Java 期末專題

作品名稱：project B

作者：通訊 2A 0096c029 楊友嘉

目錄

壹、 摘要

project B 遊戲開發的所有歷程，與經驗分享

貳、 動機

經過一學期的 java 課程後，我對於期末專題的第一個想法即是遊戲設計，遊戲設計相較於一般的程式有許多好處，對此我列出以下幾點：

1. 更能引起同儕的興趣（可以幫忙改進、或提出建議）
2. 容易找到 UI/UX 的 bug（接續 1，可以跟同性質的遊戲做比較）
3. 對於介面的穩定性要求更高，必須有良好的邏輯規劃，避免用戶迷失在介面中
4. 對於如何設計角色的物理特性非常有興趣
5. 希望能帶給大家不一樣的感覺，而不是作業的生硬感

參、 準備

在製作遊戲的一個月前，我原本是想設計貪吃蛇的遊戲，並加入 ai 的自動運行功能，後來因為一些事件導致這個計畫蒸發，此時我放眼更加複雜的遊戲，剛好在 YouTube 上被推薦了一個影片（附錄 1），是一個相似於任天堂馬力歐的遊戲，但整體並沒有流暢的動畫，只有較為簡單的角色位移，陷阱跟背景也都是由非常簡單的像素所構成，覺得整體的設計應該蠻符合我的需求的，即開始依樣畫葫蘆。做了一些 demo，如工程計算機、貪吃蛇用來熟悉 java 的各種工具和 GUI，接著便開始動工。

肆、 設計流程

一、 介面設計

初始的介面使用 frame 搭配 penal，再配上 ActionListener 跟 Timer timer = new Timer(12, this)來達成角色與所有物件的刷新。固定視窗的大小，用座標的方式來完成 penal 上所有物體的定位，不去採用 label 或其他 penal 來增加設計的不確定因素。

二、 X 軸移動

因為還不熟悉正統的遊戲設計，一開始我並沒有用 object 來定義角色，而是給他一個座標，代表角色的圖層就會根據修改後的座標的 x y 軸變更位置。

加入一個新的線程，f.addKeyListener(new Bkey())，並加入ａｄ鍵功能，試過很多種方法，常常因為高速切換鍵盤而導致角色停下來，最後使用速度的概念，再加上判定才避免了這個問題。

```
If(按a) xv = -10;
If(按d) xv = 10;
If(a d都鬆開) xv=0;
X += xv;
```

三、 Y軸跳躍

1.跳躍：

Y軸的跳躍相比x軸困難許多，一開始是做的時候套用了x軸的速度公式，但整體的移動非常不自然，幾番思索後決定拿出高中的物理課本研究重力加速度，最後的成果如下：

```
a = 1;
if(按w) yv = -10;
if(yv!=10) yv += a;
y += yv;
```

2.二連跳：

當以為這樣就結束的時候，更可怕的問題來了，“二連跳”，二連跳是什麼呢，那我們要先從他的定義開始看起，二連跳的定義是**“當角色跳過一次後可以在空中做出第二段跳躍”**所以一開始設計是當第二次跳躍時再將yv改為10，但這又產生一個問題，二連跳會以那瞬間的高度作為新的地板，再做一次的跳躍，這樣會導致我在不同時間點按下二連跳會有不一樣的最大跳躍高度（如圖1）。

