



國立台灣科技大學

微 算 機 概 論 實 習

指 導 教 授：陸敬互 教 授

---

# 微算機概論實習報告 (原創)

## 期末報告

班 級       ： 四電機二乙  
學 生       ： 陳○○、李○○  
學 號       ： B110300XX、B110300XX  
建檔日期   ： 2022/12/14

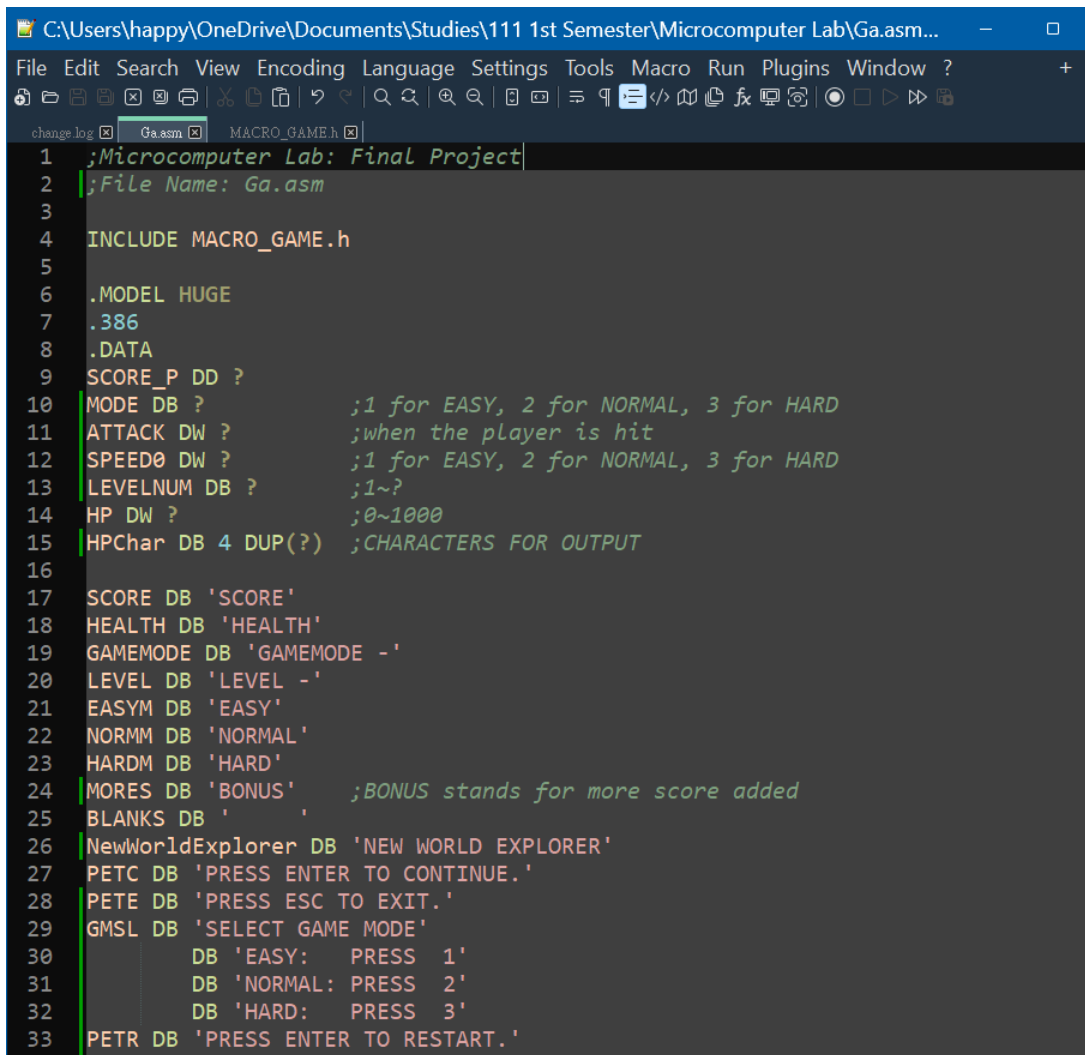
(中文字型:標楷體 英文字型:Times New Roman)

一、學習成果(程式功能說明，說明程式分幾個部分，各部分在做甚麼，搭配截圖code說明)

**遊戲名稱：新世界探索者 (New World Explorer)**

● 變數

i. 介面與字串



```
1 ;Microcomputer Lab: Final Project
2 ;File Name: Ga.asm
3
4 INCLUDE MACRO_GAME.h
5
6 .MODEL HUGE
7 .386
8 .DATA
9 SCORE_P DD ?
10 MODE DB ? ;1 for EASY, 2 for NORMAL, 3 for HARD
11 ATTACK DW ? ;when the player is hit
12 SPEED0 DW ? ;1 for EASY, 2 for NORMAL, 3 for HARD
13 LEVELNUM DB ? ;1~?
14 HP DW ? ;0~1000
15 HPChar DB 4 DUP(?) ;CHARACTERS FOR OUTPUT
16
17 SCORE DB 'SCORE'
18 HEALTH DB 'HEALTH'
19 GAMEMODE DB 'GAMEMODE -'
20 LEVEL DB 'LEVEL -'
21 EASYM DB 'EASY'
22 NORMM DB 'NORMAL'
23 HARDM DB 'HARD'
24 MORES DB 'BONUS' ;BONUS stands for more score added
25 BLANKS DB ' '
26 NewWorldExplorer DB 'NEW WORLD EXPLORER'
27 PETC DB 'PRESS ENTER TO CONTINUE.'
28 PETE DB 'PRESS ESC TO EXIT.'
29 GMSL DB 'SELECT GAME MODE'
30 DB 'EASY: PRESS 1'
31 DB 'NORMAL: PRESS 2'
32 DB 'HARD: PRESS 3'
33 PETR DB 'PRESS ENTER TO RESTART.'
```

```

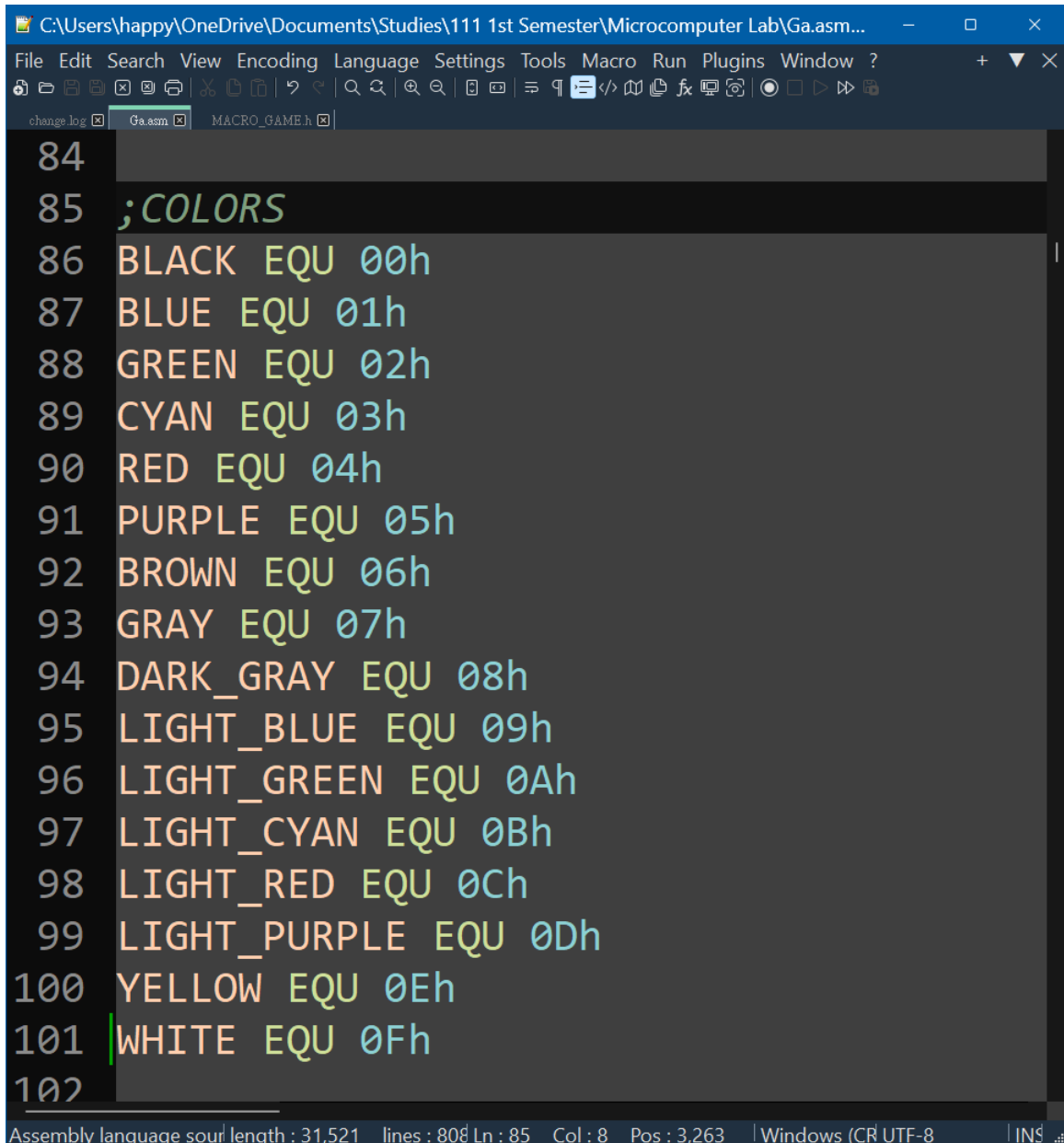
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change log Ga.asm MACRO_GAME.h
35 GAMEBGINTRO DB '      In the months leading up to      ', 0Ah, 0Dh
36             DB '      the Apocalypse...'           ', 0Ah, 0Dh
37             DB '                                     ', 0Ah, 0Dh
38             DB '                                     ', 0Ah, 0Dh
39             DB '      some initiated a project...'           ', 0Ah, 0Dh
40                                                     ;more and more people have escaped...
41             DB '                                     ', 0Ah, 0Dh
42             DB '                                     ', 0Ah, 0Dh
43             DB '                                     ', 0Ah, 0Dh
44             DB '                                     ', 0Ah, 0Dh
45             DB '                                     ', 0Ah, 0Dh
46             DB '                                     ', 0Ah, 0Dh
47             DB '                                     ', 0Ah, 0Dh
48             DB '                                     ', 0Ah, 0Dh
49             DB '                                     ', 0Ah, 0Dh
50             DB '                                     ', 0Ah, 0Dh
51             DB '                                     ', 0Ah, 0Dh
52             DB '                                     ', 0Ah, 0Dh
53             DB '      Press any key to join      ', 0Ah, 0Dh
54
55 NOWY DB '      Now you are part of the      ', ;LEN = 38
56 NWE DB '      New World Explorer !'
57
58 GAMEINTROLINE DB '-----$'
59
60 FIGUREINTRO DB '      INTRO      ', 0Ah, 0Dh
61             DB '                                     ', 0Ah, 0Dh
62             DB '      PLAYER:      METEORITE:      ', 0Ah, 0Dh
63             DB '                                     ', 0Ah, 0Dh
64             DB '                                     ', 0Ah, 0Dh
65             DB '                                     $', 0Ah, 0Dh
66
67 GAMECNTRL DB '      CONTROL      ', 0Ah, 0Dh
68             DB '      W for going upward      ', 0Ah, 0Dh
69             DB '      A for going backward      ', 0Ah, 0Dh
70             DB '      S for going downward      ', 0Ah, 0Dh
71             DB '      D for going forward      ', 0Ah, 0Dh
72             DB '                                     ', 0Ah, 0Dh
73             DB '      Score as high as U can :)      ', 0Ah, 0Dh
74             DB '      Good Luck      ', 0Ah, 0Dh
75             DB '                                     $', 0Ah, 0Dh
76
77 AUTHOR DB '      BY B11030010 and B11030040      ', 0Ah, 0Dh
78
79 GAMEOV DB 'GAME OVER'
80 END_SCORE DB 'YOU SCORED      ', ;8 blanks for score
81 NICEW DB 'NICE WORK!'
82 WLPL DB 'WELL-PLAYED!'
83 INCRE DB 'INCREDIBLE!'
Assembly language sour length : 31,521 lines : 808 Ln : 35 Col : 66 Pos : 873 | Windows (CR UTF-8 | INS

```

此區大多為用於介面的字串，其中包括了一些功能性的變數，因為分數採用double word的資料型態，因此加入了.386

## ii. 圖案

### 1. 顏色

A screenshot of an assembly language editor window. The title bar shows the file path: C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm... The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations, editing, and execution. The editor has three tabs: change.log, Ga.asm (active), and MACRO\_GAME.h. The code is displayed on a dark background with syntax highlighting. Line numbers 84 through 102 are visible on the left. The code defines sixteen color constants using the EQU directive. The status bar at the bottom shows: Assembly language sour length : 31,521 lines : 808 Ln : 85 Col : 8 Pos : 3,263 | Windows (CR UTF-8 | IN\$ ..:

