程式設計期末報告

Maple Crush

第一組

b02704016 謝宗翰

b02704011 江靖儀

b02704090 鄭棋泰

b05704007 賴昱瑋

題材介紹

楓之谷是我們組別小時候的共同回憶，所以我們希望能將楓之谷的元素融入這次專案當中。因此，我們將楓之谷與 Candy Crush 的規則結合，也就是移動方塊，如果有三個以上相同圖形相連才會被消除，並且加入計分、倒數計時系統，最後形成我們的 Maple Crush。

系統設計與演算法

1. 函式庫

我們在考慮了 Qt 跟 SFML 後，還是決定使用 SFML，原因有以下兩點：

(1) Qt 需要在另外安裝自己的 IDE，但 SFML 是 portable 的，因此所有組員的電腦環境設定較能簡單統一。

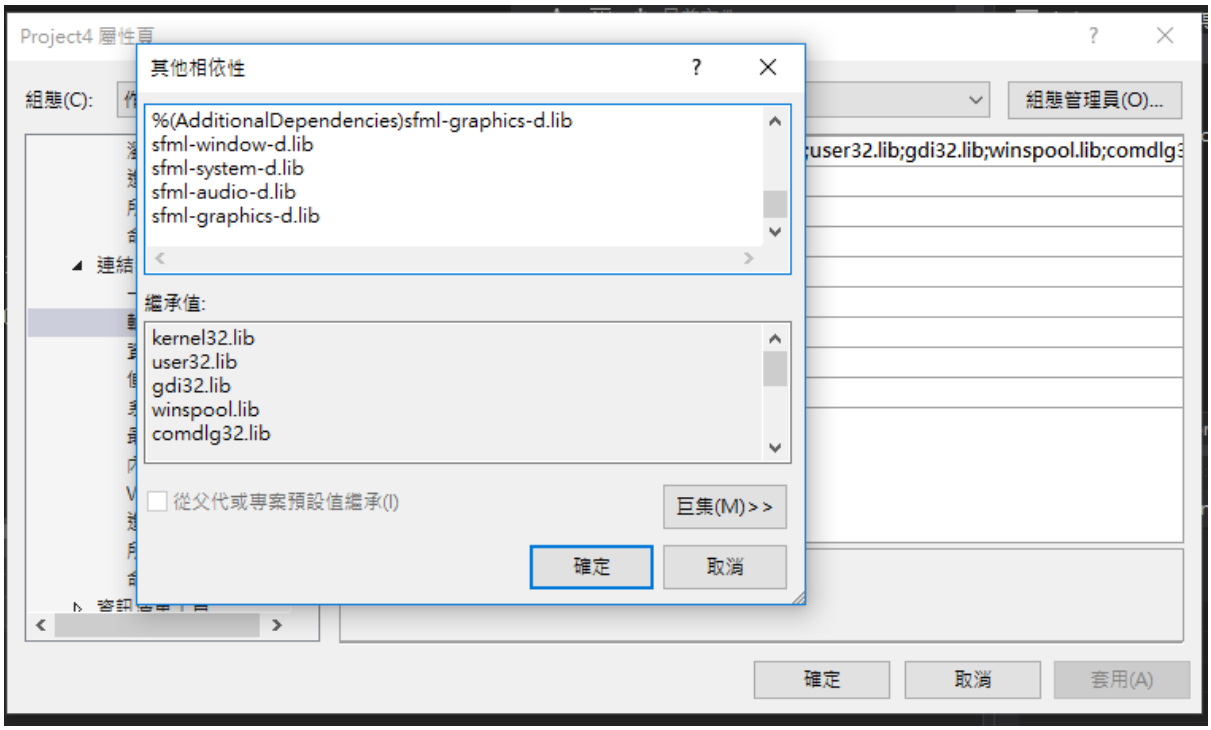
(2) SFML 內建的函數雖然比 Qt 少，但卻比較直觀、簡單快速，而且 SFML 對我們設計的遊戲來說也足夠使用，因此我們最後決定使用 SFML。