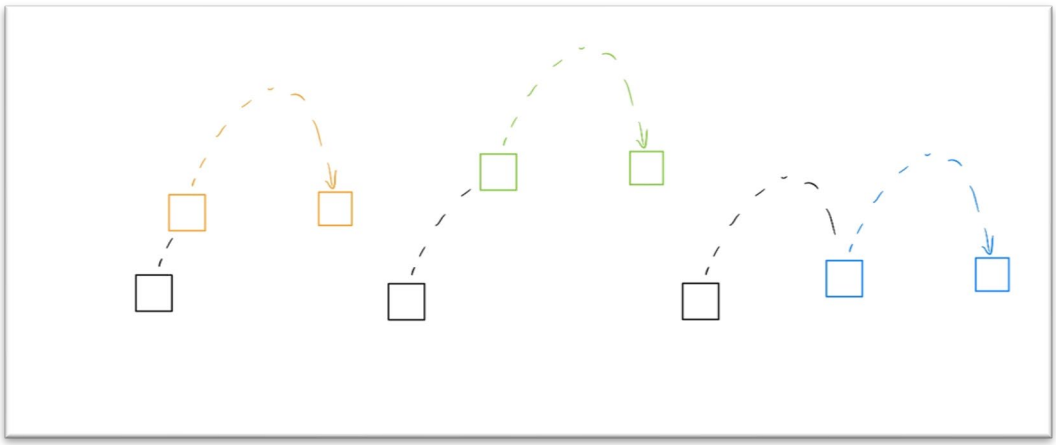
這個現象經測試後發現無法妥協，並嘗試解決，最終想出先設置最大跳躍高度，並在二連跳的同時依據剩餘的可跳躍距離來算出新的速度給yv(由距離推算速度），成果如下：

```
yv = -Math.pow(2*(jumphigh（最大限制高度）-groundtemp（二連跳起跳點）+y), 0.5);
```

這樣才得以順利的解決了整個問題。

Ps:按鍵的判定過於繁瑣就不再重複說明了。

圖1：



四、 場景判定

上一章講到關於起跳地面的判定，這一章來介紹整體座標的設計，因為那時的我對於 vector 和 array list 不太熟，而且剛好場地需要固定大小，思考後還是決定用 2D array 開工，概念有了下一步就是不斷的嘗試了，用 bubble 迴圈來讓整個地方填滿需要的數值，並且因為角色的本體（x y）只有左上角的一點（如圖 2），所以很多時候必須要減掉角色的 size。

```
set space(場地 x-size , 場地 y -size, 場地寬 +size, 場地高 +size);
for(i)for(j)
c[i][j] = 1;
```

除了障礙物的判定，我也加入了地板的判定，在每一方塊的表面都設置一層地板判定，讓角色在空中可以隨時讀取並刷新自己的地板，知道自己是在空中還是在地上，在空中就必須落下，在地板上就必須刷新跳躍的次數，供下一次跳躍做使用（如圖 3）。

圖 2：

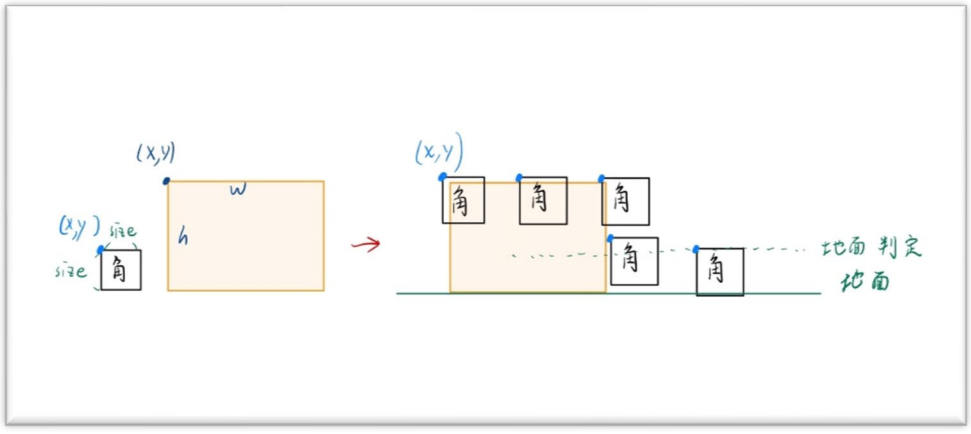
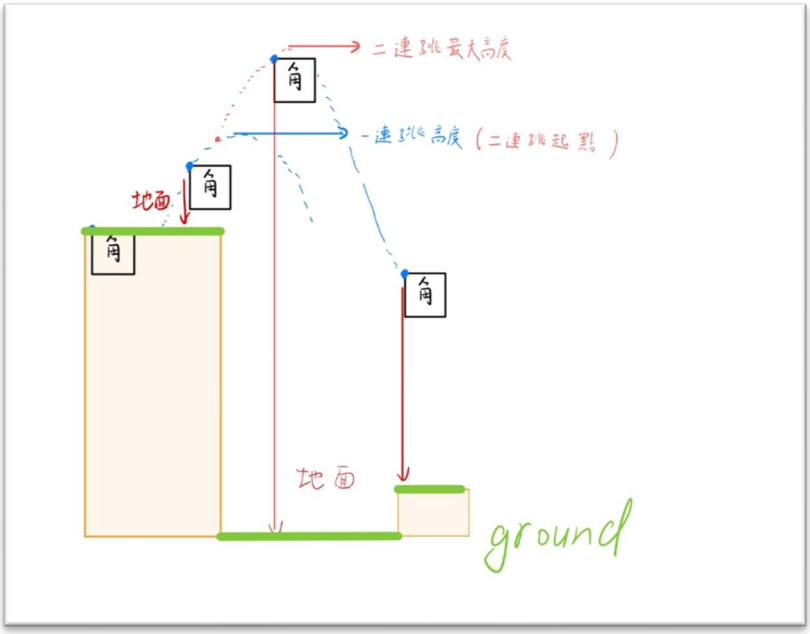


