

**組 別：第5組**

**題 目：Typing Typing**

**組 員：104590025 余鎧企**

**104590029 黃省喬**

**指導老師：陳偉凱**

目錄

1. 簡介1
2. 動機2
3. 分工2
4. 遊戲介紹4
5. 遊戲說明5
6. 遊戲圖形6
7. 遊戲音效6
8. 程式設計4
9. 程式架構5
10. 程式類別6
11. 程式技術6
12. 結語4
13. 問題與解決辦法5
14. 時間表6
15. 貢獻比例6
16. 檢核表5
17. 收穫6
18. 心得、感想6
19. 對於本課程的建議6

附錄6

1. **簡介**
2. 動機：

剛學習電腦的時候，打字必為入門基礎。尤其在資工系這塊領域，若打字速度跟不上腦袋思考，必定會錯過許多新穎的想法。而我們在這片學海無涯的道路上，開始了我們的根基。自幼時開始學習英打，接觸過許多網絡上的軟件。有文章式、逐行式、單字類，但因為內容太過於單調與呆版，不受我們的喜愛與大眾的需求。在一個偶然的機遇下，我們找到了「ZType」！，儘管同樣是一個英打練習的網頁，與眾不同的是，它以「遊戲」的方式來進行，既有傳統的計算速度與正確率，又有新穎的華麗特效，成功吸引了初學者的目光。為了攻略這個遊戲、我們夜以繼日的練習。而後，我們英打獲得突飛猛進的突破。而當我們修完上學期的”物件導向程式設計”後，便充分的利用所學，想要在這學期實現出這個令我們曾經感動的遊戲。藉由我們的實作，帶給其他想學打字的人一條有效的途徑，並提供給玩家前所未有聽覺及視覺的雙重響宴。

1. **分工**：

余鎧企：

程式碼撰寫：檔案(File)讀寫處理、紀錄(Record)相關保存、音效(Audio)素材蒐集與處理、魔王(Boss)構思。

黃省喬：

程式碼撰寫：主角(Me)設計、怪物(Enemy)相關技術、特效處理、選單(List)頁面撰寫。

美術介面：選單頁面設計、主角皮膚(Skin)、關卡切換特效、遊戲特效設計。

1. **遊戲介紹**
2. 遊戲說明：敵人們正從天而降！趕緊鍵入怪物身上的單字來消滅他們！

玩法：

在本遊戲中有各種不同外貌的敵人(Enemy)，但是他們都有一個共同點，那就是他們的身上都帶著一組英文單字。玩家必須在敵人接近之前，輸入其身上的單字，才能成功的把的敵人消滅。

規則：

當玩家鎖定攻擊了其中一隻怪物，正輸入到一半的時候，必須繼續將其單字輸入完畢，改敵人被消滅之後，才可以再鎖定其他敵人攻擊。

敵人：

本遊戲除了普通的敵人外，另外還有多種強大的Boss，除了字數較長外，還各自擁有不同的技能。

普通敵人(Enemy) **：**

普通的敵人，會以正常速度從天而降。

Boss A**：**

技能：在消滅此Boss之前，他會每間隔一段時間，召喚出一隻普通敵人。

Boss B**：**

技能：在消滅此Boss之前，他會每間隔一段時間，召喚多隻字數為1的敵人，以扇狀的方式朝遠方發射。

特殊功能：

當身邊有太多敵人即將接近，而且來不及將他們消滅的時候，玩家在每場遊戲，有三次的機會可以使用技能。按下"Enter"釋放出電磁脈衝(EMP)，快速的將身邊的敵人消滅。

密技：

作弊碼(CheatCode)，在”關於”頁面輸入**：**

104590025：開啟Debug功能

104590029：解鎖全角色

1. **遊戲圖形**

我們遊戲圖形是以8 BITS復古風作為設計風格，它最大的特色就是在圖形的邊緣，看起來會有明顯的鋸齒。為了達成像素畫(Pixel Art) 的美術風格，大部分的圖形我們都是利用最基本最傳統的“小畫家”來繪製。和一般的繪圖軟體相比，我認為小畫家非常適合來製作點陣圖，因為他可以很精確的在每一格中填色，正好符合了我們的需求。

敵人：

普通敵人**：** C:\Users\sean\AppData\Local\Microsoft\Windows\INetCache\Content.Word\faces.bmp