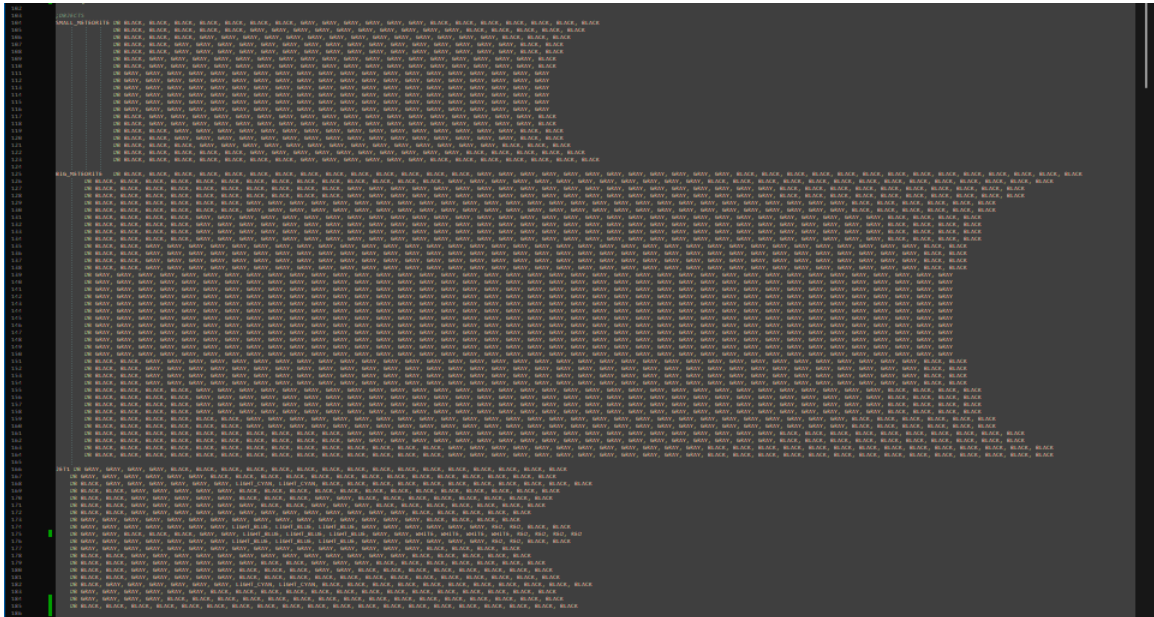
```

84
85 ;COLORS
86 BLACK EQU 00h
87 BLUE EQU 01h
88 GREEN EQU 02h
89 CYAN EQU 03h
90 RED EQU 04h
91 PURPLE EQU 05h
92 BROWN EQU 06h
93 GRAY EQU 07h
94 DARK_GRAY EQU 08h
95 LIGHT_BLUE EQU 09h
96 LIGHT_GREEN EQU 0Ah
97 LIGHT_CYAN EQU 0Bh
98 LIGHT_RED EQU 0Ch
99 LIGHT_PURPLE EQU 0Dh
100 YELLOW EQU 0Eh
101 WHITE EQU 0Fh
102

```

我們使用EQU賦予十六色各一個名稱（標籤），以提升可用性與易讀性

## 2. 小隕石、大隕石與玩家



這部分使用人工，將一個個色塊放進一陣列中，在遊戲中則以迴圈方式將圖案輸出。小隕石與玩家大小為 $20 \times 20 \text{ pixels}^2$ ，大隕石則為 $40 \times 40 \text{ pixels}^2$

### iii. 參數

```
187 ;Random Ys
188 YPOS DW 99,41,88,32,120,87,49,87,92,165,135,109,154,56,114,72,135,54,72,49,124,104,50,139,34,157,69,64,41,171,92,98
189 DW 82,52,177,62,84,161,64,47,84,97,118,102,35,175,47,121,119,63,38,159,72,152,170,165,177,83,165,83,107,97,41,57
190 DW 166,109,68,152,77,169,85,150,39,59,127,112,111,148,42,131,153,146,172,73,125,159,78,130,55,150,54,130,88,106
191 DW 90,105,147,180,101,121,57,177,111,63,131,60,156,97,116,149,33,63,151,86,107,34,115,141,129,180,141,51,57,84
192 DW 133,74,55,172,50,54,73,68,93,80,99,67,126,97,56,67,34,73,83,80,114,119,108,166,47,94,89,106,88,92,173,38,47
193 DW 141,127,76,72,132,109,169,138,38,82,60,116,163,59,144,145,154,32,122,56,160,36,63,89,116,155,104,62,51,152,102
194 DW 151,168,89,37,136,99,111,147,139,78,40,132
195
196 ;PLAYER POSITION
197 ;The dot is the (x, y)
198 ;
199 ;
200 ;
201 PLAYER_X DW ? ;right
202 PLAYER_Y DW ? ;top
203 STEPLENGTH EQU 15 ;pixels per step
204
205 SMALL_METEORITE_X DW ? ;The x and y of the obstacle
206 SMALL_METEORITE_Y DW ?
207 SSTA DW 0 ;Small M. Status
208 SCOUNTER DW 0050H ;Counter (Delay)
209 SFCOUNTER DW 0050H ;Reset Counter (Fixed)
210
211 YPOSINDEX DW 0H ;Y position index (A backup)
212
```

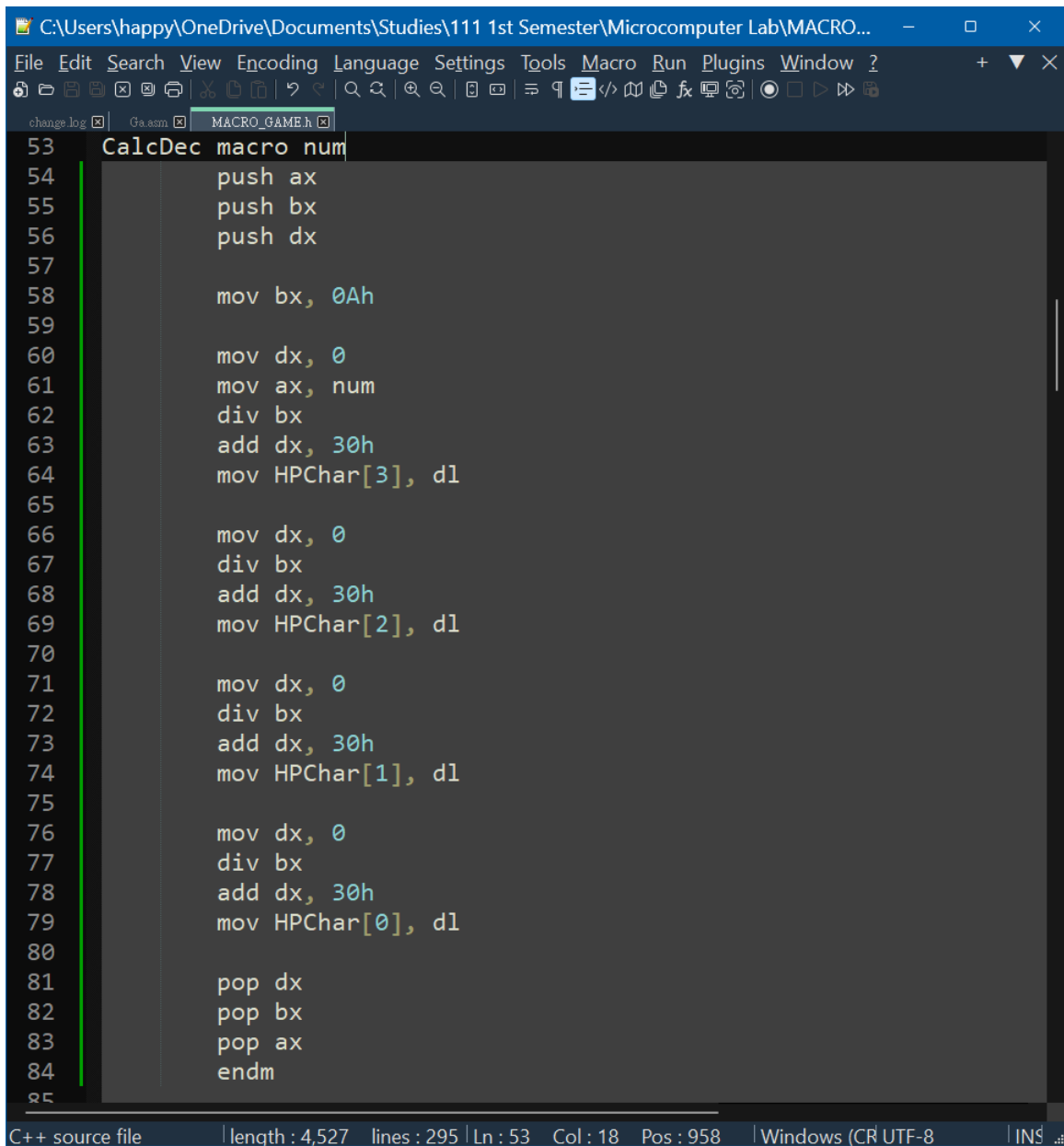
YPOS為一隨機數字之陣列，使障礙物能夠隨機出現，共200個，YPOSINDEX為YPOS的索引。玩家與障礙物都各有一組位置(x,y)，代表其圖案之左上方的那個像素，而玩家每一步的移動（STEPLENGTH）為15個像素，COUNTER與STA分別用於延遲與表示障礙物的存在與否，詳細內容在巨集與副程式的介紹中。

- 巨集：
  - i. 簡單的功能

```
1 ;File Name: MACRO_GAME.h
2
3 SetMode macro mode ;03h for text or 12h for graphics
4     mov ah, 00h
5     mov al, mode
6     int 10h
7     endm
8
9 SetColor macro color ;Set Background Color
10    mov ah, 0bh
11    mov bh, 00h
12    mov bl, color
13    int 10h
14    endm
15
16 SetCursor macro col, row ;Set Cursor(Position)
17    push dx
18    push bx
19    mov dh, row
20    mov dl, col
21    mov bx, 00h
22    mov ah, 02h
23    int 10h
24    pop bx
25    pop dx
26    endm
27
28 PrintStr macro string ;Print String
29    mov ah, 09h
30    mov dx, offset string
31    int 21h
32    endm
33
34 PrintChar macro char ;Print Char
35    mov ah, 02h
36    mov dl, char
37    int 21h
38    endm
39
40 GetKey macro ;This does not wait
41    push dx
42    mov ah, 06h
43    mov dl, 0ffh
44    int 21h
45    pop dx
46    endm
47
48 WaitKey macro
49    mov ah, 10h
50    int 16h
51    endm
52
```

這些為基本的指令，在這邊不贅述

## ii. CalcDec 十六進制轉十進制



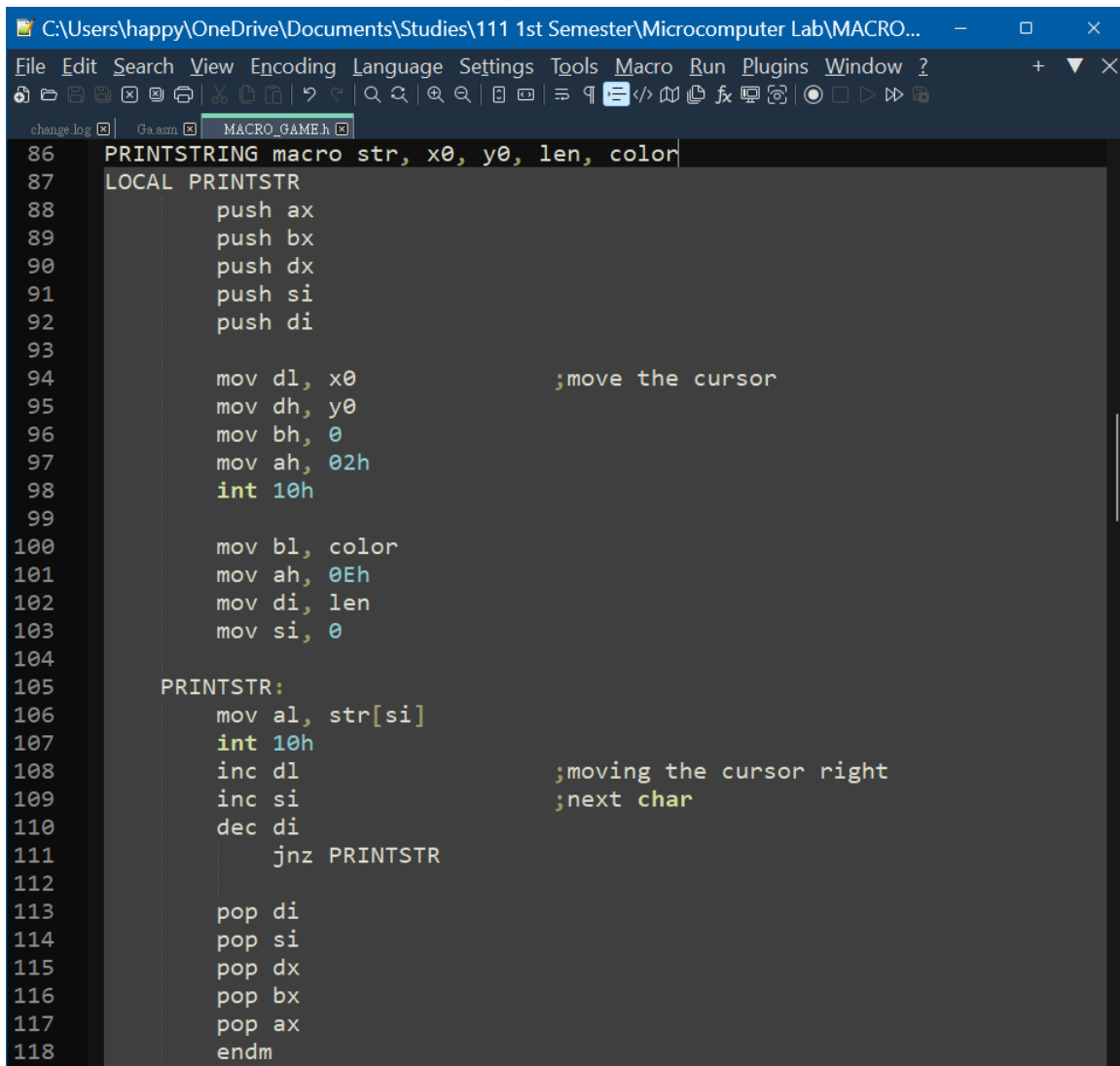
```
53 CalcDec macro num
54     push ax
55     push bx
56     push dx
57
58     mov bx, 0Ah
59
60     mov dx, 0
61     mov ax, num
62     div bx
63     add dx, 30h
64     mov HPChar[3], dl
65
66     mov dx, 0
67     div bx
68     add dx, 30h
69     mov HPChar[2], dl
70
71     mov dx, 0
72     div bx
73     add dx, 30h
74     mov HPChar[1], dl
75
76     mov dx, 0
77     div bx
78     add dx, 30h
79     mov HPChar[0], dl
80
81     pop dx
82     pop bx
83     pop ax
84     endm
85
```

C++ source file | length : 4,527 | lines : 295 | Ln : 53 | Col : 18 | Pos : 958 | Windows (CR UTF-8) | INS