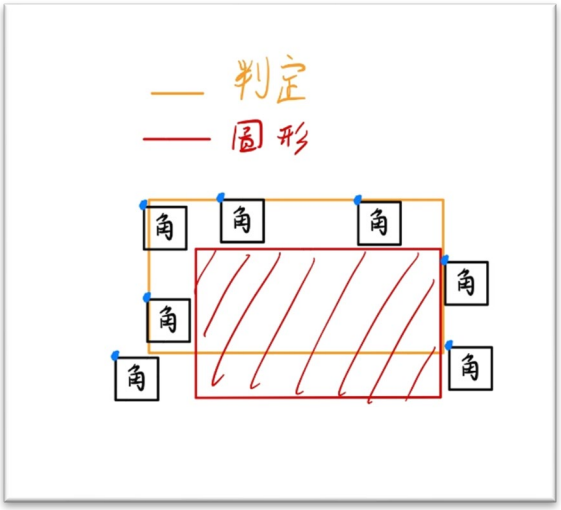
圖 3



五、 場景辦定 2（場景繪製）

繪製場景跟製作判定不太一樣，可以從（圖 4）中發現。
用 function 把輸入物體的（x, y, 寬, 高）一次完成兩邊的事情（繪製與判定）。

圖 4



六、 數值統一定位

寫一個 class 來存放所有數字再用 static 讓其餘每個 class 都可以存取到

優點：

1. 隨時可以做調整，在測試遊戲的過程中臨時要修改數值或想要切換場景可以直接修改
2. 可以隨時 print 出來，不管程式出了什麼 bug，都可以在 penal 中直接寫一段用來監看數據，查看哪些數值不正確（如圖 5）
3. 可以省略很多 function 的傳入和傳回，明確的知道每一個數值的所在地
4. 美觀，不會大海撈針找數據

