



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

### 期末報告

班 級       ：四電二甲  
學 生       ：楊修旻、黃鈺善  
學 號       ：B10707009、B10707049  
建檔日期   ：2020/1/6

## I. 學習成果：

此程式為使用x86組合語言所編寫的遊戲，為一PvP戰車對戰遊戲。遊戲流程為：先客製化自己的戰車，選擇難易度，選擇地圖後開始遊戲。玩家一使用W、S、A、D與空白鍵控制，玩家二使用方向鍵上、下、左、右及Enter鍵控制，擊中對方戰車者獲勝。

下面將開始說明關於技術層面的特色：

1. 使用SVGA，800\*600 256色顯示。
2. 直接存取影像記憶體，繞開int 10h加速繪圖過程。
3. 取代鍵盤中斷向量9h，達到同時讀取多個按鍵的目的。
4. 音效IO，增加遊戲樂趣。
5. 遊戲紀錄保存，保存玩家精心配色的戰車。

程式分為2個asm檔、3個h檔與一個inc檔。Game.asm為主程式，也是程式的進入點。pj5.asm為負責畫戰車與音效輸出的程式，透過pj5.inc來宣告模型，從主程式用invoke呼叫。GameDraw.h負責程式繪圖的部分，有儲存像素至影像記憶體、畫圓形與設定背景顏色功能，關於繪圖部分，後面會詳細說明。GameObject.h為遊戲相關的物件控制都放置於此，如標題繪製、戰車控制、砲彈發射、砲彈與戰車的偵測等。最後，GamePrint.h為字串輸出。

為了最佳遊戲體驗，建議使用DOSBOX運行，

網址：<https://www.dosbox.com/>

程式已放置於壓縮檔與github上，

網址：[https://github.com/shinco0327/uProcessor\\_Game](https://github.com/shinco0327/uProcessor_Game)

遊戲示範影片，

網址：<https://youtu.be/PjO2UNw02XE>

## 程式詳細介紹：

### A. SVGA 800\*600 256色顯示

這次專題，我們使用SVGA 800\*600 256色顯示，使用256色的原因在於我們想要直接存取影像記憶體，繞開使用int 10h，加速繪圖過程；而使用256色是因其大小為1 Byte，相比16 bit的存取，256色簡單許多。選擇800\*600這個解析度我們認為最為適中，相比640\*480解析度，800\*600能發揮的空間更多。

設定SVGA影像模式，首先要使用ax = 4f02h，bx = 影像模式（800\*600 256色為103h），int 10h來設定影像模式。

Mode = 103h的影像記憶體位於A000:0000的位置，但是，在real mode下，只能定址0FFFFH大小的空間，因此必須使用ax = 4f05h int 10h來控制記憶體視窗。我們使用的是速度更快的方法，運用far call透過特定的CS:IP來呼叫此功能。至於CS:IP如何獲得呢？使用ax = 4f01h int 10h，cx = 103h，然後指定一塊記憶體區塊給ES:DI（需要256 Bytes大小）來獲取800\*600 256色模式的相關資訊。執行完後，CS:IP會在[DI+0CH]的地方。

在寫入像素方面，我們撰寫了一個巨集，傳入值為X軸、Y軸及顏色。為了對應相對的記憶體空間，將Y軸傳入值乘上800後加上X軸大小，右邊2 Bytes的大小即為offset；左邊的即為顯示區塊，放入ax暫存器後使用far call呼叫視窗記憶體控制。

在寫入背景部分，我們使用rep stosb，將cx設為0FFFFh，ES:DI為A000:0000H，配合視窗記憶體控制可以快速的寫入背景顏色。

## **B. 設定中斷向量，取代鍵盤中斷向量9H**

起初我們使用int 16h功能來讀取鍵盤輸入，發現在兩個使用者都按下按鍵後，系統只會接收後來按下的一個。這就會造成使用者必須瘋狂點擊鍵盤來使戰車前進，使用者體驗將大幅下降。