1字小怪**：**



BOSS：

\\Mac\Home\Pictures\fraps\fin\bosses.bmp

角色：

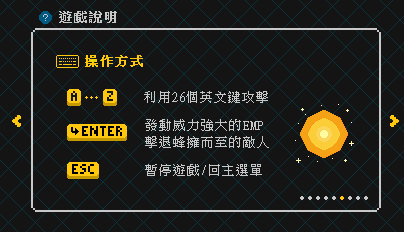
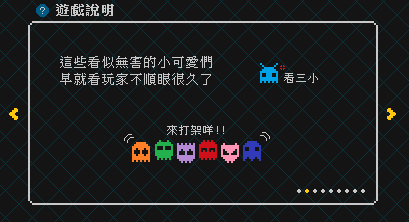


選單及界面：

開頭選單**：**



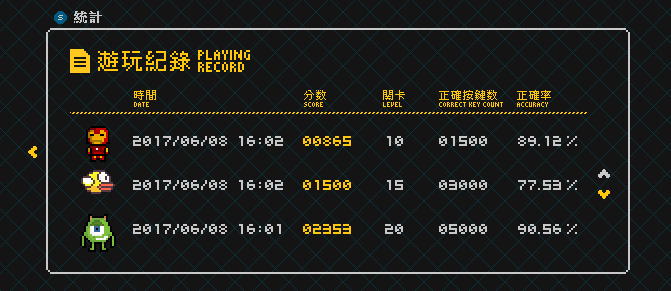
遊戲說明頁面**：**

角色選擇頁面**：**

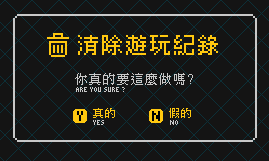
 

統計頁面**：**

關於頁面 / 背景音樂(BGM)及音效(SE)開關 / 清除遊玩紀律**：**







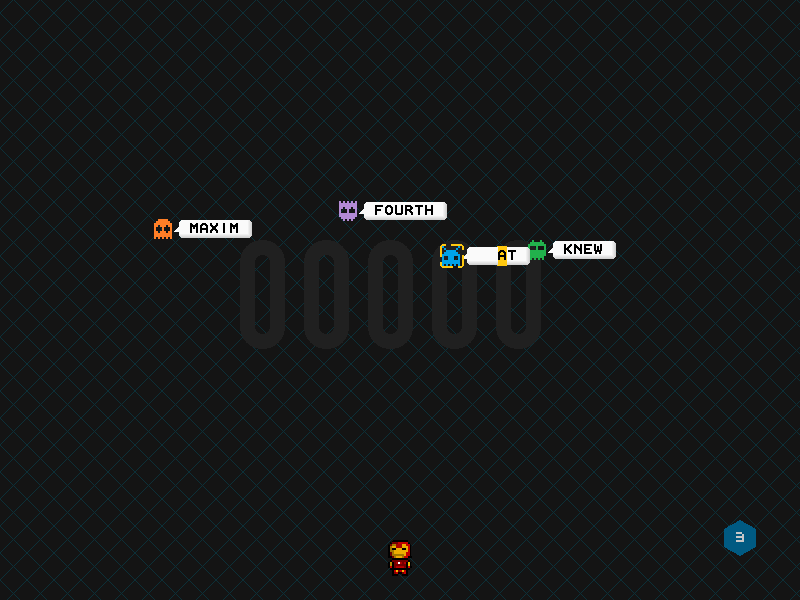
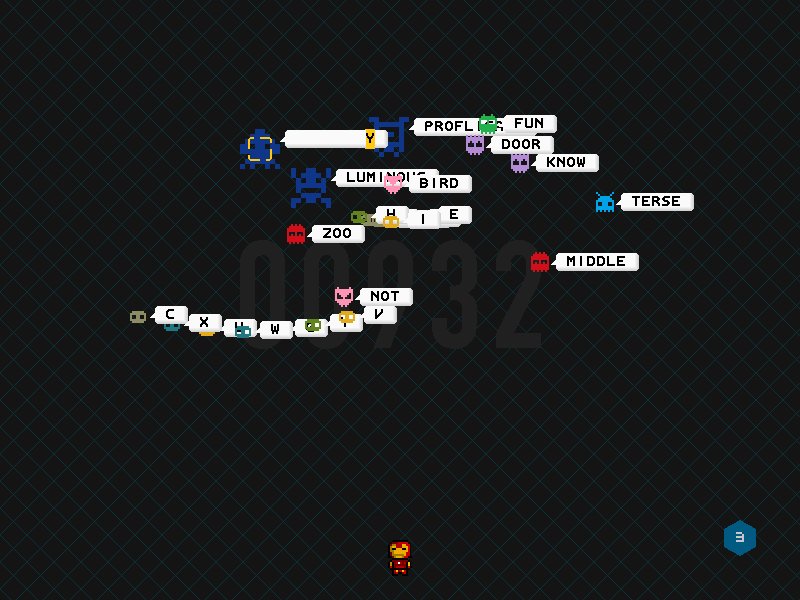
關卡切換的動畫/ Gameover結算畫面**：**