缺點：

目前尚不知道這樣做會帶給程式什麼樣的負擔或違反什麼設計原則，唯一知道的應該是我應該先認識 SOLID

SOLID 是 5 大原則的簡稱，分別為：

S =單一責任原則（SRP）= 職責原則

O =開放-封閉原則（OCP）= 開放式

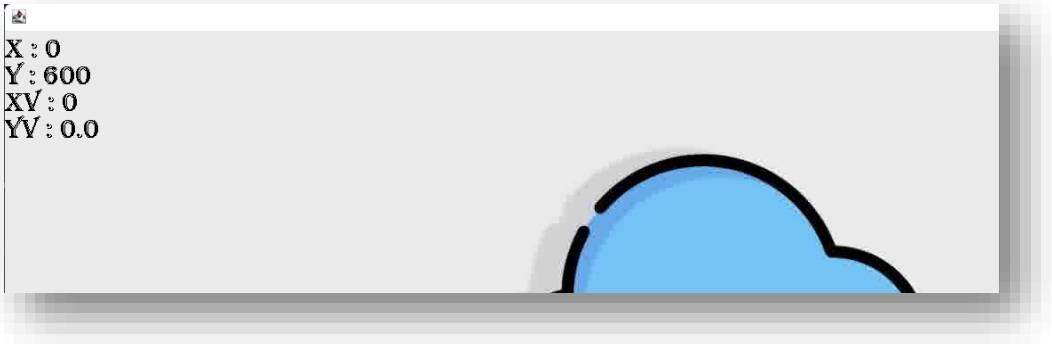
L = 里氏替換原則（LSP）= 里氏替換原則

I =接口隔離原則（ISP）= 面隔離原則

D =依賴倒置原則（DIP）= 依賴獨立原則

很明顯很多都沒有符合，但這只是個人專案，我覺得這點並不致命。

圖 5



七、 播放影片（死亡場景）

1. 陷阱

再來就是遊戲中最重要 part，陷阱

陷阱作為解迷遊戲中的經典元素，當然不能只有簡單的陷阱，我總共設計了三種陷阱

A. 普通陷阱，撞上去就老老實實重新再來

B. 假陷阱，看上去非常真實，但並沒有設定陷阱的判定，而且進入移動的範圍內就會消失（圖像的部分）
C. 藏陷阱，跟上者剛好相反，有設置陷阱判定，但要靠近才會顯示圖像

就靠著這三種陷阱來完成遊戲的配置。

2. 死亡場景（影片）

有陷阱那當然要有死亡畫面，死亡畫面亦開始當然是簡單的大字報顯示“you are dead”，後續在做改進的時候想到可以加入迷因影片的設計，就開始上網找資料，但發現 java GUI 只有插入音樂和圖片，並沒讀取影片的方法，最終我想到了如果用大量的圖片再搭配影片的音樂，應該可以模擬出影片的效果，上網用程式把影片轉成數百張的 jpg，使用相同的檔名加上數字來讀取，微調讀取的速度，配合上影片的音樂，還真的被我用出來了。

八、 地形建設

地形建設使用**場景辦定 2**和**數值統一定位**中提到的函示，只要輸入座標給矩陣就可以完成，輕鬆了很多。我會先用 ai（註：Adobe Illustrator）來繪製場景大概的定位，再回 java 裡反覆微調，最終完成場地建設（陷阱的部分同上）。中間有大改幾次地形，應玩家的一些反饋來修改難度，才成為現在看到的地圖。

九、 介面擴增

第一地圖、死亡和勝利場景都完成後開始規劃更多的地形，新增後場景數來到五個：
1. 選單
2. 第一地圖
3. 第二地圖
4. 第三地圖（時間因素暫不啟用）
5. 勝利
6. 死亡

其中第二地圖的設計理念恰恰跟第一地圖相反，第一地圖是由簡單的浮空島嶼所構成，可以大幅的減少卡牆的機率，第二張圖就以山洞為模型，以牆壁環繞道路，像是迷宮的感覺，而這個改變還讓我發現到很多原本沒有注意到的 bug，順帶把判定系統更新了一遍。

十、 加入選單與開場動畫

開頭的動畫是我做的，cloud animation 是我的動畫標籤，希望能有個人工作室出品的感覺，那跟結尾動畫一樣是用照片加音樂，再去微調使其配合。在選單，多新增了選到某個東西時有稍微放大的效果。

十一、 技能設定

技能，一個完整的遊戲怎能少的了技能呢？本遊戲礙於設計的理念，不想加入過多解迷以外的元素進來（例如：攻擊、武器、商店、經驗等），但如果是能幫助解迷的技能那就另當別論了。這次一共加了兩個技能，分別是時間刪除跟時間暫停，其靈感都是來自於經典動漫 JOJO 的奇幻冒險，因為在查資料的時候被強迫看了不少，越看越有感覺，而且技能內容剛好都是很好發揮的題材，就直接加到遊戲裡了。