因此，我們決定寫一個自己的中斷取代鍵盤中斷向量int 9h。中斷的流程如下：從port 60h取得掃描碼；掃描碼最高位元為0時為按下，掃描碼最高位元為1時表示釋放。透過掃描碼來確認使用者的按鍵為壓下還是釋放，如此一來，就算後一個使用者壓下按鍵，在前一使用者按鍵的釋放掃描碼傳回前，我們就能確認兩個按鍵皆為壓下狀態。

按下  Scan Code: 1Eh

按下  Scan Code: 48h

  均為按下狀態

釋放  Scan Code: 9Eh


 為按下狀態

Figure 1. 鍵盤偵測示意圖

```

MyInterrupt proc
    cli                                ;Disable Interrupt
    pushf
    push    ax
    push    cx
    push    di
    push    es
    mov     ax, @data
    mov     es, ax
    in      al, 60h                    ;Get Scan code
    push    ax
    cld
    and     al, 01111111b
    lea     di, char_table             ;which key?
    mov     cx, LENGTHOF char_table
    repne   scasb
    pop     ax
    je      FindChar                   ;Find in char_table
    jmp     exit_Handler
FindChar:
    sub     di, offset char_table
    dec     di
    bt      ax, 7                      ;Press or release
    jnc     Press
    jc      Release
Press:
    mov     byte ptr char_status[di], 1
    jmp     exit_Handler
Release:
    mov     byte ptr char_status[di], 0
    jmp     exit_Handler
exit_Handler:
    mov     al, 20h                    ;acknowledge the interrupt
    out     20h, al
    pop     es
    pop     di
    pop     cx
    pop     ax
    popf
    sti                                ;Enable the interrupt
    iret
MyInterrupt endp

```

Figure 2. 中斷內容

## C. 音效輸出

為了增加遊戲體驗，我們增加了音效輸出，在遊戲開始或結束時撥放。我們將三個8度的音符所對應的頻率製成陣列。音效的格式為音符加上持續時間所組成。例如，若要播放低音Do為0001，中音Do為0010，高音Do為0100；只要知道簡譜，就能快速的組成音樂。

```
M_Welcome    dw 0ff87h, 0050h, 0010h
              dw 0000h, 0002h
              dw 0050h, 0010h
              dw 0000h, 0002h
              dw 0030h, 0010h
              dw 0000h, 0002h
              dw 0030h, 0010h
              dw 0000h, 0002h
              dw 0020h, 0010h
              dw 0000h, 0002h
              dw 0020h, 0010h
              dw 0000h, 0002h
              dw 0010h, 0080h, 0ffffh
```

Figure 3. 音效範例

在上圖中能看到，0ff87h為音效開頭，0ffffh為音效結尾。每一行均有兩個數值，左邊的即為音符，右邊的則為持續時間。

## D. 紀錄保存

由於玩家可以客製化自己的戰車，我們希望能將戰車的顏色數據保存起來，因此使用了int 21h 檔案控制的功能。在程式開始時從GAMEDATA.txt遊戲存檔中載入資料至程式的.data段。若使用者設定完成戰車顏色，再將資料儲存進GAMEDATA.txt中。

透過GAMEDATA.txt的建立與否，來判斷是不是初次遊玩，若是初次遊玩遊戲，則顯示教程，方便使用者使用。

## E. 子彈動作流程與偵測說明

遊戲分成了4個難易度，分別代表了子彈反射的次數：Easy模式，子彈不反射。Normal模式，子彈碰到牆壁會反彈一次；Hard模式反彈3次及Expert模式反彈5次。

當子彈接近牆壁或障礙物的時候，會判斷牆壁與子彈x軸與y軸的距離，x軸較近或y軸較近決定了不同的反彈方向。偵測子彈是否打到戰車的方法為：將透過子彈中心與戰車中心點的距離來判斷，若小於戰車的半徑則判定打到。