這是將資料從十六進制轉為十進制的巨集，在遊戲中僅使用在轉換生命值



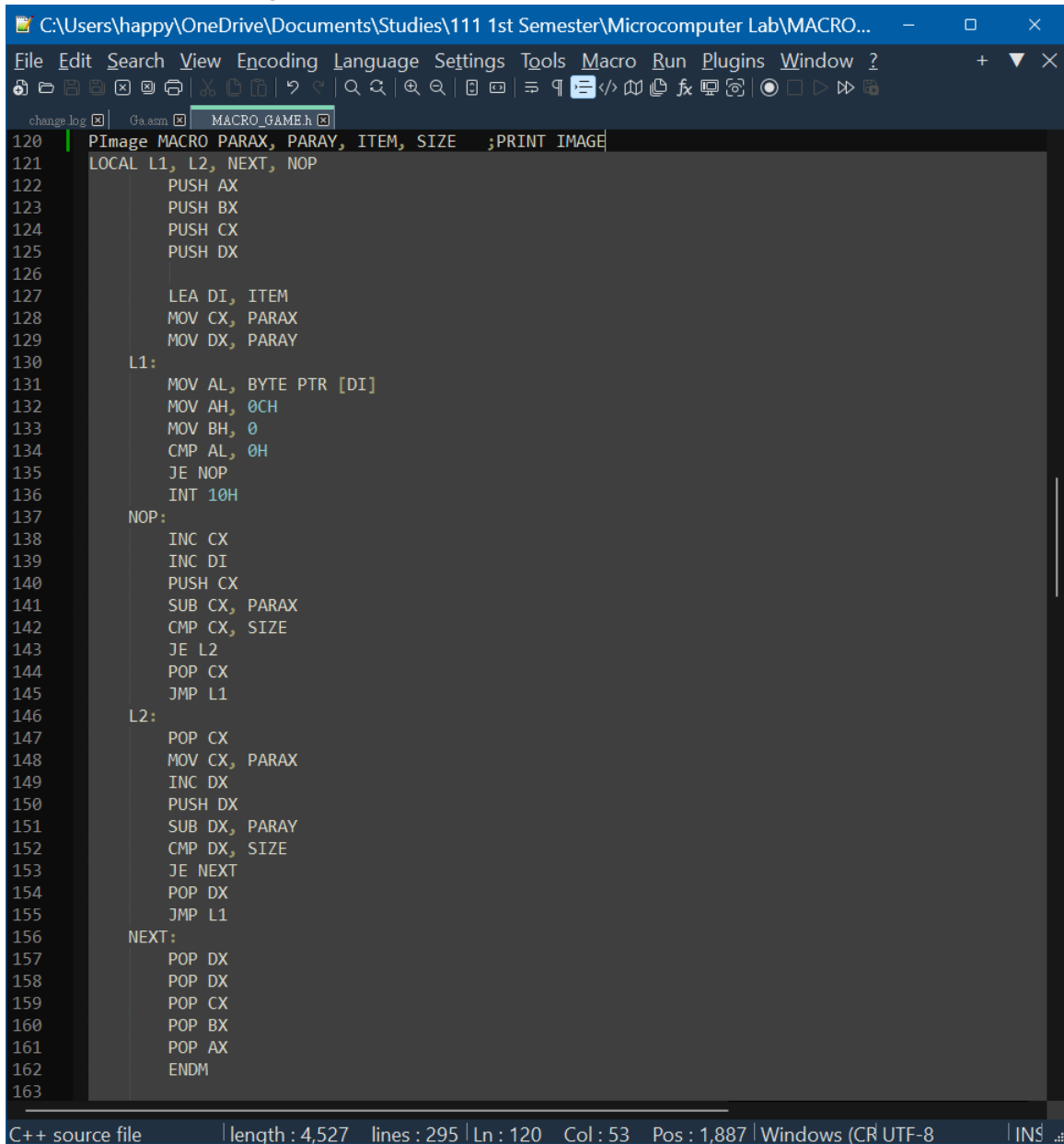
### iii. PRINTSTRING 印出有顏色的字串



```
86 PRINTSTRING macro str, x0, y0, len, color
87 LOCAL PRINTSTR
88     push ax
89     push bx
90     push dx
91     push si
92     push di
93
94     mov dl, x0           ;move the cursor
95     mov dh, y0
96     mov bh, 0
97     mov ah, 02h
98     int 10h
99
100    mov bl, color
101    mov ah, 0Eh
102    mov di, len
103    mov si, 0
104
105    PRINTSTR:
106        mov al, str[si]
107        int 10h
108        inc dl             ;moving the cursor right
109        inc si             ;next char
110        dec di
111        jnz PRINTSTR
112
113    pop di
114    pop si
115    pop dx
116    pop bx
117    pop ax
118    endm
```

這邊使用到了INT 10H的0EH功能，能輸入有顏色的字串

#### iv. PImage 印出圖案

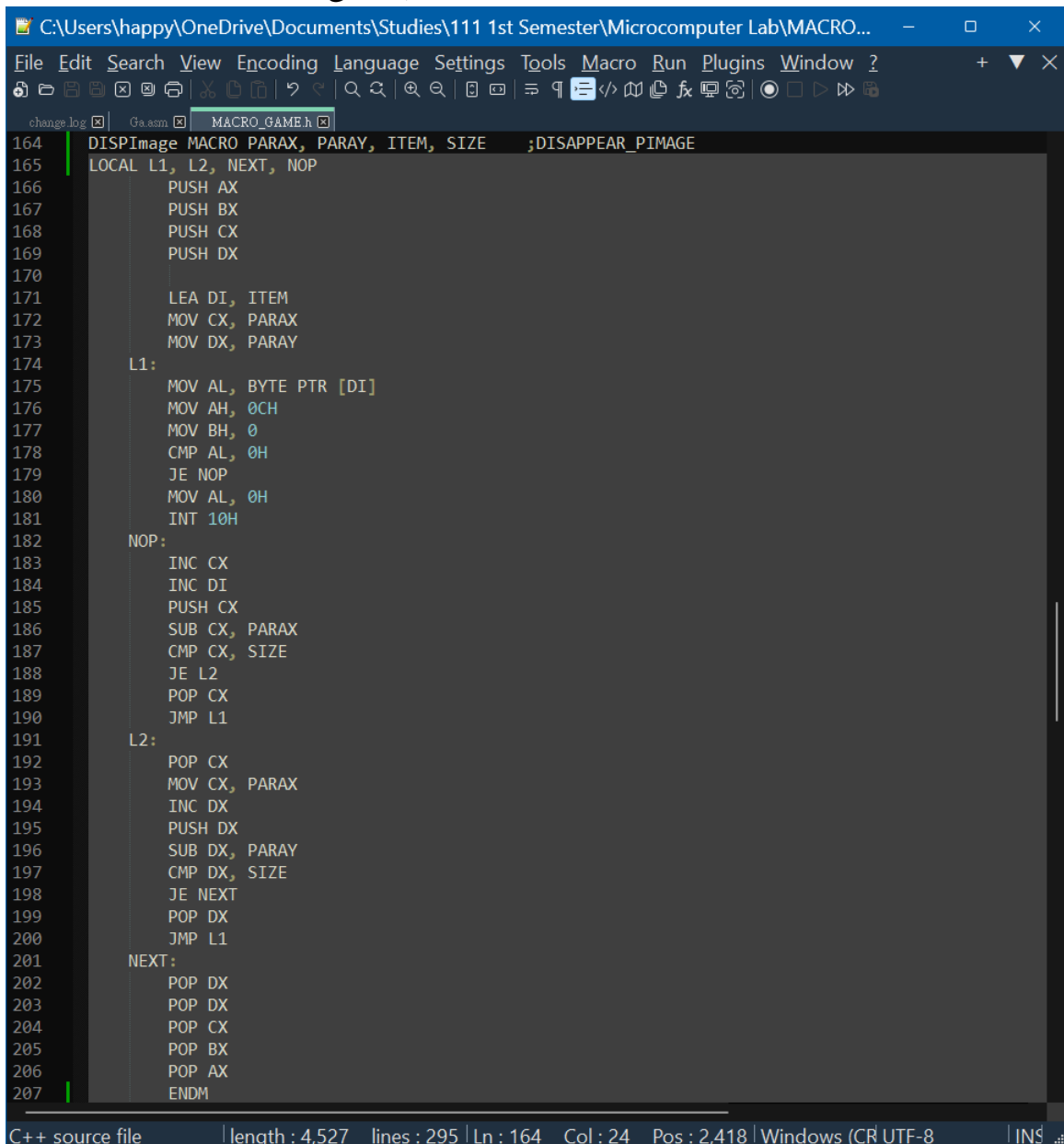


```
120 | PImage MACRO PARAX, PARAY, ITEM, SIZE ;PRINT IMAGE
121 | LOCAL L1, L2, NEXT, NOP
122 |     PUSH AX
123 |     PUSH BX
124 |     PUSH CX
125 |     PUSH DX
126 |
127 |     LEA DI, ITEM
128 |     MOV CX, PARAX
129 |     MOV DX, PARAY
130 |
131 | L1:
132 |     MOV AL, BYTE PTR [DI]
133 |     MOV AH, 0CH
134 |     MOV BH, 0
135 |     CMP AL, 0H
136 |     JE NOP
137 |     INT 10H
138 |
139 | NOP:
140 |     INC CX
141 |     INC DI
142 |     PUSH CX
143 |     SUB CX, PARAX
144 |     CMP CX, SIZE
145 |     JE L2
146 |     POP CX
147 |     JMP L1
148 |
149 | L2:
150 |     POP CX
151 |     MOV CX, PARAX
152 |     INC DX
153 |     PUSH DX
154 |     SUB DX, PARAY
155 |     CMP DX, SIZE
156 |     JE NEXT
157 |     POP DX
158 |     POP DX
159 |     POP CX
160 |     POP BX
161 |     POP AX
162 |     ENDM
163 |
```

C++ source file | length : 4,527 | lines : 295 | Ln : 120 | Col : 53 | Pos : 1,887 | Windows (CR UTF-8) | INS

此巨集將圖案印出，印出前會檢查欲印之色塊，若為黑色（玩家或障礙物的邊角，背景色的地方）則不執行中斷，以此印出乾淨的圖形

## v. DISPImage 印出黑色以清除圖案



```
164  DISPImage MACRO PARAX, PARAY, ITEM, SIZE ;DISAPPEAR_PIMAGE
165  LOCAL L1, L2, NEXT, NOP
166      PUSH AX
167      PUSH BX
168      PUSH CX
169      PUSH DX
170
171      LEA DI, ITEM
172      MOV CX, PARAX
173      MOV DX, PARAY
174
175  L1:
176      MOV AL, BYTE PTR [DI]
177      MOV AH, 0CH
178      MOV BH, 0
179      CMP AL, 0H
180      JE NOP
181      MOV AL, 0H
182      INT 10H
183
184  NOP:
185      INC CX
186      INC DI
187      PUSH CX
188      SUB CX, PARAX
189      CMP CX, SIZE
190      JE L2
191      POP CX
192      JMP L1
193
194  L2:
195      POP CX
196      MOV CX, PARAX
197      INC DX
198      PUSH DX
199      SUB DX, PARAY
200      CMP DX, SIZE
201      JE NEXT
202      POP DX
203      JMP L1
204
205  NEXT:
206      POP DX
207      POP DX
208      POP CX
209      POP BX
210      POP AX
211      ENDM
```

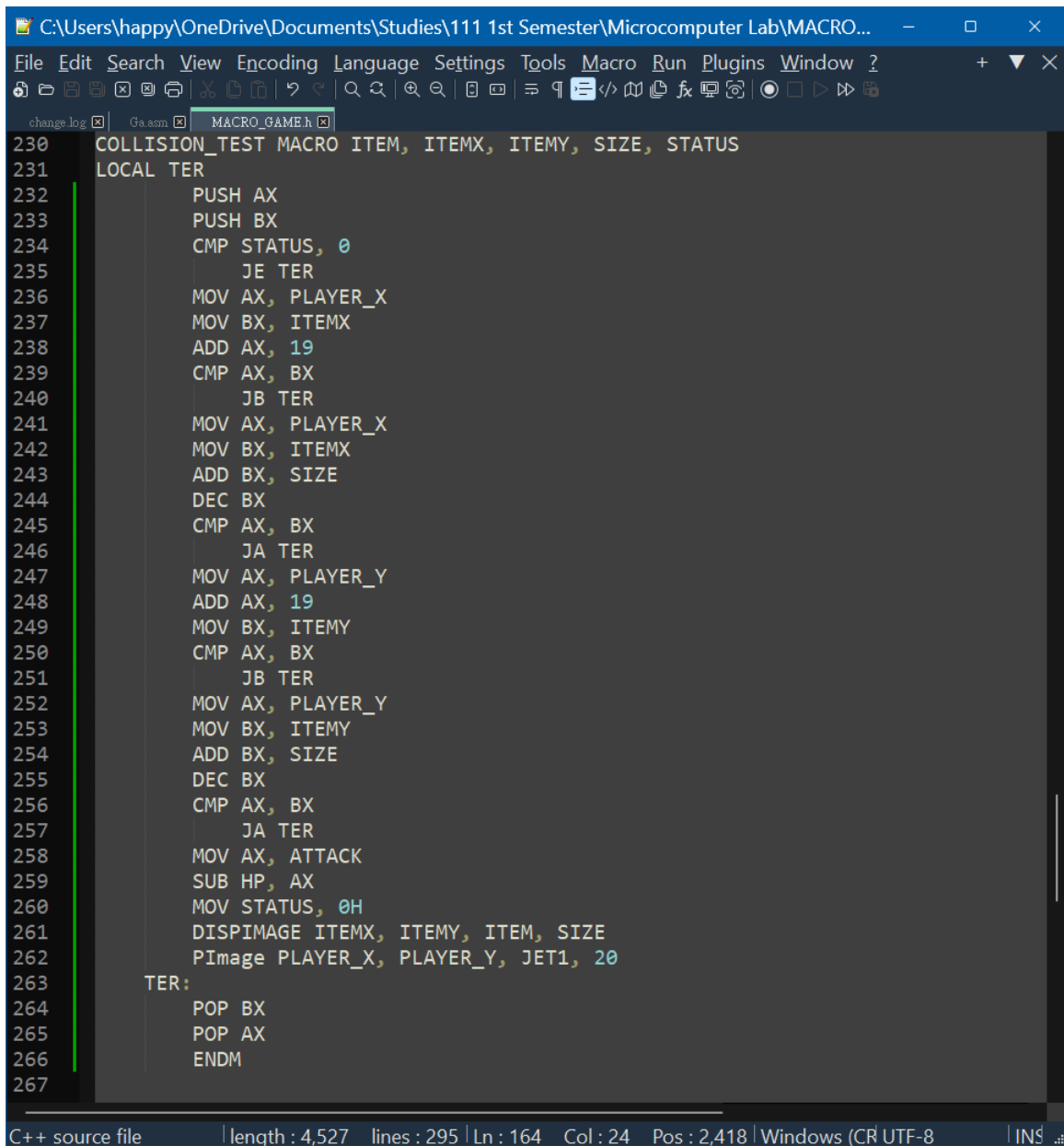
在圖案移動前，須將原圖案清除，原本我們以 $20 \times 20 \text{ pixels}^2$ 的黑色方格用於清除，進行改良後，就多了這個巨集，同時也微幅提升了程式的效率。此巨集與上一個印出圖案的巨集相似，差別僅在於印出的色塊只會是黑色，印在非黑色的位置，以蓋掉原先的圖案。