1. 時間暫停

能力名：The World，為動畫中其中一個主角的技能（詳細情況說來複雜就線這樣解釋了）

能力解析：用法是當技能發動後會有幾秒的時間固定 y 座標，並將 x 座標的速度降為 1/8，可以在跳躍的時候使用，就可以實現短時間滯空的效果。

出現地點與解鎖方法：只有第二關可以使用此技能，而只有碰觸到第二關的燈泡才會看到詳細的技能解說與按鍵位置。

2. 時間刪除

能力名：King Crimson，為動畫中其中一個反派 boss 的技能

能力解析：如果有成功抵達第二關的玩家，在過程中不幸踩到陷阱，回到第一關時即會跳出此技能提示，按下 k 可以發動技能。按下之後會有一串動畫，並在動畫播放過程中將角色移動到第一關的終點，呼應了原作中的效果（將事物中間的過程刪除只留下結果）。

十二、 美化場景

1. 場地的美化，原本的場地是用內建色塊拼疊而成，而我要做的事情就是直接打開 ai 重新繪製一次場地，然後直接拿做好的圖去覆蓋掉原本的場地（圖 6、圖 7）
2. 角色的美化，讓角色在左右移動時面向自己移動的方向
3. 陷阱的美化，讓第二關採用不同的陷阱與不同的死亡動畫
4. 起始介面的美化，用色彩的漸變做開場效果，設計新的封面 logo
5. 結束畫面的美化，並且可以讓其存檔讀檔，讀取歷史記錄
6. 新增難易度，困難實在太難了，便新增一個簡單版，只有普通陷阱與第一關，降低玩家的破關難度
7. 技能提示與技能美化，所有技能都做了提示（圖 8、9）
8. 進入遊戲時的提示（圖 10）
9. 將所有場景分別獨立成 class，比較好做修改