## II. 流程圖

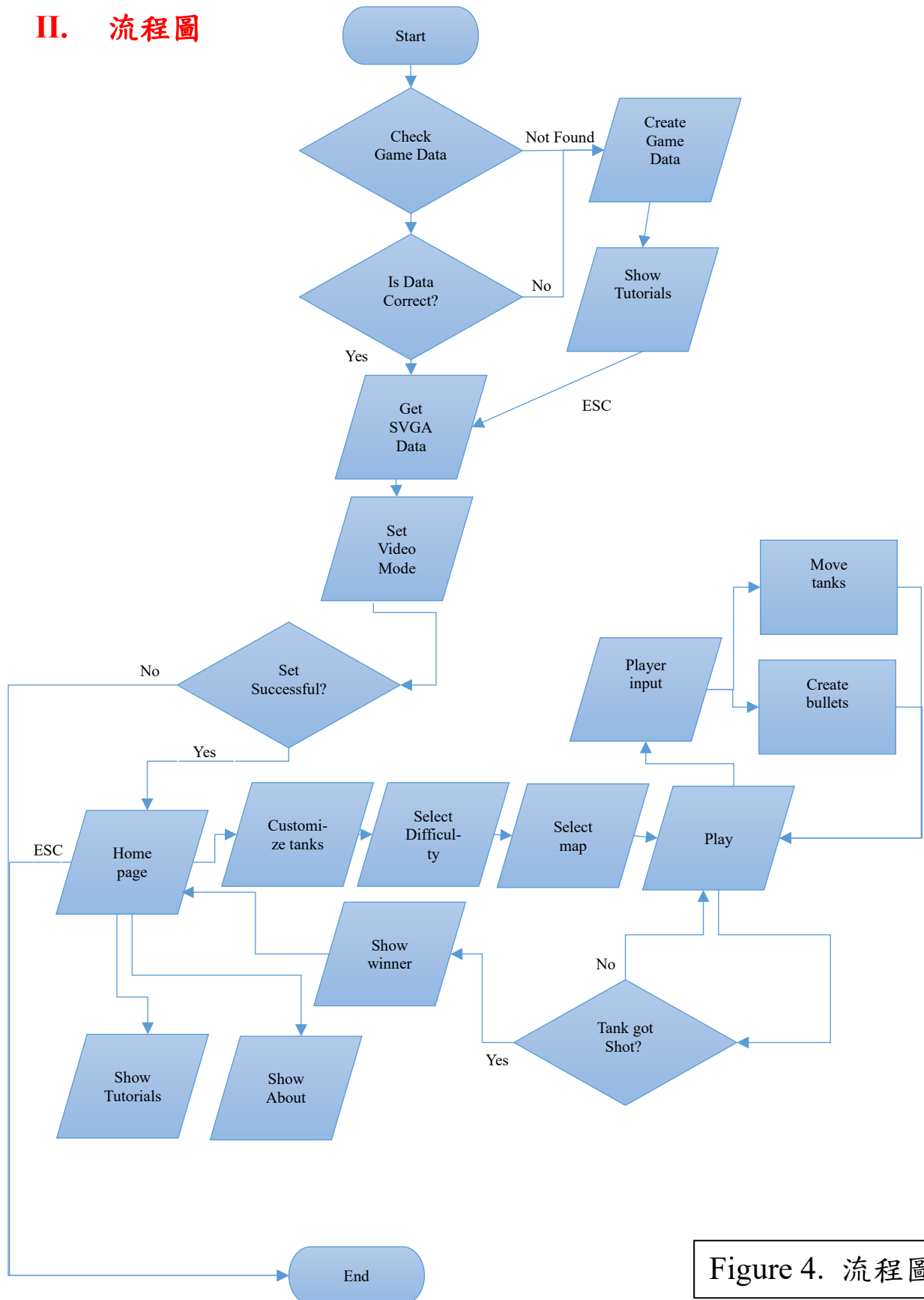


Figure 4. 流程圖

### III. 實習結果

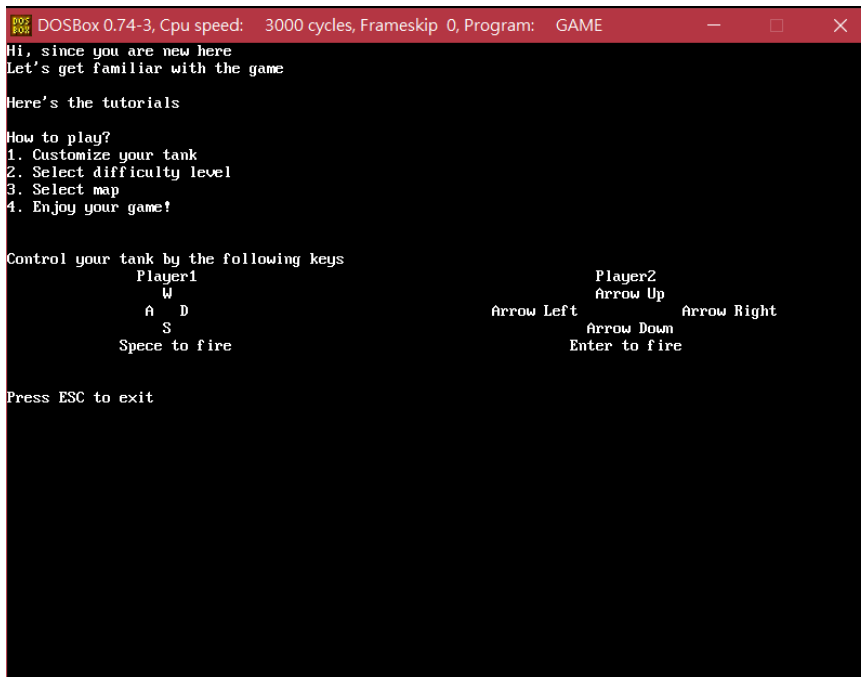


Figure 5.初次使用的教程

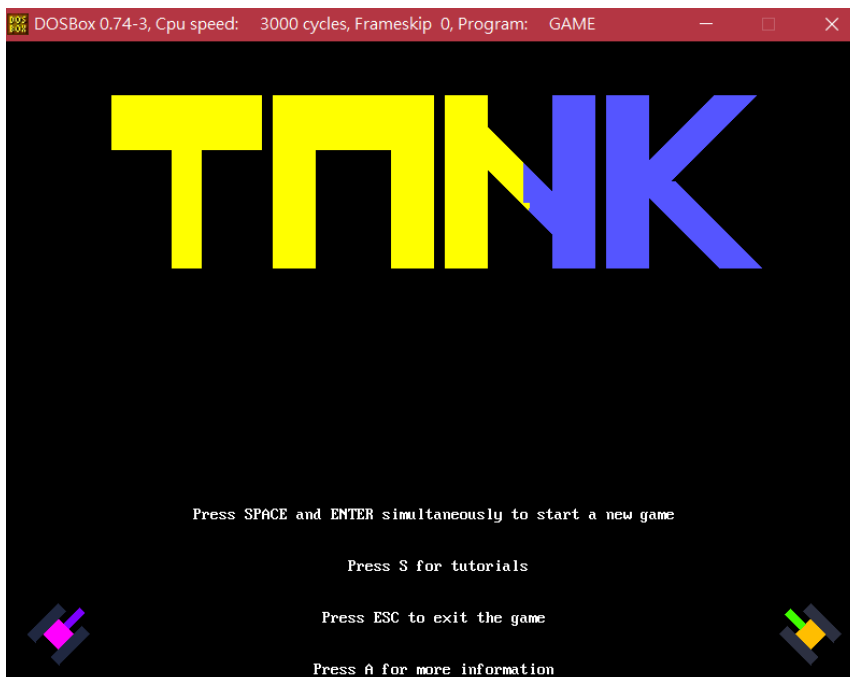


Figure 6.遊戲主畫面



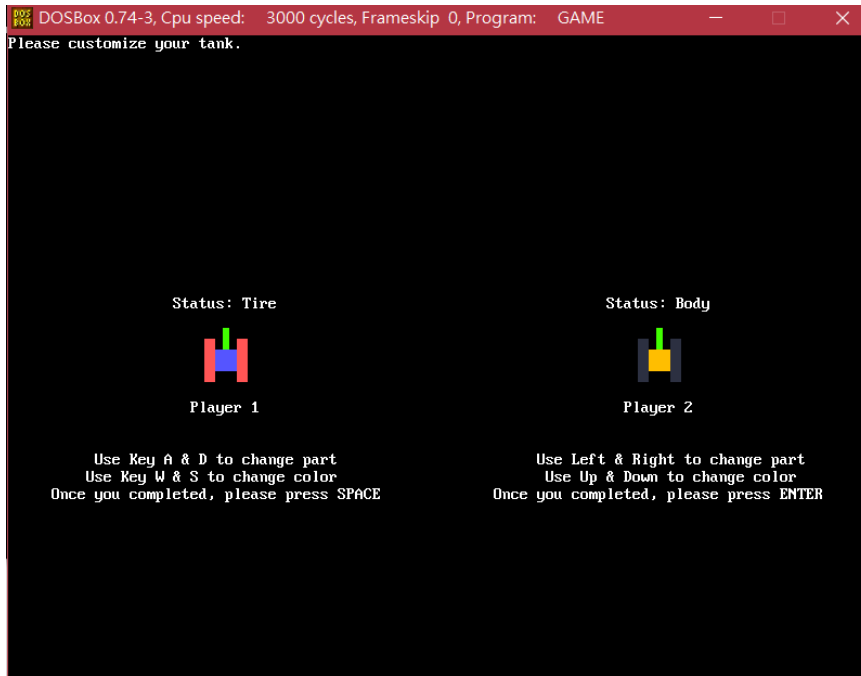


Figure 7. 開始遊戲，首先客製化自己的戰車

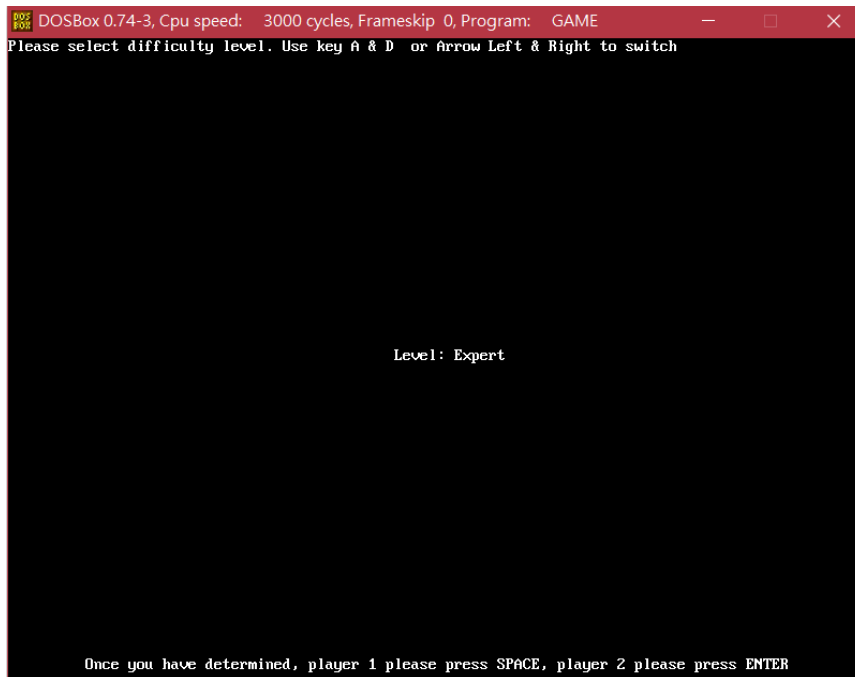


Figure 8. 選擇難易度