## vi. MOVEOBSTACLE 障礙物移動

```
209 MOVEOBSTACLE MACRO ITEM, ITEMX, ITEMY, SIZE, STEP, STA
210 LOCAL MOVE, EXIT, L1, KEEPMOVING
211     CMP STA, 0
212     JE EXIT
213     MOVE:
214     MOV SI, STEP
215     L1:
216     DISPIMAGE ITEMX, ITEMY, ITEM, SIZE
217     CMP ITEMX, 2
218     JA KEEPMOVING
219     MOV STA, 0
220     JMP EXIT
221     KEEPMOVING:
222     SUB ITEMX, 2           ;2 pixels per step
223     PIMAGE ITEMX, ITEMY, ITEM, SIZE
224     DEC SI
225     CMP SI, 0H
226     JA L1
227 EXIT:
228 ENDM
```

我們設置障礙物移動每步為2個像素，依照STEP輸入去做出移動速度。當障礙物不存在時，直接跳出巨集，不執行任何指令。當障礙物跑到畫面邊界時，就將障礙物之狀態設置為0並跳出巨集，因為在清除圖案後沒有再印上移動過後的圖案，所以障礙物到邊界時就會消失在畫面上。

## vii. COLLISION\_TEST 碰撞判定

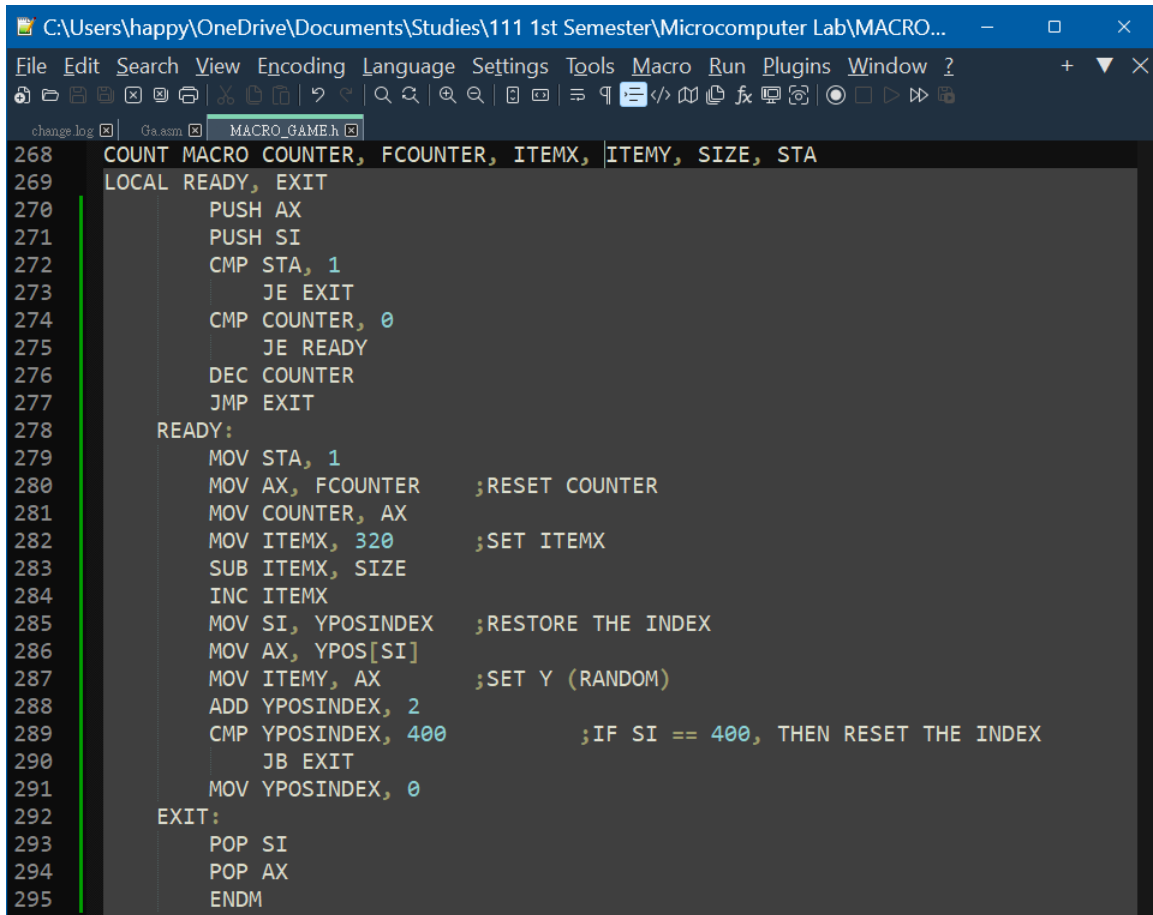


```
230 COLLISION_TEST MACRO ITEM, ITEMX, ITEMY, SIZE, STATUS
231 LOCAL TER
232     PUSH AX
233     PUSH BX
234     CMP STATUS, 0
235     JE TER
236     MOV AX, PLAYER_X
237     MOV BX, ITEMX
238     ADD AX, 19
239     CMP AX, BX
240     JB TER
241     MOV AX, PLAYER_X
242     MOV BX, ITEMX
243     ADD BX, SIZE
244     DEC BX
245     CMP AX, BX
246     JA TER
247     MOV AX, PLAYER_Y
248     ADD AX, 19
249     MOV BX, ITEMY
250     CMP AX, BX
251     JB TER
252     MOV AX, PLAYER_Y
253     MOV BX, ITEMY
254     ADD BX, SIZE
255     DEC BX
256     CMP AX, BX
257     JA TER
258     MOV AX, ATTACK
259     SUB HP, AX
260     MOV STATUS, 0H
261     DISPIIMAGE ITEMX, ITEMY, ITEM, SIZE
262     PImage PLAYER_X, PLAYER_Y, JET1, 20
263 TER:
264     POP BX
265     POP AX
266 ENDM
267
```

C++ source file | length : 4,527 | lines : 295 | Ln : 164 | Col : 24 | Pos : 2,418 | Windows (CR UTF-8) | INS

碰撞的產生源自於圖案重疊，那麼在一個個判斷的篩選後，仍沒被篩掉（跳出巨集）的情況，就是碰撞產生的情況了。原本我們對玩家的三個點（形狀近似於三角形）進行是否碰撞的判定，實作完成後我們發覺有太多重複且多餘的部分，因此我們進行改良，僅留下原本1/3長的程式碼，提升了程式效率，在權衡之下也犧牲了一些精確度。

### viii. COUNT 障礙物出現之延遲



```
268 COUNT MACRO COUNTER, FCOUNT, ITEMX, ITEM, SIZE, STA
269 LOCAL READY, EXIT
270     PUSH AX
271     PUSH SI
272     CMP STA, 1
273     JE EXIT
274     CMP COUNTER, 0
275     JE READY
276     DEC COUNTER
277     JMP EXIT
278 READY:
279     MOV STA, 1
280     MOV AX, FCOUNT      ;RESET COUNTER
281     MOV COUNTER, AX
282     MOV ITEMX, 320      ;SET ITEMX
283     SUB ITEMX, SIZE
284     INC ITEMX
285     MOV SI, YPOSINDEX   ;RESTORE THE INDEX
286     MOV AX, YPOS[SI]
287     MOV ITEM, AX        ;SET Y (RANDOM)
288     ADD YPOSINDEX, 2
289     CMP YPOSINDEX, 400   ;IF SI == 400, THEN RESET THE INDEX
290     JB EXIT
291     MOV YPOSINDEX, 0
292 EXIT:
293     POP SI
294     POP AX
295 ENDM
```

若在消失或碰撞後（障礙物不存在）又馬上出現新的障礙物，會讓節奏太緊張，因此我們加入了COUNT巨集，讓程式在障礙物不存在時開始倒數，等到COUNTER為零時才允許其出現，並賦予其初始位置、重設COUNTER，以達到延遲的效果。

## 關於SIZE

事實上，上面所看到的SIZE輸入本來是為了讓巨集能夠調整印出或判斷邊長為20或40的障礙物，以讓遊戲更豐富，但在測試時我們發現這在實現上有問題，因為印出四倍的色塊（ $1600 = 400 * 4$ ）實在太耗時間，導致移動速度受影響，變得很慢沒挑戰性，也讓球體看起來比較不完整（印很久），因此最終放棄了更大的障礙物。另外，本來我們想使多個障礙物出現於畫面上，不過也產生了同樣的問題，會有移動速度變慢、跑分變慢的問題，因此最終這兩項設計宣告失敗。

以下當時測試使用DOSBox執行遊戲的狀況

3000cycles是一般的執行速度，可以看到障礙物的移動上還頗尷尬的，也能感覺到沒有挑戰性。6000cycles則是兩倍的執行速度，很明顯的，被撞之後剩一顆的速度變得快許多，會打亂遊戲節奏，這些都是無法被克服的。



3000cycles.mp4



6000cycles.mp4

副程式：

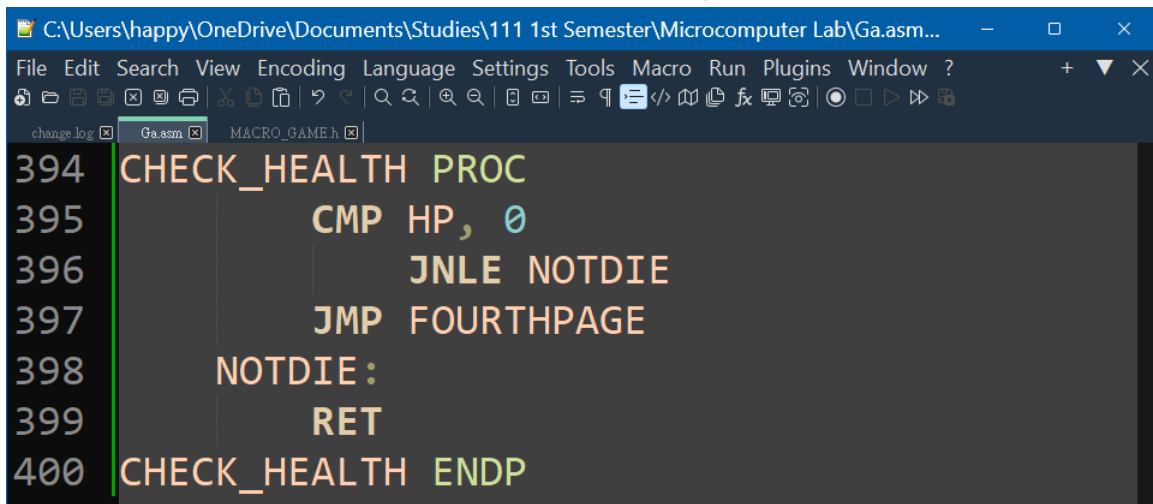
i. SPEEDUP 加速

```
331 SPEEDUP PROC
332     CMP MODE, 3
333     JE HARD
334     CMP MODE, 2
335     JE NORMAL
336     ;-----EASY
337     CMP SCORE_P, 1000H
338     JA ELEVEL2
339     RET
340 ELEVEL2:
341     MOV LEVELNUM, 32H
342     MOV SPEED0, 2
343     CMP SCORE_P, 1000H
344     JA ELEVEL3
345     RET
346 ELEVEL3:
347     MOV LEVELNUM, 33H
348     MOV SPEED0, 3
349     RET
350     ;-----NORMAL
351     NORMAL:
352     CMP SCORE_P, 1000H
353     JA NLEVEL2
354     RET
355 NLEVEL2:
356     MOV LEVELNUM, 32H
357     MOV SPEED0, 3
358     CMP SCORE_P, 2000H
359     JA NLEVEL3
360     RET
361 NLEVEL3:
362     MOV LEVELNUM, 33H
363     MOV SPEED0, 4
364     RET
365     ;-----HARD
366     HARD:
367     CMP SCORE_P, 2000H
368     JA HLEVEL2
369     RET
370 HLEVEL2:
371     MOV LEVELNUM, 32H
372     MOV SPEED0, 4
373     CMP SCORE_P, 3000H
374     JA HLEVEL3
375     RET
376 HLEVEL3:
377     MOV LEVELNUM, 33H
378     MOV SPEED0, 5
379     CMP SCORE_P, 4000H
380     JA HLEVEL4
381     RET
382 HLEVEL4:
383     MOV LEVELNUM, 34H
384     MOV SPEED0, 7
385     CMP SCORE_P, 5000H
386     JA HLEVEL5
387     RET
388 HLEVEL5:
389     MOV LEVELNUM, 35H
390     MOV SPEED0, 9
391     RET
392 SPEEDUP ENDP
393
```

這個巨集會判斷玩家的積分以決定是否加速。簡單模式與普通模式的最高等級為3，最高速度分別為3與4，困難模式的最高等級為5，最高速度則高達9。積分界定的依據是測試與感覺，因此並非線性。



## ii. CHECK\_HEALTH 確認生命值



```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
394 CHECK_HEALTH PROC
395     CMP HP, 0
396     JNLE NOTDIE
397     JMP FOURTHPAGE
398     NOTDIE:
399     RET
400 CHECK_HEALTH ENDP
```