圖 6

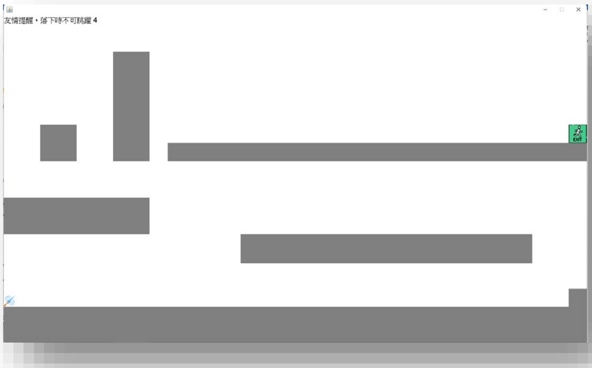

友情提醒，落下時不可跳躍 4

圖 7

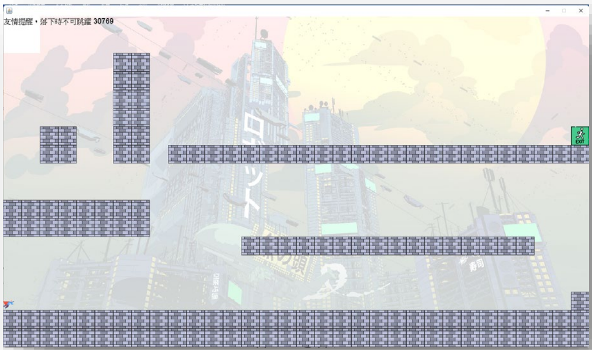

友情提醒，落下時不可跳躍 30769

圖 8

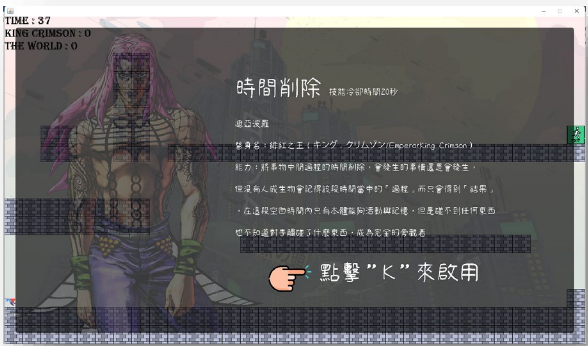

TIME : 37
KING CRIMSON : 0
THE WORLD : 0
時間削除 技能冷卻時間20秒
迪亞波羅
替身名：緋紅之王（キング．クリムゾン/EmperorKing Crimson）
能力：將事物中間過程的時間削除，會發生的事情還是會發生，
但沒有人或生物會記得該段時間當中的「過程」而只會得到「結果」
，在這段空白時間內只有本體能夠活動與記憶，但是碰不到任何東西
也不知道對手觸碰了什麼東西，成為完全的旁觀者
點擊"K"來啟用

圖 9

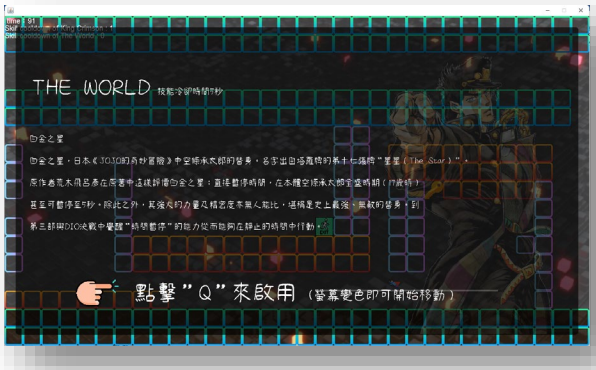


圖 10

十三、　時間判定與影片音效

在測試的時候發現了一個問題，就是播放影片的需要加載大量的 jpg，因此在第一次播放影片的時候會因為計算量大而速度變慢，第二次播放的時候會因為計算量小而速度變快，這會使觀影體驗大打折扣，因為影片內容跟聲音完全對不上。突然想到如過再創造一個精準的線程，只用來計時跟修改數字沒有其他的負擔，主線程在拿來讀取數字，或許就可以解決我的問題了，上網尋找了一下，還真的有為精準計時所生的線程，那這部分就照搬修改一下拿來用了（附錄 2），拜他所賜，遊戲內的所有時間都是準確的，只有毫秒級別的誤差。

十四、　除錯

最後的部分就是除錯了，除了自己之外，玩過測試版的同學也會發現一些 bug，有同學覺得不習慣用 w 鍵來跳躍，我就把跳躍功能同步到空白鍵上，覺得陷阱不明顯，就把陷阱的圖像重新設計，在一天天討論的過程中，看到遊戲一步一步從陽春版慢慢升級，到最後變成現在的樣子，真的太感動了。

關於 bug 的部分相信還有不少，沒有一個程式是完美的，但已經不會出現卡牆或

是當機等等必須關掉重來的大漏洞了，那其餘的 bug 就當作是遊戲的不完美，或許哪天可以看到玩家利用 bug 來通關也說不定。

伍、 問題與討論

一、 遊戲部分

1. 遊戲設計初衷？

讓使用者可以沉浸在解迷和探索，同時摸索出自己的破關方法來挑戰紀錄，可以先從場地版來找尋解答，再透過簡單版來熟悉遊戲角色的移動，最後再挑戰困難版。

2. 遊戲的定位？

單機、不連網小遊戲，可以保存歷史記錄，遊玩時間應為半小時到一小時，沒有商店、道具和儲存，每一次載入遊戲都會刷新所有物件，陷阱位置也是固定的，沒有隨機性，所以設計的太簡單可能不耐玩，才會有地獄難度的困難版。