2. 功能

除了「消除怪物」的功能之外，為了增加遊戲趣味性與挑戰性，我們還新增了數項功能，包含：背景音樂、音效、動畫、計時器、計分板、歷史紀錄、停止畫面，以下就分點進行功能介紹。

a. 背景音樂及音效

為了提高遊戲完整性，除了加入楓之谷的背景圖片及使用「怪物」來取代寶石之外，我們也加入遊戲的背景音樂來提升「楓之谷」臨場感。另外，在消除怪物時，也搭配音效讓使用者體驗更完善。這裡所使用的是 sfml 中的 music 來播放音樂。



首先，要先將 sgml-audio-d.lib 加至 library 裡面。（我們一開始也在這上面花了很長一段時間 debug ，因為之前都沒有引入一個全新的函式庫的經驗）

```
sf::Music music;

if (!music.openFromFile("toytown.ogg"))
      return -1; // error
music.setVolume(50);
music.setLoop(true);
music.play();
```

值得注意的是，在 C++ 裡面，是不接受 .wmv 或 .mp3 這樣的檔案，要先轉檔成 .ogg 檔，才可以在程式裏面播放。

```
if (tem_cnt != 0 && move_time != 0)
{
      new_combos++;
      matched.setPitch(origin_pitch);
      matched.play();
      if (new_combos >= 2)
      {
            matched.setPitch(origin_pitch + 1);
            matched.play();
      }
      get_score = get_score + tem_cnt*new_combos;
}
```

而在音效方面，我們設計讓 combo 時的音效和原先音效不同，這裡需先儲存原音樂的 pitch，在 combo 時讓 pitch +1。

b. 動畫

在動畫的方面，SFML 呈現的方式是利用迴圈一格畫面一格畫面下去跑，因次在這邊我們將滑鼠點擊後的格子交換距離，利用迴圈分割成許多次，除了能讓動畫看起來變順暢之外，還可以透過調整迴圈跑的數量來加速動畫。

此外，因為無時無刻都在跑迴圈以顯示動畫與檢查消除規則，我們特別為這個情況建立了一個 Flag 名為 isMoving，以區分現在是否在動畫的階段，還是已經是動畫結束的正常情況，這樣才有辦法計算例如 combo 等變數。

c. 計時器

預期計時器中預期呈現的是一個指定時間，以秒數作為單位，隨現實時間流動而遞減，並在時間歸零時終止原程式的函式物件，在實作上選擇以 c 語言內建的 time.h 進行設計。

```
unsigned int x_seconds = 0;
unsigned int x_milliseconds = 0;
unsigned int total_time = 0, count_down = 0, time_left = 0;
clock_t start_time, count_time;
count_down = 60;
start_time = clock(); // record beginning time
time_left = count_down;
```

首先在程式運行的起點，以 clock() 函數取得當前的時間，即 start_time。此時間物件將被以 clock_t 的型態儲存，此類別的資料將能夠實現時間差的運算。在每次遊戲主體的迴圈中，將以 clock() 取得當次循環的時間，即 count_time。兩者相減，將會得到一個時間差，即「已經經過的時間」，與最初設立的倒數基準之差即為「剩餘時間」，即 time_left。

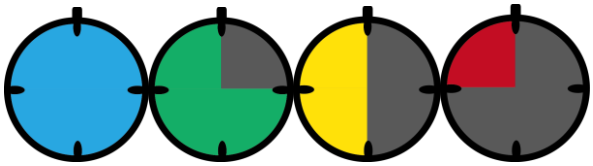
```
while (app.isOpen() && time_left > 0)
{

    // Count-down timer - 2
    count_time = clock();
    x_milliseconds = count_time - start_time;
    x_seconds = x_milliseconds / (CLOCKS_PER_SEC);
    time_left = count_down - x_seconds;

    std::string s;
    std::stringstream out;
    out << time_left;

    text.setString(out.str());
```