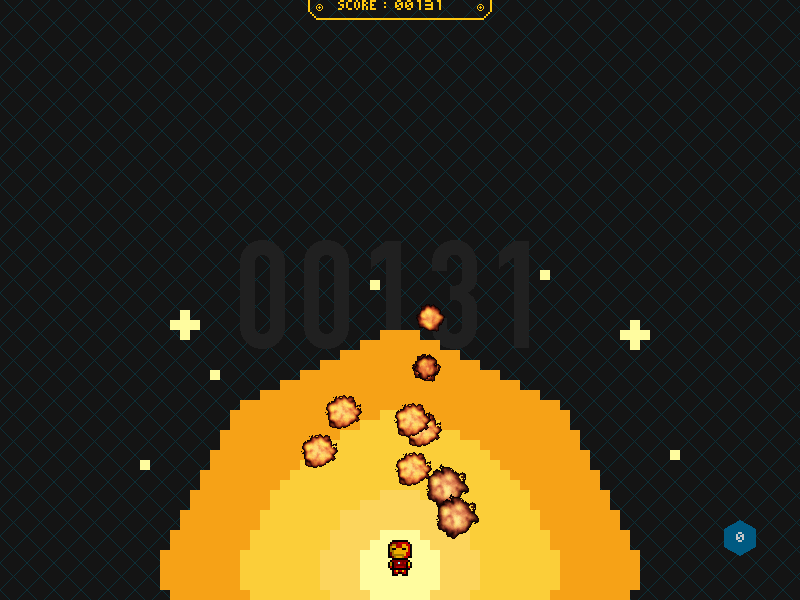


遊戲暫停**：**



遊戲畫面**：**



1. **遊戲音效：**

遊戲背景音樂：

取自於YouTube音樂庫中的The\_Coming\_Storm，營造出緊張刺激的感覺，提升遊玩上的好感度與刺激性。

正確按鍵音效：

輸入正確的按鍵，會發出酷似手槍射擊的聲音，讓使用者在遊玩上獲得激勵。

錯誤按鍵音效：

若輸入錯誤，則會發出「喀喀喀」的音效，提醒使用者請勿盲目地亂敲鍵盤。

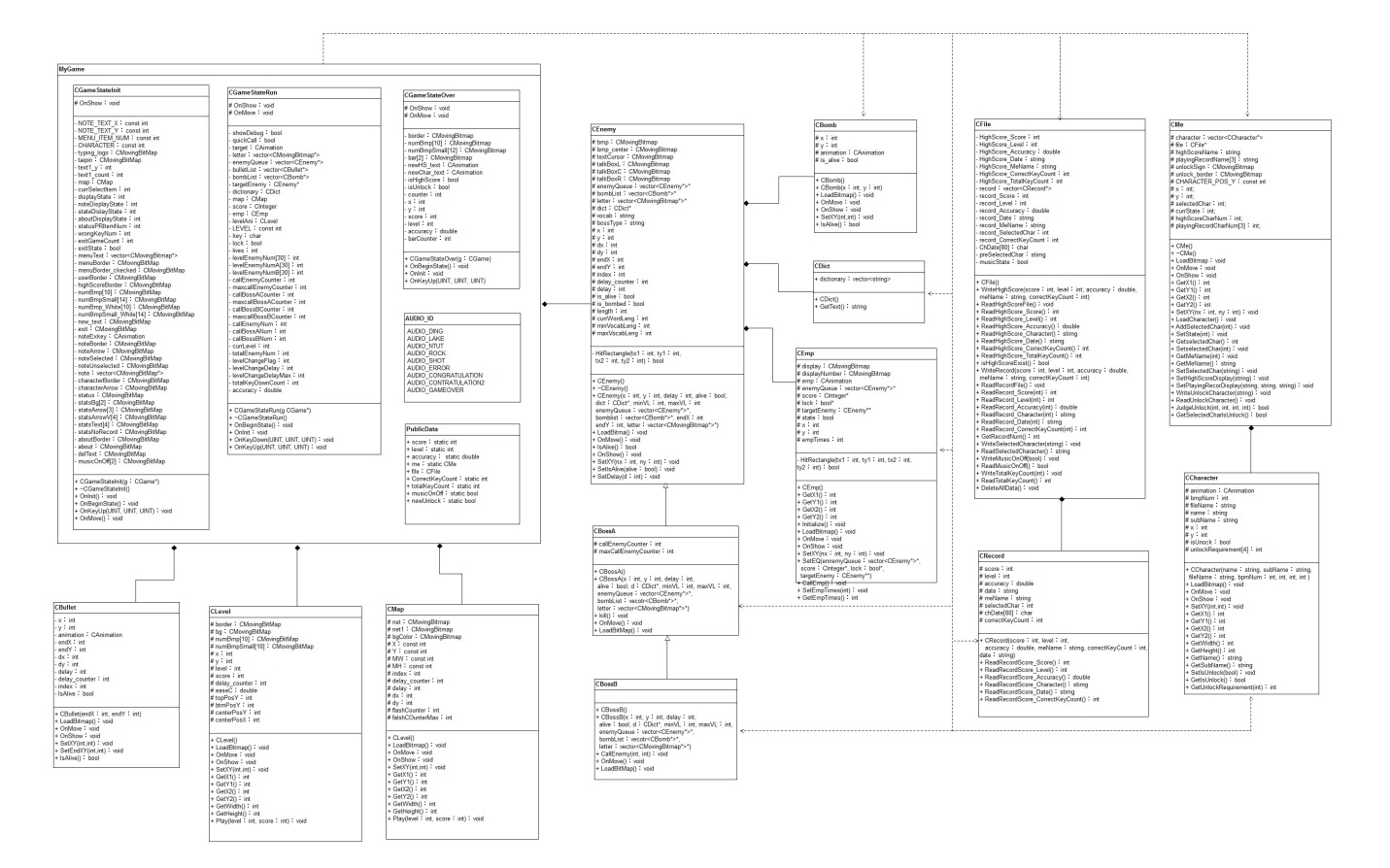
EMP(電磁脈衝)音效：

利用「爆炸」的音效，令使用者在狗急跳牆之際得以獲得一絲喘息空間。

GameOver音效：

除了提醒使用者遊戲結束，利用詼諧逗趣的音樂，令使用者想再重新來過，提升自我實力！

1. **程式設計**
2. 程式架構：



1. 程式類別：

|  |  |  |  |
| --- | --- | --- | --- |
| 類別名稱 | .h檔行數 | .cpp檔行數 | 說明 |
| CEnemy | 65 | 205 | 普通敵人，會朝特定方向飛行。 |
| CBossA | 15 | 92 | 魔王A，每隔一段時間會召喚一隻Enemy。 |