一簡單的判斷，若玩家生命值非小於等於0時，則繼續遊戲，否則進入第四頁（遊戲結束）。

### iii. KEY\_ACTIONS 檢查按鍵

```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
402 KEY_ACTIONS PROC USES ax bx cx dx
403     ;IDENTIFY THE KEY PRESSED
404     CMP AL, 77H
405         JE UP                ;w
406     CMP AL, 57H
407         JE UP                ;W
408
409     CMP AL, 61H
410         JE LEFT             ;a
411     CMP AL, 41H
412         JE LEFT             ;A
413
414     CMP AL, 73H
415         JE DOWN             ;s
416     CMP AL, 53H
417         JE DOWN             ;S
418
419     CMP AL, 64H
420         JE RIGHT            ;d
421     CMP AL, 44H
422         JE RIGHT            ;D
423
424     CMP AL, 1Bh
425         JE TERMINATE
426     CMP AL, 0Dh
427         JE FOURTHPAGE
428     RET
429     ;CHECK POSITION BEFORE MOVING
```

```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
430 UP:
431     MOV AX, PLAYER_Y
432     CMP AX, 31
433     JLE CANNOTMOVE
434     DISPIImage PLAYER_X, PLAYER_Y, JET1, 20
435     SUB PLAYER_Y, STEPLENGTH
436     PImage PLAYER_X, PLAYER_Y, JET1, 20
437     COLLISION_TEST SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
438     RET
439 LEFT:
440     MOV AX, PLAYER_X
441     CMP AX, 11
442     JLE CANNOTMOVE
443     DISPIImage PLAYER_X, PLAYER_Y, JET1, 20
444     SUB PLAYER_X, STEPLENGTH
445     PImage PLAYER_X, PLAYER_Y, JET1, 20
446     COLLISION_TEST SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
447     RET
448 DOWN:
449     MOV AX, PLAYER_Y
450     CMP AX, 180
451     JGE CANNOTMOVE
452     DISPIImage PLAYER_X, PLAYER_Y, JET1, 20
453     ADD PLAYER_Y, STEPLENGTH
454     PImage PLAYER_X, PLAYER_Y, JET1, 20
455     COLLISION_TEST SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
456     RET
457 RIGHT:
458     MOV AX, PLAYER_X
459     CMP AX, 280
460     JGE CANNOTMOVE
461     DISPIImage PLAYER_X, PLAYER_Y, JET1, 20
462     ADD PLAYER_X, STEPLENGTH
463     PImage PLAYER_X, PLAYER_Y, JET1, 20
464     COLLISION_TEST SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
465     RET
466 CANNOTMOVE:
467     RET
468 KEY_ACTIONS ENDP
Assembly language sour length : 31,507 lines : 806 Ln : 402 Col : 34 Pos : 24,079 Windows (CR UTF-8 | INS
```

接收玩家輸入後，先辨別按鍵為何，再決定動作。移動前會確認玩家是否能繼續於遊戲畫面內移動而不超出範圍，若成功移動完則會再進行一次碰撞判定。

#### iv. UPDATESCORE 更新積分

```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
471 UPDATESCORE PROC USES AX BX CX DX ;PRINT HEX. NUMBERS
472 LOCAL n1: word, n2: word, temp: byte
473     mov cl, 16
474     mov eax, SCORE_P
475     shr eax, cl
476     mov n1, ax
477     mov eax, SCORE_P
478     shl eax, cl
479     shr eax, cl
480     mov n2, ax
481
482     mov cl, 4
483 P1:
484     mov ax, n1
485     shr ah, cl
486     cmp ah, 10
487     jae PHex1
488     add ah, 30h
489     mov temp, ah
490     PrintChar temp
491     jmp P2
492 PHex1:
493     add ah, 55 ;10+55=65 (0Ah+37h=41h)
494     mov temp, ah
495     PrintChar temp
496
497 P2:
```

Assembly language sourl length : 31,507 lines : 806 Ln : 471 Col : 56 Pos : 25,703 Windows (CR UTF-8) | INS

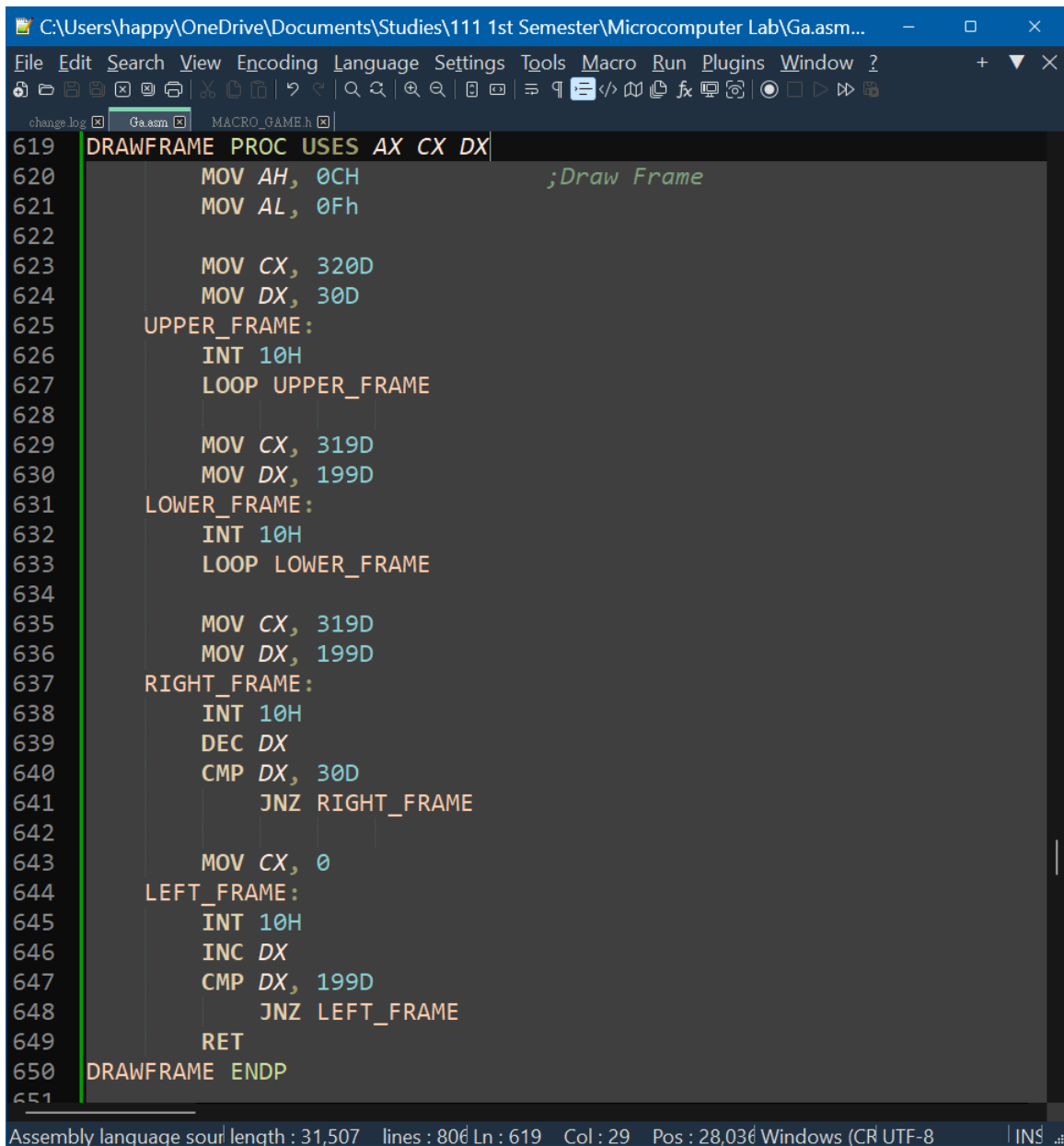
此副程式用於印出遊戲積分，十六進制數字共8位數，因程式碼重複性高，於此僅呈現一部分。

## v. INITIALIZE\_GAME 遊戲介面初始化

```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
595 INITIALIZE_GAME PROC
596     PImage PLAYER_X, PLAYER_Y, JET1, 20
597     PRINTSTRING GAMEMODE, 12, 1, 10, LIGHT_CYAN
598     CMP MODE, 1
599     JE PE
600     CMP MODE, 2
601     JE PN
602     CMP MODE, 3
603     JE PH
604     PE:
605         PRINTSTRING EASYM, 23, 1, 4, LIGHT_CYAN
606         JMP FINISH
607     PN:
608         PRINTSTRING NORMM, 23, 1, 6, LIGHT_CYAN
609         JMP FINISH
610     PH:
611         PRINTSTRING HARDM, 23, 1, 4, LIGHT_CYAN
612     FINISH:
613         PRINTSTRING SCORE, 31, 1, 5, YELLOW
614         PRINTSTRING HEALTH, 1, 1, 6, LIGHT_PURPLE
615         PRINTSTRING LEVEL, 15, 2, 7, LIGHT_BLUE
616         RET
617 INITIALIZE_GAME ENDP
618
```

將遊戲畫面上方的狀態列印出

## vi. DRAWFRAME 畫遊戲的白框



```
619 DRAWFRAME PROC USES AX CX DX
620     MOV AH, 0CH                ;Draw Frame
621     MOV AL, 0Fh
622
623     MOV CX, 320D
624     MOV DX, 30D
625     UPPER_FRAME:
626     INT 10H
627     LOOP UPPER_FRAME
628
629     MOV CX, 319D
630     MOV DX, 199D
631     LOWER_FRAME:
632     INT 10H
633     LOOP LOWER_FRAME
634
635     MOV CX, 319D
636     MOV DX, 199D
637     RIGHT_FRAME:
638     INT 10H
639     DEC DX
640     CMP DX, 30D
641     JNZ RIGHT_FRAME
642
643     MOV CX, 0
644     LEFT_FRAME:
645     INT 10H
646     INC DX
647     CMP DX, 199D
648     JNZ LEFT_FRAME
649     RET
650 DRAWFRAME ENDP
651
```

Assembly language sourl length : 31,507 lines : 806 Ln : 619 Col : 29 Pos : 28,036 Windows (CR UTF-8) | INS

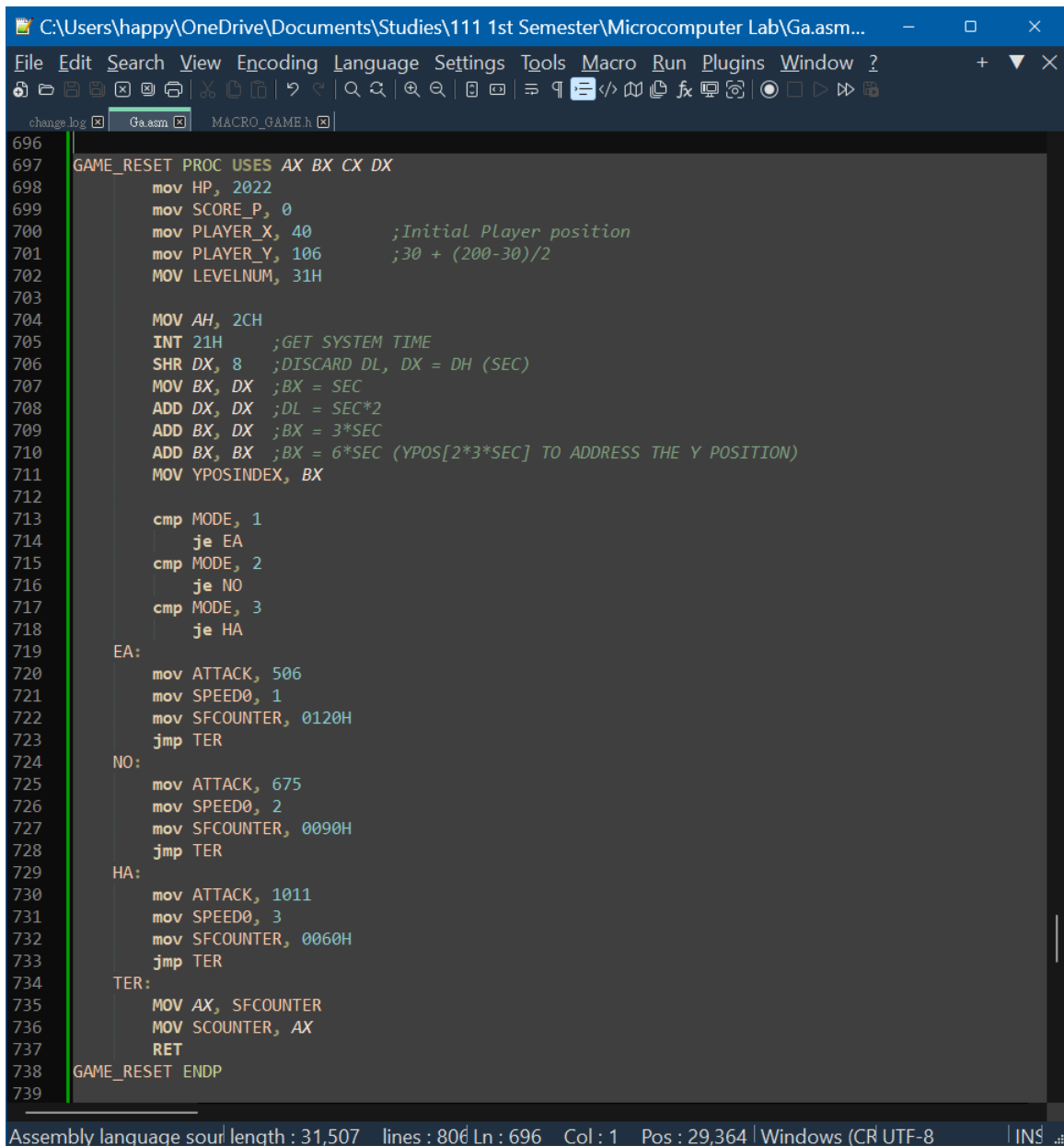
此白框將狀態列和玩家與障礙物的移動範圍區隔開

## vii. ADDSCORE 加分

```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change log Ga.asm MACRO_GAME.h
652 ADDSCORE PROC USES EAX EBX ;according to player position and game mode
653     CMP PLAYER_X, 115
654     JB NOBONUS
655     CMP MODE, 1
656     JE add2
657     CMP MODE, 2
658     JE add4
659     CMP MODE, 3
660     JE add6
661 add2:
662     ADD SCORE_P, 2
663     PRINTSTRING MORES, 8, 2, 5, LIGHT_GREEN
664     JMP CONT
665 add4:
666     ADD SCORE_P, 4
667     PRINTSTRING MORES, 8, 2, 5, LIGHT_GREEN
668     JMP CONT
669 add6:
670     ADD SCORE_P, 6
671     PRINTSTRING MORES, 8, 2, 5, LIGHT_GREEN
672     JMP CONT
673 NOBONUS:
674     PRINTSTRING BLANKS, 8, 2, 5, BLACK
675 CONT:
676     mov eax, 0
677     mov al, mode
678     cmp al, 1
679     je EASY
680     cmp al, 2
681     je NORMAL
682     cmp al, 3
683     je HARD
684 EASY:
685     add SCORE_P, eax
686     JMP FINISH
687 NORMAL:
688     add SCORE_P, eax
689     JMP FINISH
690 HARD:
691     add SCORE_P, eax
692     JMP FINISH
693 FINISH:
694     RET
695 ADDSCORE ENDP
Assembly language sour length : 31,507 lines : 806 Ln : 652 Col : 84 Pos : 28,678 Windows (CR UTF-8 | INS
```