然而當時間彼此進行相減時，被計算出來的將不會是預期的「秒」，將會是一個數毫秒為單位的 clock ，因此必須轉換。 CLOCKS_PER_SEC 為一個常數，表示在此硬體中 clock 與秒的相對量，因此將原始計算出的時間差除以此數，即可得到秒數，進一步的獲取每個循環中的 「剩餘秒數」。



以新的秒數數字為元件，將每個循環所剩餘的秒數輸出為字串，即可用 SFML 的內建顯示文字功能呈現。此外，time_left 也將可以作為其他判斷之用，例如顯示動畫鐘的依據，以及是否繼續遊戲等。

d. 計分板

利用產生新的方塊處，建立一個暫時的計數變數 tem_cnt 計算該次產生的新磚塊數，此外，在迴圈之外加上 combo 計算，當進行一次的消除即加一，當沒有再掉落新磚塊時，則歸回零。

```
int tem_cnt = 0;
//產生新的補空格
for (int j = 1; j <= 8; j++){
      for (int i = 8, n = 0; i > 0; i--){
            if (grid[i][j].match){
                  //產生新磚塊的計算
                  tem_cnt++;
            }
      }
}
if (tem_cnt != 0 && move_time != 0){
      new_combos++;
      get_score = get_score + tem_cnt*new_combos;
      //在這裡將分數乘以 combo 數後，加上原來的分數繼續累加
}
else if (tem_cnt == 0 && move_time != 0) //如果沒有再產生磚塊，則讓 combo 歸零
      new_combos = 0;
```

而輸出計分板方面，依樣是利用 SFML 的視窗顯示 library。

```
//計分板
Text record_board;  //輸出需要做型態轉換，將分數記錄成 string
sprintf(output_score, "%d", get_score);
record_board.setString(output_score);
//設定字型
sf::Font font;
if (!font.loadFromFile("digital-7.ttf")){
      return -1;
}
//設定輸出樣式
record_board.setString(output_score);
record_board.setFont(font);
//…...如上面的方式引用 class 的函數，設定字形、大小、顏色、畫面位置
app.draw(record_board); //記分板顯示
```

e. 歷史紀錄

```
ifstream file("score.txt");
int historyscore;

if (file)
    file >> historyscore;

file.close();
```

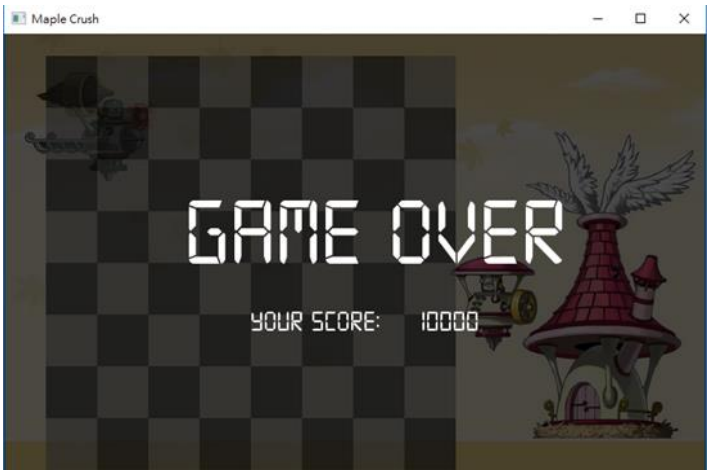
```
int new_score = get_score;

if (new_score > historyscore)
{
    ofstream file2;
    file2.open("score.txt");
    file2 << new_score;
    file2.close();
}
```

使用 <fstream> 進行建構。首先在資料夾內建立內容為 0 的 txt 純文字檔。在開始時，讀入此檔案並將其轉為數字，顯示於畫面。在計分結束後，與此次分數進行比較。若分數刷新，則重新以寫入模式開啟檔案，覆蓋原始數字後關閉。由於歷史最高記錄數字原則上為單向成長，因此不必考慮其他字串複寫問題，如數字長度等。

f. 停止畫面

當時間歸零的瞬間，將會跳出第一個遊戲循環，進入第二個 while 迴圈。此迴圈將以 SFML 內建的插入圖片功能，長時間顯示帶有深色遮罩的背景，並顯示 gameover 字樣與此次獲得分數。在此區段的遊戲系統中，並無建立任何獲取鍵盤或滑鼠指令的函數，為一的指令為在關閉視窗的同時結束此應用程式。