| CBossB | 12 | 104 | 魔王B，每隔一段時間會以扇狀的方式召喚多隻字數為1的Enemy。 |
| CEmp | 39 | 139 | 電磁脈衝，使用後會依照動畫清除範圍內的Enemy |
| CBullet | 25 | 71 | 子彈，從主角的位置不偏不倚地射向敵人。 |
| CBomb | 23 | 55 | 爆炸效果，當敵人死亡時便會顯示此效果。 |
| CMe | 52 | 345 | 顯示主角的圖樣，在三個遊戲狀態中皆會以不同的形式出現。 |
| CCharacter | 31 | 94 | 記錄主角的屬性：外觀與解鎖條件。 |
| CFile | 67 | 262 | 記錄遊戲結果，可以讀寫TXT檔，用以儲存最高記錄、每場遊玩記錄及選擇的角色等。 |
| CRecord | 33 | 53 | 顯示每場遊玩記錄時，用來儲存每筆資料裡面的分數、關卡、按鍵數、正確率及角色。 |
| CMap | 24 | 58 | 在遊戲畫面顯示持續移動的網狀背景。 |
| CDict | 14 | 46 | 字典檔，用於讀取txt文字檔中的單字，並儲存於vector。 |
| CLevel | 27 | 115 | 遊戲內中的換關動畫。 |
| myGame | 194 | 1261 | 遊戲程式碼本身，概括分為：遊戲開始、遊戲進行中、遊戲結束下去著墨、延伸 |
| 總行數 | 618 | 2900 |  |