二、 程式部分

1. 物件導向

在一開始設計的時候並沒有建立明確的物件導向概念，角色和陷阱都是通過簡單的位置數據儲存，並沒有使用到物件導向，希望能在下一個遊戲中加入。

2. Array

網路上有人形容 array 是低階的資料儲存方法，沒辦法快速檢視資料，又有大小限制，每一次都要開泡泡迴圈令人頭大，下次會用更適合的方式儲存，至少用到 vector 跟 array list。

3. 資料存儲

在資料存儲的部分我使用了一個 class 來放所有靜態（static）的資料，可以隨時用來存取修改，不用一直傳入傳出迷失在這個 function 裡，不知道這樣做是否多此一舉，但至少目前來看是沒有大問題的，會想請教各位老師是否能替我釐清一下這樣設計的優點和缺點呢？

4. 物理引擎

在我接出 unity 之前還不知道有物理引擎的存在，所以就用自己的想像力去設計碰撞，地面判定，移動速度，跳躍加速度等等的，很多部分比起現成的物理引擎還要遜色不少，那正因為知道了這件事，下一個作品就不用花時間在這方面的撰寫上了，直接借用 unity 的物理引擎就好了。

三、 介面部分

1. 如何優化使用者體驗？

設計 UI/UX，除了自己有真實玩過，體驗過測試版遊戲的同學、朋友們也提供了很大的幫助，每個人在遊玩時都會注意到不同的點，而我再去依據問題的大小來判斷是否要砍掉重新設計，還是加個提示或改流程，在開發的同時也常常詢問自己想要打造的遊戲是什麼風格的，玩家會用怎麼樣的心情走在這片地圖上以及想要從遊戲中獲得什麼內容，是成就感，是沉浸體驗（Flow Experience），是踏破所有地圖和所有陷阱的完美主義者，抑或是 BUG 追尋者。種種不同的遊玩模式帶給我不同的反饋，再根據每個人的意見去作微調，創造出一個最合適的遊戲。

2. 美術部分

A. 圖像設計

我對於繪畫部分非常不擅長，但只論設計的話都還算可以端出一點東西，遊戲內的大部分元件都是由小型的 PNG 構成，到網路抓取合適的方塊再拼接而成，大型物件都是自己設計，只做到了還可以看的程度，如果接下來還有其他企劃的話，會想練習繪畫，繪製自己的主角，一筆一畫都由自己生成，豈不酷哉。

B. 影像部分

相對於平面設計，我對製作影片或是動畫等就十分拿手，開頭的動畫用 PR 製成，大概花了一個小時去製作，在加上後面簡單微調一下就成功了，配上 YT 的現成無版權素材，結果讓我非常滿意。

技能組方面真是下了苦心啊，為了實現動畫中間能有透明部分，把圖片轉成了 PNG，但又為了避免檔案過於龐大，再把沒有透明部分的畫面改回 JPG，前前後後花了很多時間，檔案全得重新命名，其中察覺網路上的影片轉圖片會自動合併靜幀，只能偷偷加一個不起眼的動畫，讓每幀都有細微的差異，才能讓轉檔順利。下一步也是難題，想讓動畫順暢又可以精準的合上聲音是需要大量的微調，而不同步的問題也是因為用上了新的計時線程才得以解決，說是奇蹟似的走到這一步也不為過。

C. 音樂部分

雖然音樂部分是我的專長（編曲、錄音、鋼琴、即興），但這次就不多花時間在上面了，直接上 YT 下載無版權音樂來用，但在聲音的部分上還是有盡量做到卡點和優化，但遺憾礙於時間緊迫，沒有設置音量調節，無法控制聲音大小或靜音，這是我的疏失。

陸、 未來展望

在製作本遊戲之前，我沒有任何的遊戲開發經驗，連 GUI 也都只有做過上課的簡

單練習，我覺得經過這次的磨練學到很多，遇到的很多問題也都用滿腔的熱血順利解決，所以接下來我會學習 c#、unity、blander 來製作下一款遊戲，有著更多更複雜的內容，美術風格也會再度升級，也會稍微改變一下遊戲的設計初衷與風格，有可能是開放世界，有可能是刺激的動作遊戲，也有可能是有教育意義的益智遊戲。

