計算機程式期末個人專案

姓名：林家毅

系級：國企三

學號：B04704016

一、組內分工表

組員姓名	工作內容
林家毅	噴射機物件 Class 子彈物件 Class Main function 功能 遊戲介面與圖像設計 Demo 影片剪輯與後製
江宥呈	敵人物件 Class Class 繼承 Main function 功能 Demo 影片素材錄製
林祐丞	開始與結束頁面 Class 技能物件 Class Class 繼承 Main function 功能

二、個人工作內容簡介

專案前期：

1. 設計遊戲介面（物件圖樣、顏色、字體、排版）
2. 建立 Spaceship Class （噴射機），使其跟隨滑鼠指標位置旋轉
3. 建立 Bullet Class （子彈），使其隨噴射機指向發射

專案中期：

1. 將子彈數量從有限變為無限
2. 處理子彈與射擊敵人而使敵人消失的功能

專案後期：

1. 遊戲介面優化（加入技能特效、等級提升功能與特效）
2. Demo 影片剪輯與後製

三、程式碼部分之詳細內容

Spaceship Class：

1. 為了讓噴射機能隨著滑鼠指標位置旋轉，用到 Class 中的一個 private 變數 angle 和兩個 public 函數：

   void render(int x, int y, int width, int height);

   （spaceship.cpp 26~30 行）

   void handleEvent(SDL_Event* e);

   （spaceship.cpp 32~41 行）

   handleEvent() 用來偵測滑鼠是否在螢幕內移動，若有則會將滑鼠當下的位置記錄下來，並透過 atan2(SCREEN_HEIGHT / 2 - y, SCREEN_WIDTH / 2 – x)) * 180 / 3.14159265 – 90 的計算式，將值 assign 給 angle，因為滑鼠位置與噴射機中心的向量可以換算出旋轉弧度，再使用 180/π 轉換成角度，最後加上常數調整正確的角度（這邊是 90 度）即可。

   render() 會直接接受四個參數，包含 x 軸座標、y 軸座標、自訂物件寬度、自訂物件高度，目的是為了在 render 出物件的時候就可以直接控制好長寬，而不用另外寫函數去設定。另外在函數中會使用到 handleEvent() 所不斷改變的 angle ，在這個 Class 中把 angle 設為 private 變數，因為我們沒有任何時候需要去特別修改它的值，而是只需要函數所給我們的結果，在不斷的執行 handleEvent() 和 render() 的過程中，噴射機就會隨著滑鼠指標旋轉了！

2. 噴射機的生命（在噴射機後面的線條）會根據當時的生命值顯示對應的樣貌，使用到 public 變數 life、public 函數 getLife() 和 main function 中的 switch() 控制：

   int getLife();

   （spaceship.cpp 43~46 行）

   switch(theSpaceship.getLife()){...;}

   （main.cpp 287~308 行）

getLife() 用於回傳 life 變數的值，在 main function 裡為了分別達到讓生命減少和回復生命的功能，使用 life -= 1; 和 life = 4; ，不過其實也可以使用另一種做法，就是將 life 改為 private 變數，並另外寫兩個函數（例如：loseLife()、resetLife()）來取得一樣的效果，不過因為這裡的結構並不複雜且不容易出錯，所以決定直接用 public 變數操作。

switch(theSpaceship.getLife()) 會根據噴射機生命值（介於 0~4）來判定要 render 哪一種圖示，這裡讓所有圖示物件都用 Spaceship Class 包裝，除了因為所需要用到的函數大多重複之外，也是因為直觀而言噴射機和生命基本上會同時並存，因此用同一個 class 包裝再適合不過了。

3. 噴射機等級的升級功能也包裝在 Spaceship Class 中，這裡的想法是當分數每到一個升級門檻時，就讓「Level Up」的圖示短暫呈現，因此會需要用到 Ticks 相關的功能來維持呈現的時間：

```
public :
    bool levelUpIsShowed();
    void showLevelUp();
    void dontShow();
    Uint32 getTicks();
    bool checkShow();
```

（spaceship.cpp 57~97 行）

```
private :
    Uint32 mStartTicks;
    bool mStarted;
    bool show;
```