1. 程式技術：

亂樹種子(Rand) ：

我們利用系統時間產生亂數，令敵人落下時的位置具有隨機性。但因為美術圖外型的關係，所以我們以預防了會友跑出畫面外的狀況。

三角函數計算：

在敵人BossB的設計中，我們設定它每隔一段時間便會以扇狀的方式召喚一群小怪，而在扇狀的計算中， 我們利用了角度與徑度的換算、二維座標定位採取極座標之運算，利用迴圈與變數，讓就算有再多的小怪，也可以精確地計算出位置， 但考慮視覺的舒適度與畫面的流暢度，我們最後將小怪定為7隻。

繼承(Inheriance) ：

由於落下的敵人不只是一般的普通小怪，還有難度提升的BossA與較難應付的BossB。在設計的過程中， 我們將BossA繼承原有Enemy，並新增有別與原有Enemy的美術圖與新的時間差，再利用下一項提及的「多型」設計新功能函數。 而BossB更繼承了BossA，設計出了極具巧思的特有功能。

多型(Polymophism) ：

在多型方面，我們利用了虛擬函數(Vitual Function)。我們在普通小怪、BossA、BossB上做了明確的區分，但因為是繼承關係， 所以在載入圖片、移動(move)函數與死亡設定(kill)函數上運用了虛擬函數做出獨特性。

Vector、迭代(Iterator) ：

我們採用Vector的方式來存放場上現有的敵人，而非利用Array來儲存，會這麼做的原因有以下考量。使用Vector可以快速的在集合尾端插入資料，而且也支援隨機存取，方便我們可以將已死亡的敵人，從陣列中移除。同時也可以利用.size()函數，很方便的取得當前場上敵人的總數，用來判斷是否可以接到下一關，利用Iterator和迴圈可以從頭到尾尋訪所有的敵人，鎖定特定條件的敵人。

遊玩記錄之保存：

我們利用了大一所學習的檔案讀寫功能，實作出歷史紀錄的保存，同時我們也讀取出最高分的紀錄，來提供玩家一個超越的指標，同時達到振奮效果。除此之外我們也未忽略使用者的遊玩體驗，在小細節上，遊戲也會記錄下玩家上次遊玩所選擇的角色及音效的開關狀態，下次開啟遊戲時，即可保留玩家的使用習性。

作弊(Cheat)、除錯(Debug)功能：

在「關於」頁面中我們暗藏作弊碼，只要輸入正確即可快速解鎖全角色與開啟Debug的功能。Debug功能中，我們利用快捷鍵即可開啟顯示怪物列表及狀態，與快速清除怪物。在遊戲狀態的操作上，我們實作出自由進出State，可以加快我們遊戲處理速度。

1. **結語**
2. 問題及解決方法：

在利用vector.size()來跑For迴圈時，出現"將警告視為錯誤處理 - 沒有產生 'object' 檔案"之錯誤。

以vector.size()，它回傳的值是unsigned int，而平時撰寫程式時，通常習慣直接使用int來宣告i，然而此framework對於形態的規則比較嚴格，因此他將int和unsigned int視為不同的東西，因此只需要將變數i設定成unsigned int即可排除此錯誤。

利用CDC螢幕字形來顯示怪物身上的單字，當怪物數量變多時，遊戲會嚴重卡頓。

原本遇到這個問題時，還不知道為何遊戲執行到後期，會越來越卡頓。後來看到了老師的Framework註解中提到說，盡量不要以此方式顯示文字，而是以CMovingBitmap來取而代之比較好。於是我們該採，將26個英文單字都製作成一個bmp檔，並且研究比對各種不同字體所呈現出來的效果，於是再次執行後果然效率得到大大的改善，執行更為流暢。然而使用Bitmap來顯示文字有個缺點，就是文字的顯示位置必須謹慎拿捏，因為我們是靠迴圈來一個個貼上字母，經過反復的測試才抓到最佳的顯示位置。

Float與除法運算時，計算結果總不正確。

經過上網查詢資料後，發現float運算時有小陷阱，這個陷阱以前我們未曾留意到，是在計算怪物飛行偏移量dx,dy時才首次發生的。例如計算double x= 3/2;，此預期的結果x應該要等於1.5，然而運算結果總等於1。這是因為算式的3和2都被視為int，因此3/2整體也被看做是一個int。後來發現只要預先將數字或變數強制轉型成double，就可以解決此Bug。如上面例子所示，可以改寫成double x= double(3)/double(2);。若是針對常數運算，也可直接在該整數的後方加上小數點，例如double x= 3.0/2.0;，如此一來x也能得到正確的運算結果。