按照遊戲難度累加積分與判定玩家位置並給予獎勵分數

## viii. GAME\_RESET 設定遊戲參數



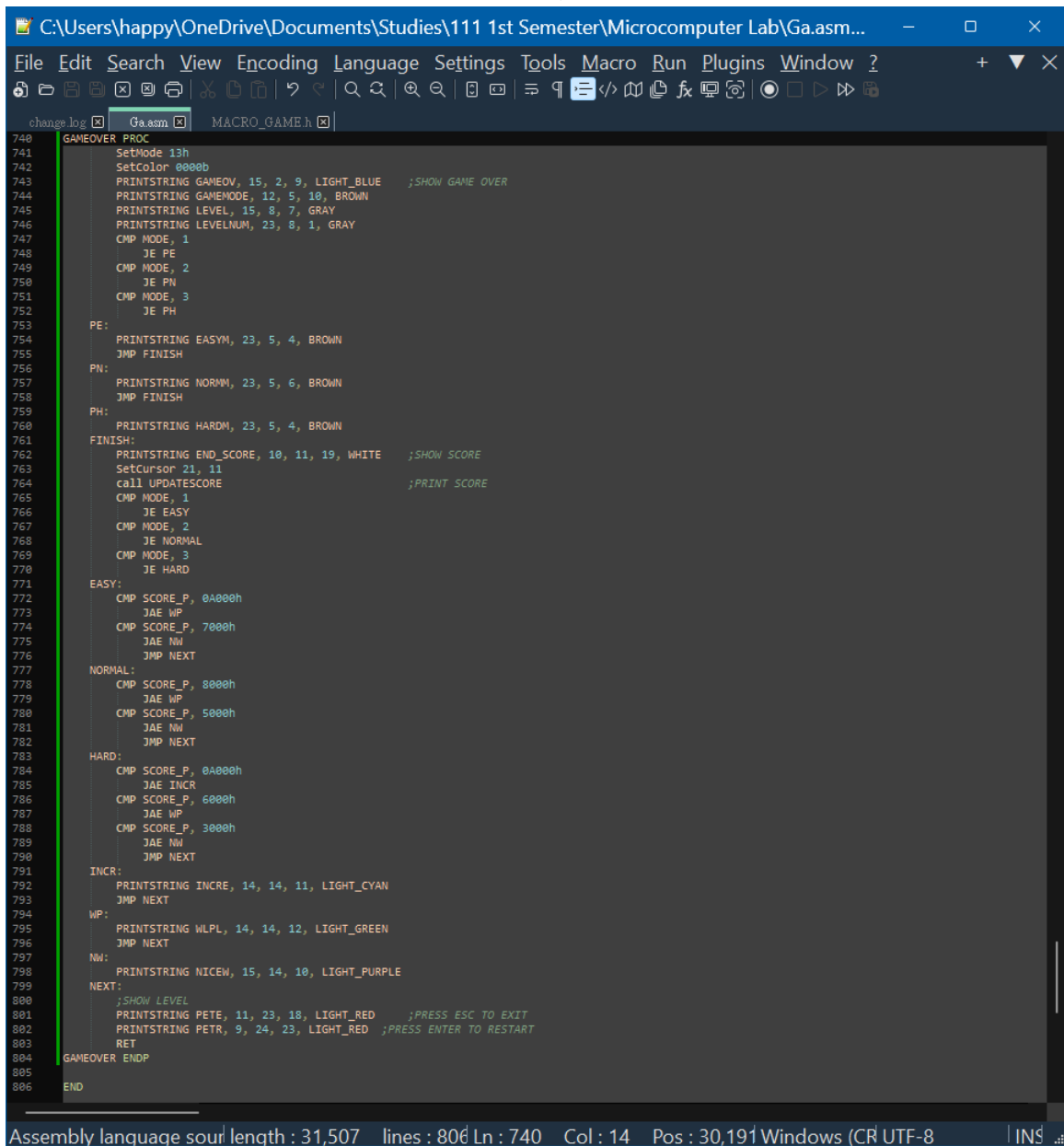
```
696
697 GAME_RESET PROC USES AX BX CX DX
698     mov HP, 2022
699     mov SCORE_P, 0
700     mov PLAYER_X, 40      ;Initial Player position
701     mov PLAYER_Y, 106     ;30 + (200-30)/2
702     MOV LEVELNUM, 31H
703
704     MOV AH, 2CH
705     INT 21H      ;GET SYSTEM TIME
706     SHR DX, 8    ;DISCARD DL, DX = DH (SEC)
707     MOV BX, DX   ;BX = SEC
708     ADD DX, DX   ;DL = SEC*2
709     ADD BX, DX   ;BX = 3*SEC
710     ADD BX, BX   ;BX = 6*SEC (YPOS[2*3*SEC] TO ADDRESS THE Y POSITION)
711     MOV YPOSINDEX, BX
712
713     cmp MODE, 1
714     je EA
715     cmp MODE, 2
716     je NO
717     cmp MODE, 3
718     je HA
719
720 EA:
721     mov ATTACK, 506
722     mov SPEED0, 1
723     mov SFCOUNTER, 0120H
724     jmp TER
725
726 NO:
727     mov ATTACK, 675
728     mov SPEED0, 2
729     mov SFCOUNTER, 0090H
730     jmp TER
731
732 HA:
733     mov ATTACK, 1011
734     mov SPEED0, 3
735     mov SFCOUNTER, 0060H
736     jmp TER
737
738 TER:
739     MOV AX, SFCOUNTER
740     MOV SCOUNTER, AX
741     RET
742 GAME_RESET ENDP
743
```

Assembly language sour length : 31,507 lines : 806 Ln : 696 Col : 1 Pos : 29,364 | Windows (CR UTF-8) | INS

遊戲開始前將玩家位置、等級、分數、生命值等回歸初始設定，並調整遊戲難度。我們取系統時間（秒數），將其六倍（可為0~354中的一些數字）做為YPOSINDEX的值，即可使400bytes的YPOS中的某些數字作為障礙物的初始（遊戲開始後的第一個）Y值，以此便可做出隨機的效果。



## ix. GAMEOVER 遊戲結束



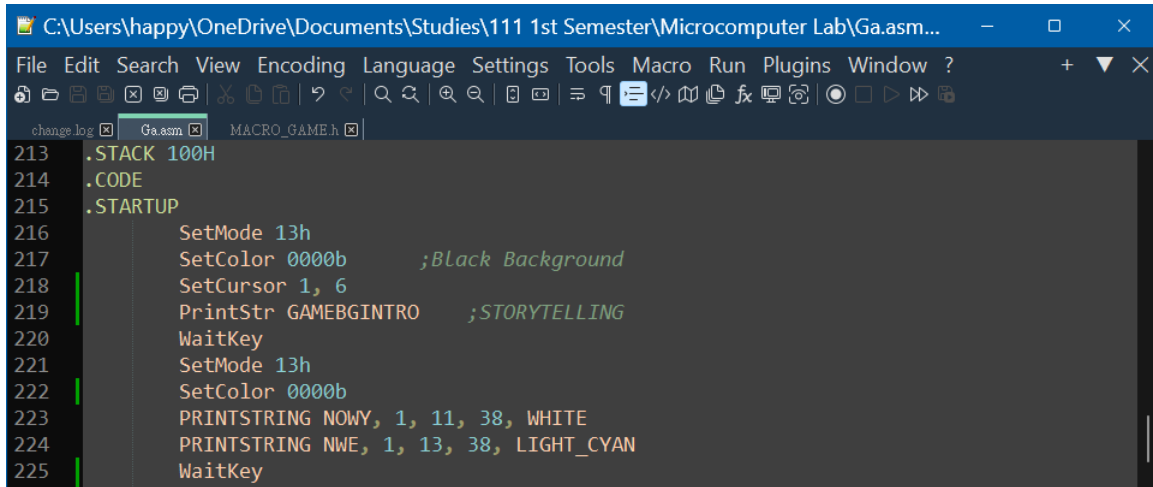
```
740 GAMEOVER PROC
741     SetMode 13h
742     SetColor 0000b
743     PRINTSTRING GAMEOV, 15, 2, 9, LIGHT_BLUE ;SHOW GAME OVER
744     PRINTSTRING GAMEMODE, 12, 5, 10, BROWN
745     PRINTSTRING LEVEL, 15, 8, 7, GRAY
746     PRINTSTRING LEVELNUM, 23, 8, 1, GRAY
747     CMP MODE, 1
748     JE PE
749     CMP MODE, 2
750     JE PN
751     CMP MODE, 3
752     JE PH
753
754     PE: PRINTSTRING EASYM, 23, 5, 4, BROWN
755         JMP FINISH
756
757     PN: PRINTSTRING NORMM, 23, 5, 6, BROWN
758         JMP FINISH
759
760     PH: PRINTSTRING HARDM, 23, 5, 4, BROWN
761
762     FINISH: PRINTSTRING END_SCORE, 10, 11, 19, WHITE ;SHOW SCORE
763             SetCursor 21, 11
764             call UPDATESCORE ;PRINT SCORE
765             CMP MODE, 1
766             JE EASY
767             CMP MODE, 2
768             JE NORMAL
769             CMP MODE, 3
770             JE HARD
771
772     EASY: CMP SCORE_P, 0A000h
773           JAE WP
774           CMP SCORE_P, 7000h
775           JAE NW
776           JMP NEXT
777
778     NORMAL: CMP SCORE_P, 8000h
779             JAE WP
780             CMP SCORE_P, 5000h
781             JAE NW
782             JMP NEXT
783
784     HARD: CMP SCORE_P, 0A000h
785           JAE INCR
786           CMP SCORE_P, 6000h
787           JAE WP
788           CMP SCORE_P, 3000h
789           JAE NW
790           JMP NEXT
791
792     INCR: PRINTSTRING INCR, 14, 14, 11, LIGHT_CYAN
793           JMP NEXT
794
795     WP: PRINTSTRING WLPL, 14, 14, 12, LIGHT_GREEN
796         JMP NEXT
797
798     NW: PRINTSTRING NICEW, 15, 14, 10, LIGHT_PURPLE
799
800     NEXT: ;SHOW LEVEL
801           PRINTSTRING PETE, 11, 23, 18, LIGHT_RED ;PRESS ESC TO EXIT
802           PRINTSTRING PETR, 9, 24, 23, LIGHT_RED ;PRESS ENTER TO RESTART
803           RET
804 GAMEOVER ENDP
805
806 END
```

Assembly language source length : 31,507 lines : 806 Ln : 740 Col : 14 Pos : 30,191 Windows (CR UTF-8) | INS .it

印出相關字串（如實習結果所示），並在每個遊戲難度下判斷該玩家的成績高低給予相對應的鼓勵話語

- 主程式：

- i. 第零頁——故事

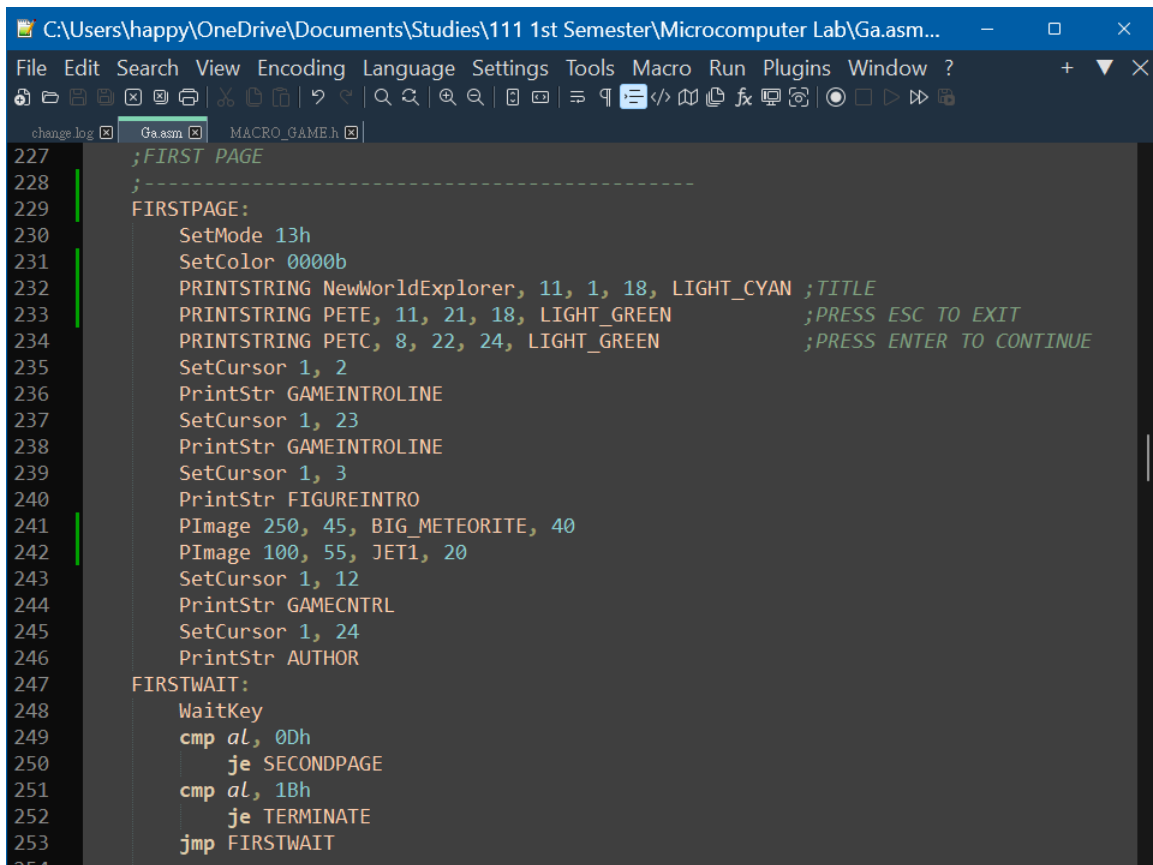


The screenshot shows a GAUSS IDE window with the title bar "C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...". The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations, editing, and execution. The editor shows three tabs: "change.log", "Ga.asm", and "MACRO\_GAME.h". The code in "Ga.asm" is as follows:

```
213 .STACK 100H
214 .CODE
215 .STARTUP
216     SetMode 13h
217     SetColor 0000b      ;Black Background
218     SetCursor 1, 6
219     PrintStr GAMEBGINTRO ;STORYTELLING
220     WaitKey
221     SetMode 13h
222     SetColor 0000b
223     PRINTSTRING NOWY, 1, 11, 38, WHITE
224     PRINTSTRING NWE, 1, 13, 38, LIGHT_CYAN
225     WaitKey
```