分工方式

我們組別的分工滿平均的，所有的工作階段（資料搜尋、建立初步 pseudocode、分工撰寫主程式、presentation 簡報以及最後的書面報告）幾乎四個人都有全程參與討論並且確實分工合力完成。

而較詳細的主程式分工，如果以人來說的話列表如下：

鄭棋泰：背景音樂及音效、動畫、計時器、歷史紀錄、停止畫面、Demo

賴昱瑋：動畫、消除規則與計分方式、計分板

江靖儀：背景音樂及音效、美工圖片、口頭報告及簡報製作

謝宗翰：美工圖片、動畫、消除規則與計分方式、Demo

心得感想

謝宗翰

做完期末報告覺得又一次突破自己的界線了，完全沒想過我們一群人，能在老師完全沒有教 GUI 的情況下，透過大量上網找資料跟討論，進而把一個我們都不知道是什麼的領域做出了一些有趣的東西。這次專案學到的東西也很多，除了證明自己能夠從無到有搞懂一個函式庫外，還磨練了跟別人合作的技巧，完全證明了老師上課所說的模組化有多重要。雖然聽完其他人的專案，覺得實在被電得很慘，但是回頭看看自己寫出來的小遊戲，還是覺得非常驕傲，尤其在經歷了各種撞牆之後，顯得更值得。

賴昱瑋

這次的期末報告我認為有頗多的收穫，在學期之初選了這門課，從來沒有預期竟然能做出這樣的程式，原來基本的概念學會了以後，確實要接觸更深入的東西有比較容易。我們做輸出的 library 是 SFML，我們一開始本想用 QT，但遇上了滿多的障礙(或許暑假可以進一步試試看這個)，無意間看見了 SFML 感覺上親近了許多，語法形式是更熟悉的樣子，但又面臨了作業系統的問題，團體寫程式真的不是很容易的，除了溝通之外還有電腦也會有些不同的狀況，還有模組化真的不是件容易的事，希望未來雖然不是主修資管，但還希望能多多接觸這方面的知識，然後自己有了更多學習 programming 的動力。

鄭棋泰

期中報告在各種因素的影響之下可說是憑一己之力完成，因此對於個人而言這個期末報告相對容易，且自己對於如何從程式面製作互動式的介面也非常有興趣。自己從小到大也接觸了不少遊戲，但親自動手才知道每一個畫面窗格和每一個動態效果都有非常多限制以及需要分毫不差的注視事項，光是從主遊戲切入另一個靜態畫面的作法就研究了許久。常見的遊戲軟體都至少由數十個不同的場景所構成，又各有不同條件下的互動效果，稍微具備程式的知識後就能夠了解這個架構龐大的程度，也不難想像在建構與維護上的困難。雖然當前的知識可能只深及皮毛，但藉此專案感覺自己已經可以窺見那些建構而成的世界的一角。

江靖儀

期末報告和期中報告相較起來有趣許多，但在一開始時，確實覺得很困難，因為從來沒有用 C++ 寫過遊戲，或寫過任何不是用 cmd 直接輸入輸出的東西。因此，一開始我們非常摸不著頭緒，光是要找到一個可用的函式庫都非常困難。還好後來看到了一些用 sfml 寫的範例遊戲，才讓我們的期末專案柳暗花明。雖然我們也是盡心盡力的想要做出最好的專案，也的確花了很多時間跟心力在這上面，但當我們看到有一組做出超強的音樂遊戲 RPG 時，真的是相形見絀，事實上，我從來沒想過修完這門課，我自己可以寫出這樣的東西。感覺他們寫出來的遊戲，與我們的遊戲已經是不同的層級了，當下除了很佩服之外，也覺得非常受到啟發，發現自己其實是有可能寫出那樣的遊戲的。但總之，我覺得我很享受這次寫遊戲的過程，也從中學到了程式設計活潑的一面。