在KeyDown事件中，明明使用者只按下一次按鈕，但程式實際卻會多執行數次。

這個Bug有點類似於設計電路時的彈跳，雖然表面上看起來只按下了一次，但程式的Cycle卻是不斷的在跑，這對於我們這種單字消除遊戲是非常困擾的，例如若出現連續字母的單字(BEE)，那在輸入到第2個字母‘E’的時候，會連帶的也把第三個字母‘E’也給消除了。於是我們使用一個字元變數key，來儲存按下的按鍵。每當OnKeyDown被執行時就判斷當前所按下的按鍵是否和key相同，否的話則正常執行並將當前按鍵存入key中。而當OnKeyUp時再將key變數設定為NULL，表示已放開了按鍵。這麼做之後，不僅解決了彈跳的問題，而且若使用者輸入速度過快，同時按下2個按鍵時，程式依然可以正確的判定。

在三個不同State間，想要共用某些變數。

我們在MyGame裡面額外新增了一個Class，專門儲存共用的變數，ex:檔案(File)的讀寫、角色(Me)的參數、分數(Score)、正確率(Accuracy)等等...，這樣成功的解決在三個Class之間變數的進出。

1. 時間表：
2. 貢獻比例：

黃省喬：60%

余鎧企：40%

1. 檢核表：

|  |  |  |
| --- | --- | --- |
| 項目 | 完成否 | 無法完成的原因 |
| 1 | 解決 Memory leak | ☑︎已完成 □未完成 |  |
| 2 | 自定遊戲 Icon | ☑︎已完成 □未完成 |  |
| 3 | 全螢幕啟動 | ☑︎已完成 □未完成 |  |
| 4 | 修改 Help->About | ☑︎已完成 □未完成 |  |
| 5 | 初始畫面說明按鍵及滑鼠之用法與密技 | ☑︎已完成 □未完成 |  |
| 6 | 上傳 setup 檔 | ☑︎已完成 □未完成 |  |
| 7 | 報告字型、點數、對齊、行 距、頁碼等格式正確 | ☑︎已完成 □未完成 |  |
| 8 | 報告封面、側邊格式正確 | ☑︎已完成 □未完成 |  |

1. 收獲：

104590029 黃省喬：

這是我第一次撰寫架構完整的遊戲，對於老師所提供的Framework也很陌生，一切都得憑藉著自己的摸索，才得以慢慢的撥雲見日。而在開發遊戲的過程中，我也得到相當多的收穫，學到很多不同的新知。

首先在以C＋＋來撰寫遊戲時，和以往的程式作業﹣僅僅透過console視窗來輸入指令進行運算不同。為了使得圖片在畫面中顯示，必須使用OnShow函數；為了使得圖片在畫面中移動，則要使用OnMove函數。這些對我來說都是非常嶄新的概念，這些函數皆各司其職且缺一不可，他們彼此間相輔相成，才造就起遊戲的穩固根基。

我了解到OnMove函數每秒執行約30次，所以我們便可以利用此特性來設計出各種動畫，如背景網狀的動畫、切換關卡時從上往下滑的提示文字、瞄準怪物時的準星動畫等，除此之外各種判斷的程式大多也在OnMove裡完成，例如搭配計數器(counter)，就可以設計每間隔一段時間便召喚一隻怪物。

由於我們設計的是“單字消除遊戲”，因此OnKeyDown的事件就顯得格外重要。在開發時我們就遇到了一個問題：在OnKeyDown時，會有類似以前設計電路時按下按鈕時彈跳的窘境。你以為你只按了一下按鍵，但程式的cycle卻是不斷在執行的，所以在按下按鍵到放開按鍵這中間，OnKeyDown可能就被執行了超過10次。為了解決這困擾，我們新增了一個變數key，用於儲存按下的按鍵。每當OnKeyDown被執行時就判斷當前所按下的按鍵是否和key相同，否的話則正常執行並將當前按鍵存入key中。而當OnKeyUp時再將key變數設定為NULL，表示已放開了按鍵。

所幸有Astyle提供自動排版，讓我們在撰寫程式的過程當中省了不少心，而看到老師提供的Framework，我才驚覺到原來程式碼和註解的對齊也是一門學問，僅有Astyle還不夠，若可以適時善用TAB及倒斜線換行來讓程式碼對齊，那麼對於日後閱讀程式碼可有相當大的助益。