簡單的輸出字串，並讓使用者按下任意鍵繼續

## ii. 第一頁——介紹

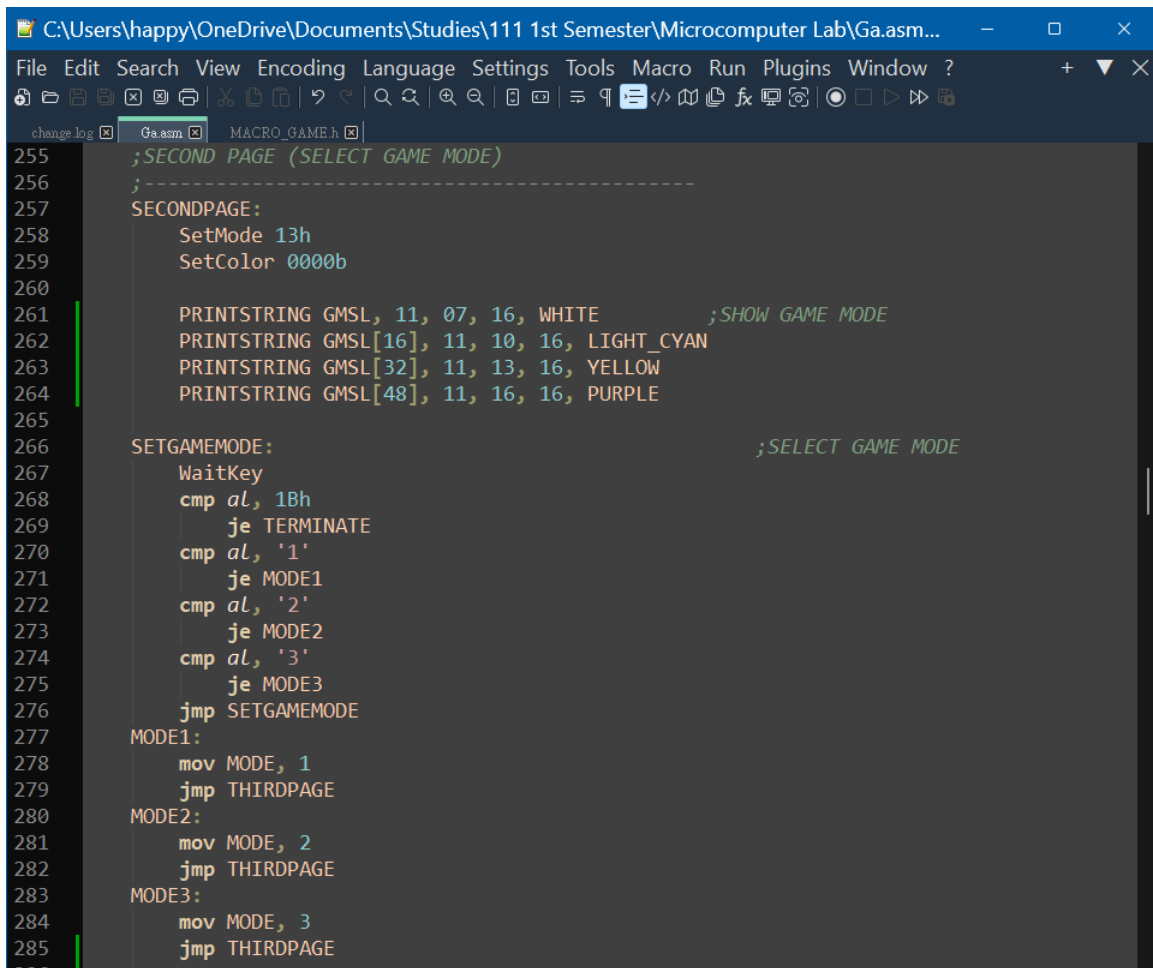


The screenshot shows a code editor window titled "C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...". The editor contains assembly code for a game introduction. The code is as follows:

```
227 ;FIRST PAGE
228 ;-----
229 FIRSTPAGE:
230     SetMode 13h
231     SetColor 0000b
232     PRINTSTRING NewWorldExplorer, 11, 1, 18, LIGHT_CYAN ;TITLE
233     PRINTSTRING PETE, 11, 21, 18, LIGHT_GREEN ;PRESS ESC TO EXIT
234     PRINTSTRING PETC, 8, 22, 24, LIGHT_GREEN ;PRESS ENTER TO CONTINUE
235     SetCursor 1, 2
236     PrintStr GAMEINTROLINE
237     SetCursor 1, 23
238     PrintStr GAMEINTROLINE
239     SetCursor 1, 3
240     PrintStr FIGUREINTRO
241     PImage 250, 45, BIG_METEORITE, 40
242     PImage 100, 55, JET1, 20
243     SetCursor 1, 12
244     PrintStr GAMECNTRL
245     SetCursor 1, 24
246     PrintStr AUTHOR
247 FIRSTWAIT:
248     WaitKey
249     cmp al, 0Dh
250     je SECONDPAGE
251     cmp al, 1Bh
252     je TERMINATE
253     jmp FIRSTWAIT
254
```

這邊給玩家簡單的遊戲控制與圖案介紹，在此畫面中僅能按下 ENTER繼續或ESC退出

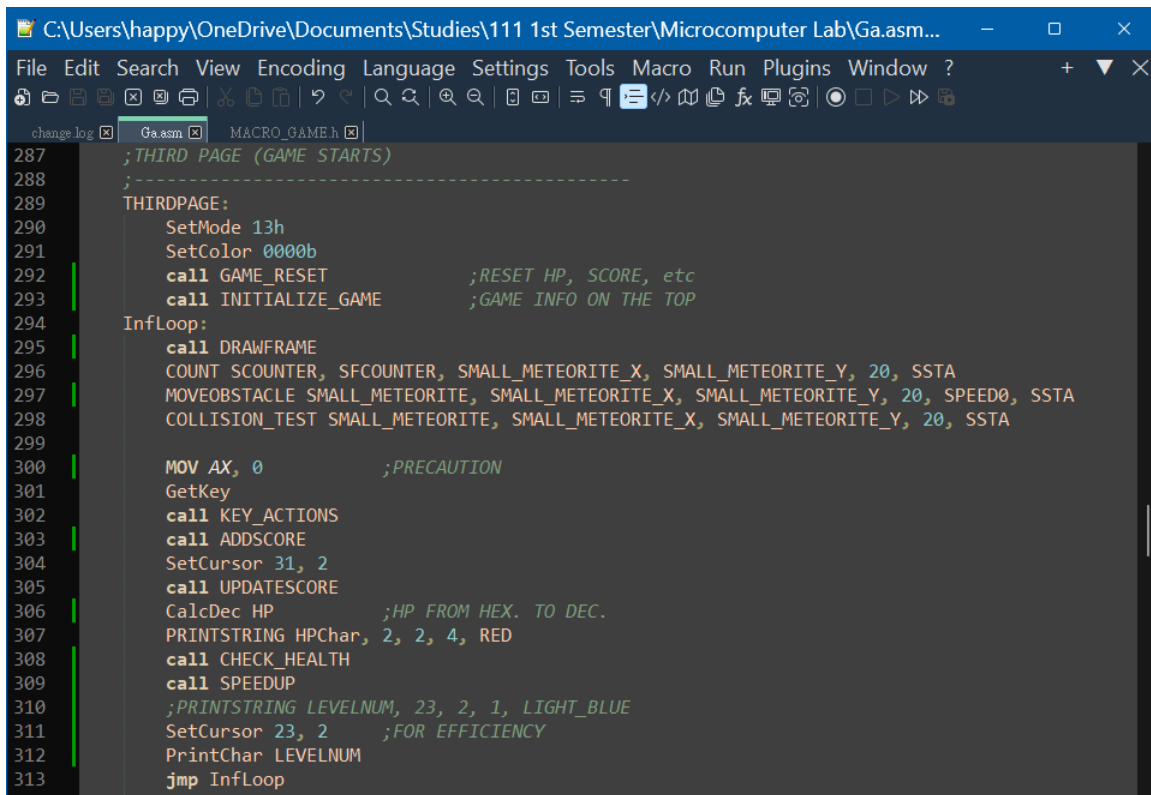
### iii. 第二頁——難度選擇



```
255 ;SECOND PAGE (SELECT GAME MODE)
256 ;-----
257 SECONDPAGE:
258     SetMode 13h
259     SetColor 0000b
260
261     PRINTSTRING GMSL, 11, 07, 16, WHITE ;SHOW GAME MODE
262     PRINTSTRING GMSL[16], 11, 10, 16, LIGHT_CYAN
263     PRINTSTRING GMSL[32], 11, 13, 16, YELLOW
264     PRINTSTRING GMSL[48], 11, 16, 16, PURPLE
265
266     SETGAMEMODE: ;SELECT GAME MODE
267     WaitKey
268     cmp al, 1Bh
269     je TERMINATE
270     cmp al, '1'
271     je MODE1
272     cmp al, '2'
273     je MODE2
274     cmp al, '3'
275     je MODE3
276     jmp SETGAMEMODE
277 MODE1:
278     mov MODE, 1
279     jmp THIRDPAGE
280 MODE2:
281     mov MODE, 2
282     jmp THIRDPAGE
283 MODE3:
284     mov MODE, 3
285     jmp THIRDPAGE
```

讓使用者輸入欲遊玩之難度或按下ESC退出

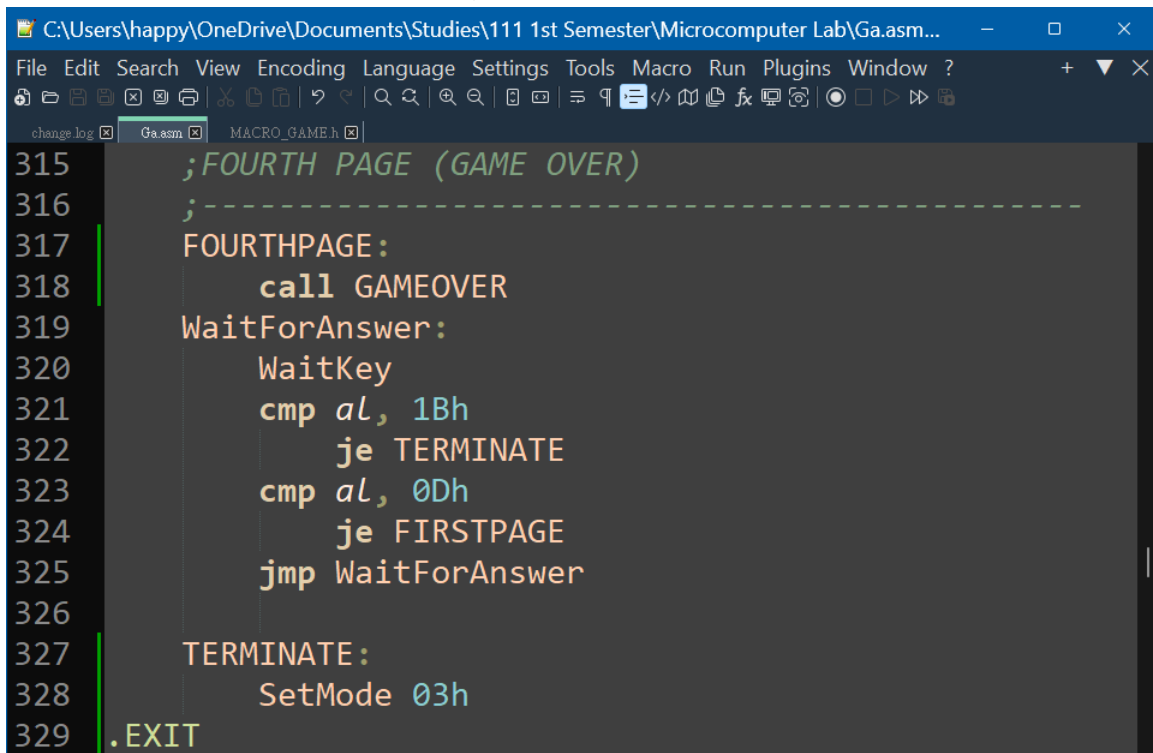
#### iv. 第三頁——遊戲主迴圈



```
287 ;THIRD PAGE (GAME STARTS)
288 ;-----
289 THIRDPAGE:
290     SetMode 13h
291     SetColor 0000b
292     call GAME_RESET           ;RESET HP, SCORE, etc
293     call INITIALIZE_GAME      ;GAME INFO ON THE TOP
294 InfLoop:
295     call DRAWFRAME
296     COUNT SCOUNTER, SFCOUNTER, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
297     MOVEOBSTACLE SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SPEED0, SSTA
298     COLLISION_TEST SMALL_METEORITE, SMALL_METEORITE_X, SMALL_METEORITE_Y, 20, SSTA
299
300     MOV AX, 0                 ;PRECAUTION
301     GetKey
302     call KEY_ACTIONS
303     call ADDSCORE
304     SetCursor 31, 2
305     call UPDATESCORE
306     CalcDec HP                ;HP FROM HEX. TO DEC.
307     PRINTSTRING HPchar, 2, 2, 4, RED
308     call CHECK_HEALTH
309     call SPEEDUP
310     ;PRINTSTRING LEVELNUM, 23, 2, 1, LIGHT_BLUE
311     SetCursor 23, 2           ;FOR EFFICIENCY
312     PrintChar LEVELNUM
313     jmp InfLoop
```