與升級功能相關的變數大多需要在各個函數中改變其值，在這種複雜的情況下不太可能從 main function 中直接改動它們，因此宣告為 private 變數，而透過各個 public 函數進行理想的操作。

當判定升級時，會讓 mStarted 、 show 值改為 true，並記錄下當時的 Ticks 給 mStartTicks；此過程中的重點是用 showLevelUp()、checkShow()、dontShow() 三個函數分別控制 show 變數，並用 Ticks

控制物件 render 的時間長度，以在 main function 中順利執行 Level Up 的功能。(main.cpp 110~114, 337~352 行)

Bullet Class：

1. 子彈 Class 除了跟噴射機 Class 一樣包含圖檔、長度、寬度、角度之外，由於子彈尚需要向外飛出，故多了兩軸向量速度的變數；且為了達到特定效果 (下方詳述)，設計了三種 render 函數，以下為相關的變數、函數宣告：

```
public：
    double x_v;
    double y_v;
    double angle;
    void handleEvent( SDL_Event* e );
    void render(int x, int y, int width, int height);
    void render(int x, int y, int width, int height, const
    double a, const double b, const double c);
    void moveWithoutRender(int x, int y, int width, int
    height, double a, double b, double c);
```

(bullet.cpp 36~77, 128~136 行)

首先，Bullet Class 中的 handleEvent() 與 Spaceship Class 有所不同，差異之處在於除了透過角度計算公式算出子彈應該旋轉的方向之外，也使用滑鼠位置與中心點向量的單位向量，作為飛行方向的基礎 (即會透過公式算出值 assign 給 x_v, y_v)，並透過 void render(int x, int y, int width, int height); 函數使子彈飛行。
(main.cpp 134~138 行)

然而，一開始只使用這樣的設計，會發現子彈自射出後，仍會跟著滑鼠移動，而改變飛行角度和飛行方向，另外因為 Class 中的 const、static 用法與在 Class 之外的用法有所差異，所以這裡並沒有在 Class 內使用這些功能來固定子彈飛行角度和飛行方向。

經過一番嘗試，才找到方法將子彈的 x_v, y_v, angle 變數用宣告在

global 裡的 const pointer 固定，並配合 Class 中的 void moveWithoutRender(int x, int y, int width, int height, double a, double b, double c); 成功讓子彈如原本的預期方式飛行，關於為何在這邊使用 moveWithoutRender 效果則在之後會另外列點說明。
（main.cpp 140~154 行）

關於這部分還有另一個細節是 constructor 的設計（bullet.cpp 4~16 行）與 main function 的結合 （main.cpp 134~154 行）。這裡將該段 main function 切割成 if-else 兩段，原因是希望只有當子彈在最為初始的狀態時，才用 handleEvent 去改變它的各項屬性，而一旦屬性改變且子彈尚未重製，便都只會使用到 const pointer 所指定的值來進行 render，因此，便先在 constructor 中設定好跟 if-else 中相互呼應的條件，來達成上述目的。

2. 以上所述只能控制一顆子彈，若要控制更多子彈則需要再經過特殊處理。雖然理論上有不少方法可以做到（例如：動態陣列），不過根據實際操作的結果認為先產生 100 個相同子彈，再根據方才所創立的子彈（簡稱領導子彈）的位置發射，是更好的方法。（main.cpp 194~222 行）

   根據該段程式碼，可以發現大致上跟領導子彈的發射方法相同，主要差異在於這些陣列構成的子彈有發射的條件限制，並且在滿足一定條件時重置。發射條件依據領導子彈位置判定，判斷式 theBullet.getDistance() >= distanceBetweenBullets * i 使領導子彈每向外飛出一個固定距離時，發射下一顆子彈，依此類推；判斷式 allBullet[i].getDistance() >= distanceBetweenBullets * 100 則是讓除了領到子彈外的所有子彈，一旦飛出螢幕的一定範圍，變會重置，而陣列中第一顆子彈重置的時間剛好是最後一顆子彈已飛出，且下一顆子彈該出現的時間，所以便創造了視覺上的無限輪迴。

3. 關於 void render(int x, int y, int width, int height, const double a, const double b, const double c); 和 void moveWithoutRender(int x, int y, int width, int height, double a, double b, double c); 兩個用