為了精簡程式碼並使其更易讀，我也學習到在載入Bitmap時，可以善用迴圈及sprintf函數來讀取路徑。例如想要隨機顯示敵人皮膚的Bitmap，我們可以預先將圖檔都編號(faceXX.bmp) ,程式中再搭配sprintf及rand函數隨機載入Bitmap。我認為用這種方法，遠比將所有的路徑儲存於字串陣列中，還來的更有效率。

因為以往寫的程式執行的時間不會太長，通常輸入完沒幾行指令程式就結束了，基本上都是簡單的運算及文字處理，所以我一直都忽視了記憶體佔用多寡的重要性。而在設計較大型的遊戲時，妥善的記憶體分配之重要性更是不容小覷。為了減少記憶體的浪費，某些常用且經常重複出現的元素，我選擇將它們宣告在mygame的protected之中，再利用指標的方式傳入需要的class中。例如我們有一個字典class叫做CDict，其用途是自txt檔讀取單字字串並存入vector，並可以回傳隨機一個單字。最初我們是在每一隻敵人(CEnemy)裡都獨立宣告一個CDict，但在中期時我們發覺，其實沒必要每new一隻敵人，同時帶著一份笨重的CDict。因為CDict裡的vector儲存了上百筆的字串，隨著敵人的數量增加，想必會對於記憶體造成極大負荷。於是我們採用讓每個敵人都以指標的方式，共用同一個CDict，結果不出所料，原本生成敵人時會有的卡頓感瞬間煙消雲散。

在vector的運用上，我學到要清空vector前，必須先將內部的資料都delete才行。以下這句金句更是狠狠地烙印我心：“new了什麼就要記得delete什麼。”

104590029 余鎧企：

從上學期的火車訂票系統中，學習到了程式架構，便以為自己懂了，直到這學期實作出這個Typing Typing的遊戲，才對整個程式有深刻得體悟。剛開始拿到老師示範的Framework，先理解大方向的三個GameState，再從GameStateRun開始著手了解。得知BeginState跟Init的差別。前者為每次都要進入的狀態，後者是遊戲一開始時便Load的資源與設定的初始值。了解到KeyUp與KeyDown的不同，KeyDown為按下便一直執行，KeyUp為按鍵起來才會執行，這方面卡住我們蠻多時間，主要反映在玩家玩遊戲時的體驗，為了整體的流暢度，我們便設定出一個變數，類似電路防彈跳的功能，再按下時儲存按下按鍵，放開時在清空儲存值，便成功了提高整體順暢度。

OnMove為遊戲主要在移動的元素，我們從一開始從Framework中移動的物體開始延伸，一開始研究紅球落下，到後來加入了數學運算，成功控制了位移的角度，而後許多東西也伴隨著產生：小怪的誕生、射出的子彈、每隻BOSS的獨特性。而我們也實作出動畫的效果，利用許多張照片中微妙的差異，置入於OnMove函數中，使肉眼產生了動畫的效果，類似EMP(電子脈衝)、主角中的微妙動畫、看似再移動的地圖背景，這也歸功於Framework中會變色的彈跳球(Bouncing Ball)。OnShow的部分，在後期的程式上，出現了一個突破性的進展，若一直從外部引用圖片，在最後會造成了程式當機。這是我們壓根都沒有想到的，而我們解決辦法是利用了老師在網站的教學，載入到Resource File的方法，解決了程式的負荷。

在音效的部分，我們利用了網路上的開放資源，Youtube的開放音樂庫與開放合法下載的音效網站，方知有這麼多的資源供我們使用。而在CAudio這裡讓我學了一課：我們並不能無條件Stop音效，因為並不知道它是否正在播送，唯有確定它在開始狀態才可暫停。而突如其來的暫停會使程式當機！

而這遊戲特別的功能：歷史保存，我們將使用者的紀錄一筆一筆的儲存下來，用的是大一學過的File處理，在這Class設計上，我們參考了火車訂票系統中的Class Diagram：TicketOffice與Ticket、Station的關係，CRecord中單純紀錄單筆資料，在CFile中以Vector方式儲存CRecord，再利用Vector與Iterator的特性迅速的回傳所需的第X筆資料。這個用法在CMe與CCharacter也有使用到，利用了這層關係，使我們在程式架構上更加得清晰，不會在最後彙整的時候產生紊亂。