1. 遊戲實際上為一個無窮迴圈，只有當玩家按下ENTER鍵或ESC鍵或生命值小於等於零時才會執行之後的程式碼(第四頁或程式結束)，所以在其他狀況下，遊戲是一直進行著的。
2. 首先我們將遊戲中各參數設置好並畫好遊戲介面，之後才進入迴圈。與玩家移動相同，每當障礙物移動完成後，須檢查一次是否有碰撞到的情況產生，接著開放使用者輸入，程式會依照輸入做出相對應的動作。每次迴圈都會檢查目前的生命值與積分，進行結束遊戲或是加速的動作。
3. 迴圈內的指令雖多，看起來也有先後之分，但實際上處理器運行十分迅速，因此可以將迴圈內程式碼視為同一時間點執行，各副程式或巨集在排序上並沒有太多限制，唯獨移動與碰撞判斷須留意。

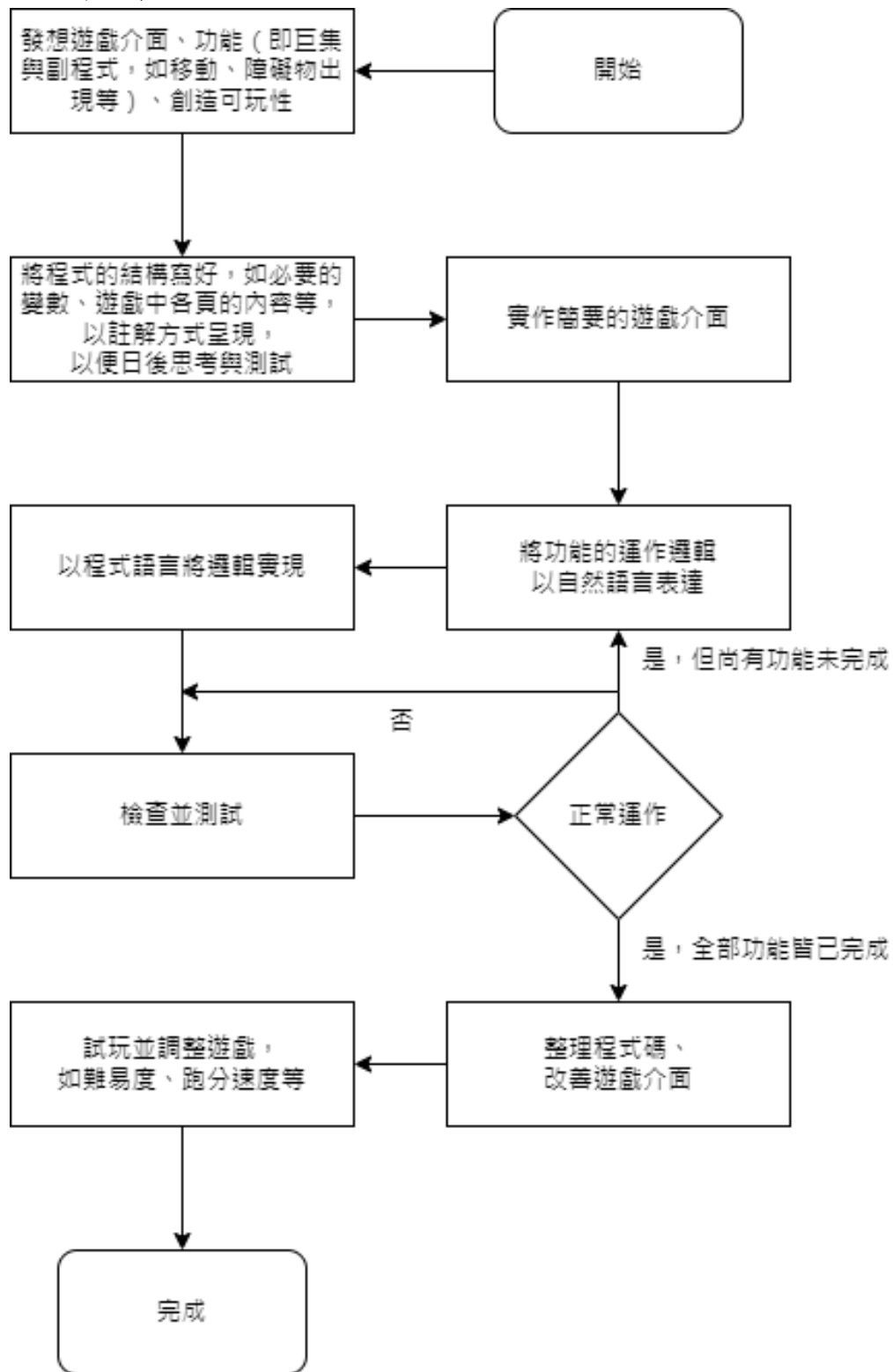
## v. 第四頁——結束



```
C:\Users\happy\OneDrive\Documents\Studies\111 1st Semester\Microcomputer Lab\Ga.asm...
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
change.log Ga.asm MACRO_GAME.h
315 ;FOURTH PAGE (GAME OVER)
316 ;-----
317 FOURTHPAGE:
318     call GAMEOVER
319 WaitForAnswer:
320     WaitKey
321     cmp al, 1Bh
322     je TERMINATE
323     cmp al, 0Dh
324     je FIRSTPAGE
325     jmp WaitForAnswer
326
327 TERMINATE:
328     SetMode 03h
329 .EXIT
```

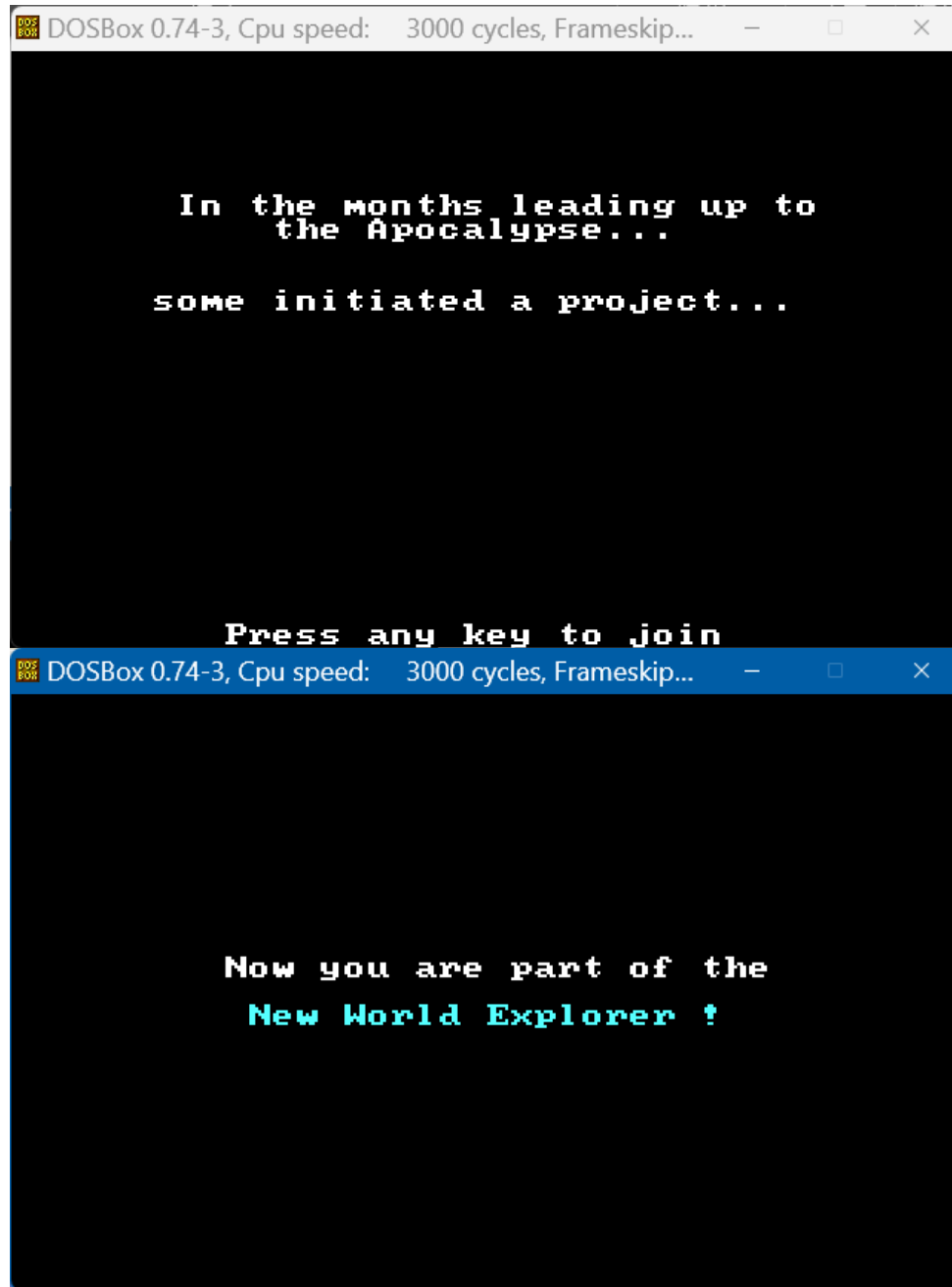
遊戲結束時會有獨立的一個畫面做總結，並提示玩家能夠重新開始或是結束遊戲

## 二、 流程圖



實習結果 (遊戲運作畫面)

I. 遊戲背景：



在末日來臨前幾個月……

有一些人發起了一個計畫……

現在你是新世界探索者的一員了！



## II. 首頁：



W、A、S、D分別可以控制遊戲角色向上，後，下，前

### III. 遊戲難度選擇畫面：



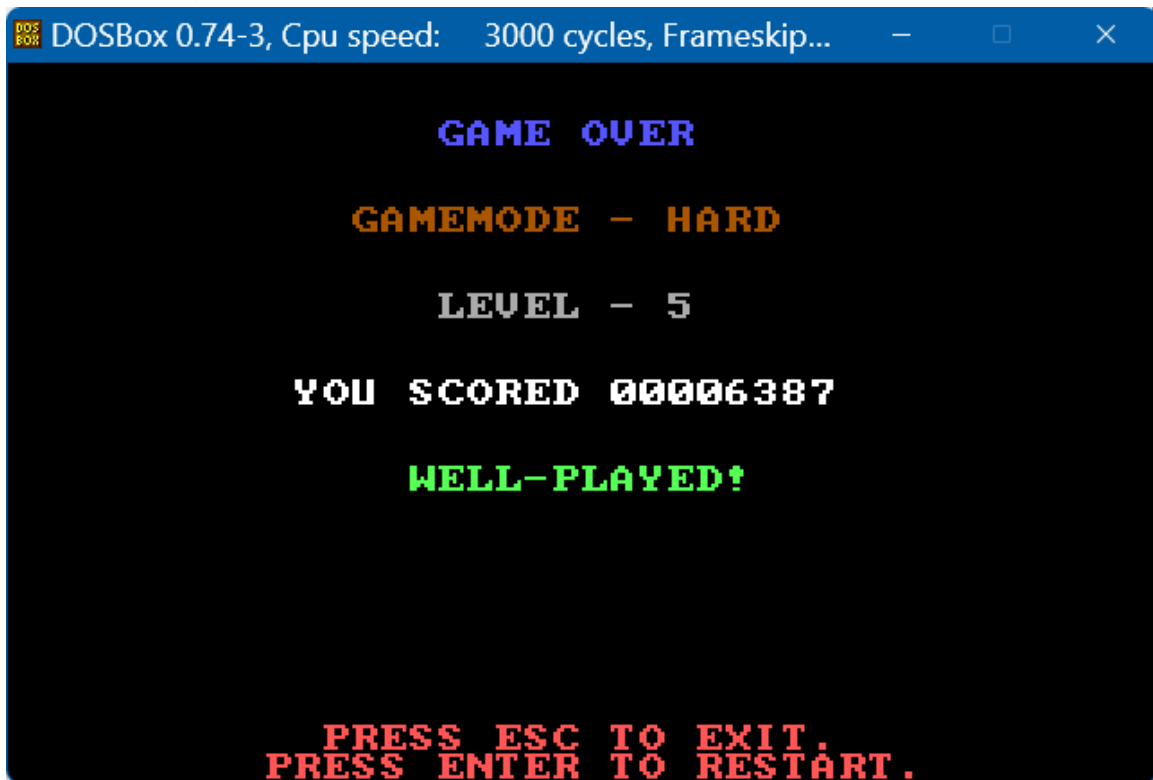
按下1、2、3，分別選擇簡單、普通、困難，於此畫面按下ESC則結束遊戲程式

#### IV. 遊戲畫面：



1. 左上角顯示生命值，設計成2022為紀念用途。依照遊戲難度簡易、普通與困難分別給予四次、三次與二次機會，機會用完時遊戲結束。
2. 當玩家移至圖中所示位置或更右側時，生命值旁會出現BONUS字樣，代表玩家很勇，依據遊戲難度給予更多分數
3. 中間則有玩家選擇之遊戲難度與等級，等級隨遊玩時間增加而增加，每增一級，速度就增加一些，簡單與普通模式最高3級，困難模式最高5級
4. 右上角則是分數，遊戲開始後將不斷增加，並不依據等級增加而改變增量，僅按照遊戲難度與有BONUS與否而改變增量
5. 於此畫面按下ESC則結束遊戲程式，ENTER則結束遊戲進入下一頁

## V. 結束畫面：



1. 生命值小於等於零時，遊戲會進入第四頁，顯示「遊戲結束」，並印出遊戲難度、等級、分數與鼓勵話語
2. 鼓勵話語依照分數與關卡決定給予“NICE WORK!”或“WELL-PLAYED!”或“INCREDIBLE!”，分數不夠高則不顯示。
3. 可按下ESC退出或按下ENTER重新開始遊戲

三、心得（同一組兩位組員要分開寫，附在同一份報告裡）

● 陳○○：

（隱私保護）

● 李○○:

(隱私保護)