來 render 子彈的函數，是特別設計來製造子彈與敵人碰撞效果的函數。配合上與敵人碰撞的判斷（globalVariableAndFunction.cpp 572~583 行）與效果（main.cpp 157~192, 232~280 行），即可讓子彈射到敵人時消失，敵人也在受到摧毀時消失。

首先，兩個 render 函數的差異只在有沒有 render 出來給使用者看到，實際上所有陣列中正在飛行的子彈都不會停止飛行，這是為了避免子彈順序錯亂而設計的功能，也就是說子彈飛到固定的距離而重置是唯一且必定會發生的重置效果，碰觸到敵人只會讓子彈改用 moveWithoutRender() 方式移動，這是根據不斷測試後認為最可靠、穩定而有效的方法。

子彈與敵人的碰撞判斷則是用到數學觀念，由於子彈可以 360 度射出，敵人也會從 360 度向中心飛入，如果單計算兩個物件的向量，會導致沒辦法處理所有可能的結果。因此，透過 double bulletEnemyDistance(const bullet &thisBullet, const Dot &thisDot) 會將子彈物件和敵人物件以 const 的方式被函數呼叫，並進一步計算兩個物件中心的向量，同時兼顧物件資料不被隨意更改。

接著則是使用 main function 中的程式碼（main.cpp 157~192, 232~280 行），將子彈與每個敵人判定是否落入其半徑，若是，則會將子彈隱形，並讓碰到的敵人損失生命（如果需要的話），而一旦某顆子彈被判定碰撞到敵人，便會觸發 break ，提前結束該子彈的判定，如此一來就不用多浪費時間判斷該子彈還跟誰碰撞了。

最後要解釋關於子彈的 moveWithoutRender() 究竟是如何真的被執行，除了領導子彈永遠都是隱形的之外（只是拿來當作其他子彈飛出的參考點，只對開發人員有意義，實際上在遊戲過程中沒有任何功用），陣列中的 100 顆子彈都對應了 bool 變數 hitEnemy （globalVariableAndFunction.cpp 57 行），它們預設為 false ，即沒有觸碰到敵人，而只要在遊戲過程中，某顆子彈碰觸到敵人，便會讓該子彈的 hitEnemy 值改為 true，同時這樣的改變會讓 render 的方法從一般 render() 變成 moveWithoutRender，直到子彈飛到我們所設

定的界線範圍而重置之後，hitEnemy 才會重置為 false，並變回遠本 render() 的 render 方式，於是達到我們所在遊戲內所呈現的所有效果。

四、總結與專案心得

這學期下來取得了好多收穫，因為自己是來自管院的學生（很像也是這學期班上的唯一一個？!），從來沒有想過能透過寫程式，開發出一個真的能玩的遊戲，透過這個專案才真正感受到學以致用原來是這麼有成就感的事。

雖然知道自己程式能力大概只有中等程度，平常在討論區也因為實在擠不出足夠的時間細細感受講義中的內容以提出問題跟大家討論，且期中考成績公布之後又發現自己實在表現得太不如自己預期（很怕被當或是 GPA 爆炸哈哈哈哈），於是下定決心要好好的發揮所長完成一個滿意的專案。

從專案的一開始就花了很多時間設計好遊戲介面讓朋友們、組員們看過，才開始動工，由於以程式的總量而言組員之間差不多是平均分配，所以加上介面設計，實在需要多花不少時間，而且過程中遇到許多程式上的困難，也花費好幾天，甚至一個多禮拜的時間慢慢解決，遊戲才漸漸成型。而在專案後期時，也為了讓玩家在遊戲過程更有感覺，又另外設計出等級制度和技能視覺效果（跟自己的期末考作對 XDD），目的就是想呈現出最完美的一面給自己和大家看到。最後在專案即將完成的時候，組員們加緊為程式碼做繼承的包裝，而同組之間大家都沒有剪輯影片的經驗，加上我對於視覺又有莫名的要求，所以即使上傳期限在即，也堅持看了許多教學文章，剪出自己認為還算是有質感的 Demo 影片（希望老師和助教可以看看哈哈，真的剪了好久），才終於把專案完成。

謝謝老師和助教們這學期以來的教學，真的從中學到很多東西，雖然還有許多不足之處，但未來仍會找尋機會更加磨練自己的能力，老師、助教們都辛苦了！！