1. 心得、感想：

104590029 黃省喬：

從小到大我接觸了各式不同種類的遊戲，並且從那個時候就對電腦遊戲感到無比的著迷。對於一個好奇心旺盛的少年來說，電腦遊戲為平時枯燥乏味的求學生活帶來無限的溫暖曙光，探索在無垠無涯的虛擬世界裡，樂趣無窮，我想這就是所謂的青春吧。

當我得知物件導向實習課，是要以寫遊戲來呈現時，其實我是十分興奮的。在經歷上學期OOP課程的洗禮後，我對於物件導向的概念已有初步的認識及瞭解，但若無法將課堂中得到的知識學以致用，那我認為學的再多都是枉然。而這堂OOP實習課，便給了我們這千載難逢的機會，讓我們可以將腦中無數創意想法付諸實際，得以活用程式設計的技巧並增加邏輯推理的能力。

而以製作遊戲來當做期末專題，對我來說也是相當耳目一新，畢竟玩過了無數大大小小的遊戲，現在卻要換我們來寫出一個遊戲，無疑是一項艱鉅的挑戰。但漸漸地，我發現我越來越沉浸於撰寫程式和DEBUG的過程當中，只要是閒暇之餘就會把程式給打開來寫，甚至到了有些廢寢忘食的地步，這真是奇妙的感覺。

無論是突然的靈光乍現解決了臭蟲，或是編輯素材時，一筆一劃刻畫出的怪物圖片，能憑藉著自己的力量，一步一個腳印的使遊戲壯大，由淺入深的內容變得豐富，看到最終完成的作品，那股成就感是不言而喻的。

對我來說這不僅僅是一門實習課，除了獲得學分之外，我認為在製作專題的過程中學習到的東西才最難能可貴。在遭遇困難時，懂得利用網路資源尋求解決方案，或是和同學請教互相討論，在如此潛移默化下，我在撰寫程式時的思路變得更加清晰有脈絡，在無形中學到的東西更遠遠超過原本我所期待。

104590025 余鎧企：

在國小的時候，參加過大大小小的打字比賽，中英打皆有，過程中藉助了很多市面上的打字練習軟體，所以對於打字練習並不陌生。有天碰上了ZType這個練習網站，便深深的著迷，並不知道原來藉由華麗的遊戲與酷炫的特效，同樣也可達到練習效果，原以為遊戲，就是拿來玩，打發時間用的。

到了大學進入資工系，從C開始累積基礎，到二上的C++開始對於程式有了初步的認知。尤其是強迫一定要新增Class的時期，起初覺得過於麻煩，一個小的程式為什麼要新增Class呢？直到了「火車訂票系統」，我認為它是對於我寫程式的一個轉捩點。一個Class Diagram、說明程式所需與限制。我認為我終於像是在寫程式了。

進入下學期的OOP，知道要從模仿一個遊戲開始時，便十分慌張與猶豫，怕一個輕易決定，都會超過我們目前的能力。最後與夥伴討論出實作出一個「打字練習」為主題的遊戲。起初對於FrameWork得進行相當無助，但縮小範圍，一步一腳印地慢慢看，很快便有了起步。我逐漸上手，先從簡單的移動到加入學過的數學式子，便開始步入主題至打字消除，其中經歷了不少的困難，像是記憶體的耗費、美術圖的應用、人性化的控制，都花了我們不少的時間。我們才知那看似簡單的功能，實際上卻非常的消耗腦力，愈是微小的地方，花的精神愈是甚多。

成功消除之後，便開始延伸至BOSS的特性，主角的應用。我們在BOSS功能與特性上花了多巧思，不然即失去了意義。在主角的設計上，我的夥伴省喬他發揮了精湛美術天賦，利用8Bit畫出了我出乎意料的圖形，也讓我對於美工有了更上一層的體悟。

到最後新增了音效與特效，發現好的程式，其中也包含了好的視覺與聽覺享受，就像是ZType帶給我的感動一樣。

在這過程中，我們不斷精進自己。培養解決問題的能力與團隊合作、溝通的能力。比起上學期獨自寫一個程式，團隊合作更顯得可貴，畢竟市面上的程式，都不是一個人可獨自完成的。學會了在沒有老師的奧援下尋求網路得到解答，或是與同學共同討論。學會了清晰的規劃程式架構，不至於在之後的結合產生紊亂。學會了以使用者的角度做設計，深刻理解到程式不只是給自己使用的。最後感謝這堂課帶來給我無窮的受益。

1. 對於本課程的建議：

# 附錄