柒、 心得

為什麼我當初選擇的是遊戲而不是中規中矩的應用呢？小時候總會被禁止玩遊戲，認為遊戲是無意義的、浪費時間的和花錢的…等等等不對，有人會花錢在上面？有消費就有商機，有商機的話，那就代表如果去設計遊戲的話就可以賺錢！（如果有人買的話）在這裡我要告訴大家，大部分遊戲確實沒有意義，但現在的大數據時代，玩遊戲不僅是有意義的，背後隱藏的更是一個大型市場，一個流行，不管今天是不是職業選手，是不是遊戲開發商，就算是個人也可以分一杯羹（帳號買賣、實況直播、二創產品的製作…），所以就算我不是一個遊戲狂熱份子，我也必須抨擊那些瞧不起遊戲的人。

思考了許久，在遊戲與程式的選擇上我有更進步的看法，相較於應用，遊戲可以讓使用者檢查到近乎所有功能，很多可能藏有瑕疵的部分會一再地重複操作或放大檢視，這讓我們可以很好的去找到錯誤並重新設計，但一般應用…就算是檢查人員也不可能把裡面所有功能全部 run 過，然後反覆測試個一萬次來看看會不會出問題（當然也可以設計檢查的 AI），而且我覺得最大的重點是自己，怎麼可能會有人想要認真的去測試一個華氏攝氏溫度轉換系統而不是一個有趣小遊戲，可能還真的有，但遊戲對於初學者或學生的吸引力絕對更大，而且更能找到漏洞。同時在測試遊戲的過程中還能融入個人的風格，成為獨一無二的遊戲，這可不是一舉兩得嗎?

好的該來說正事了，我喜歡設計（雖然不擅長畫畫），但鋼琴編曲，影片都是能讓我沉浸在其中且享受的愛好，同時也是專長。第一次接觸程式的環境並不太理想，雖然我很想抱怨，但我還是把過錯歸到自己的不上進，第二次見到程式就是這堂 java 了，對於基礎薄弱的我來說，每堂課都是大量的知識傳遞，不懂的我經常上網看影片邊看邊學，直到我打出第一個 hello world，因為程式而感動，更堅定了我自學程式的心。

在遊戲設計的過程中慢慢發現自己對於程式還蠻拿手的，尤其是腦海中常常湧出的新奇想法往往都能被保留在程式的某處，這對我來說是非常非常的…brilliant。在遊戲開發中遇到的每一個問題都是很好的挑戰，能夠讓我靜下心來好好應對，好好的去思考解決辦法，也增加了我對於整體的瞭解，往往在思考一個問題的時候就在心中列出了遊戲的下一個改進方向，一次又一次的開發經驗和成果累加也帶給我巨大的成就感，後續遇到龐大的問題也都咬牙苦撐過去，最後同學看到成品的一臉吃驚和通關時滿意的表情都讓我非常感動（還有被陷阱扎到的扭曲表情）。

我覺得這個遊戲雖然還有很多 bug 沒有解決，也還有很多可以擴充的東西，但我應該到此為止了，我會把它當作完成品，作為下一個作品的基石。

Ps：關於遊戲的命名，原本是以大寫 A 為命名的，但在撰寫過程中出現太多的錯誤，無法修改也找不到回溯點，所以毅然決然的複製一部份到新的專案裡重新編寫，因此才叫 project B。

捌、 附錄

1. 遊戲參考 https://www.youtube.com/watch?v=widYKCMVA7M
2. 第二線呈的借用 https://m.tqwba.com/x_d/jishu/305367.html
3. 我的程式 github （新手，沒有設定好請見諒）
   https://github.com/proadress/20220630-project-B-java-.git
4. Google 雲端檔案（非永久）
   https://drive.google.com/file/d/1Il2Re3su5X43Gd3LRJcVNz5yqywC-KJI/view?usp=sharing