Figure 9. 選擇地圖



Figure 10. 遊戲畫面 1



Figure 11. 遊戲畫面 2



Figure 12. 擊中畫面

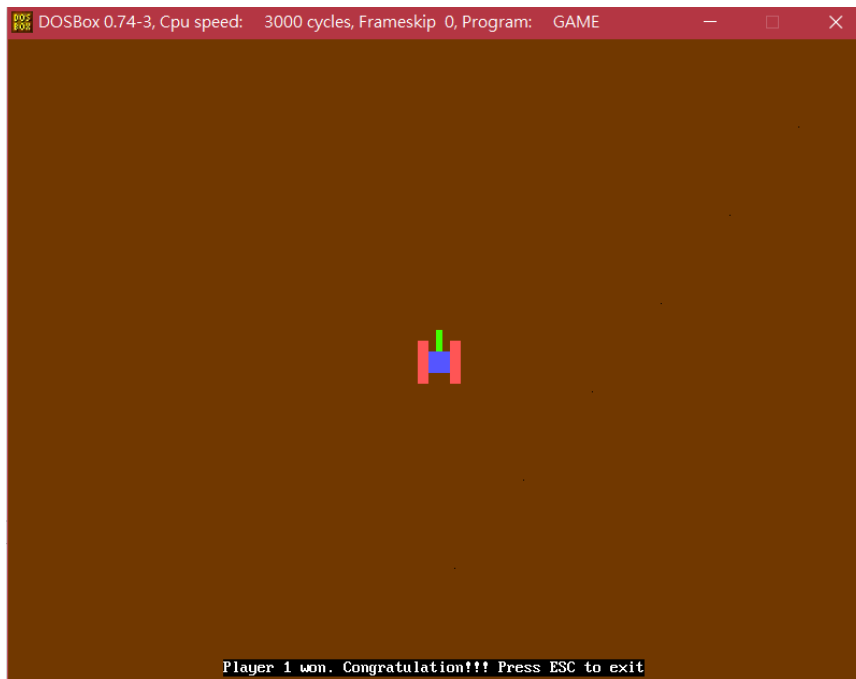


Figure 13. 顯示勝利玩家

## IV. 心得

### ● 組員一 B10707009 楊修旻:

期末專題應用上了課堂上所學大部分的指令，從一開始思考要做什麼遊戲，到一個完整遊戲的產生，是一個非常特別的過程，思考如何應用所學的指令，並參考資料看看別人有什麼其他更好的方法達到所想要的功能，由於戰車遊戲會有同個物件但不同方向的考量，因此最重要的是將每個動作和畫面模組化，而非平常實習課時把東西做出來就好，比起平常的練習最不一樣的是完整的程式設計考量，要作出一個完整的東西並沒辦法用苟且的方式去完成，因為這樣會造成之後更多的困擾和麻煩。

