d | 36 | 3f | 48 | 51 | ...\$-6?H |
| 0x00403051 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x0040305A | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x00403063 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x0040306C | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x00403075 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |
| 0x0040307E | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | |

● 程式執行流程說明：

一開始我們在 main PROC 裡我們將幾個值初始化。

然後我們使用 L 與 hi 做出雙重迴圈，讓他們分別做 9*9 次的計算

因此我們設了兩次的 ecx 為 9。

在迴圈進行前，我們將 eax 設成 1,ecx 設成 9

將 eax*ebx 輸進 eax，再輸進去 result 的地址裡。

ebx 是用來 x*y 的 x 的值，edx 是用 eax 去扣掉 ecx 剩餘要執行的次數得到現在

進行到該列的第幾個數字，紀錄到 `eax` 裡，再將 `eax` 跟 `esi` 都加 1。
依序進行，執行完一個迴圈後再進行下一個迴圈，於是進行了 81 次。
最後印出 9*9 類似方陣的數字。程式結束。

心得：

我們發現當若沒有加 `edx` 去紀錄 `eax` 的話，`eax` 的值還是原本已經乘 `bx` 的值，
然後 `eax` 就會變得越來越大，大到超過原本應該出現的值。
因此要有 `edx` 去紀錄 `eax` 的值，以防超過那個值越來越大。讓每次都能夠回復
到正常值，進行符合題意的 9*9。