在畫戰車的步驟中一開始的構思沒有很周全，導致程式碼有點冗長，也使用了非常多的Label，雖然最後也是順利畫出了四面八方想要的方向，但還是有很大的進步空間，之後一開始設計程式時應更全面的考量。

我想在這次專題中除了技術上的進步之外，要特別感謝鈺善同學在這次專題中花下的精力，很多較高深的技術都是由他突破的，在向他請教之後每每都會獲得很多進步。

● 組員二 B10707049 黃鈺善:

這次專題可以說是這學期學習微算機概論的集大成，約12月初開始製作，到了元月才製作完成。這次專題為了更好的遊玩體驗，使用了SVGA、取代int 9h、音效iO及檔案輸入輸出等。相比於C或Python，組合語言的資源都比較偏向英文資源，相關網站也比較少；但是，Stack Overflow等論壇也都有許多厲害的高手在。這次偵測多個按鍵按下的方法便是在Stack Overflow上看到的，雖然只提供了文字說明，不過再融合其他資料後，順利地將功能完成了。

本次遇到的最大瓶頸莫過於子彈碰到障礙物的反彈了。由於碰到垂直物體與水平物體的反射方向不一樣，必須分開實作。在碰到直角的地方，子彈容易有非預期的反射發生，這是因為垂直與水平的交界所造成；經過多次的嘗試後，這樣的狀況應較為改善。

從零開始構思遊戲是一個蠻新奇的過程，一開始有遊戲要怎麼玩的想法，但仍不知道要如何實作。透過先將部分模組建立好，再來將完成的模組套在一起。有了一個大概的主體後，便能清楚的知道接下來的流程與改善方向。