目錄

一、 簡介

1. 動機

最一開始我們想了許多遊戲，像是越南大戰、馬力歐兄弟……等等，但因太普遍且學長們都做過了，所以我們決定改變方向，至於為什麼選擇跑跑薑餅人當主題呢。當初在想一款比較少人知道的遊戲時，我們剛好想到可以把手機遊戲做成電腦版的，於是就在手機上找尋適合的遊戲，最後在千挑萬選後，選擇了跑跑薑餅人，原因除了這款遊戲的操作比較好做成電腦版，最主要的還是我玩過這款遊戲，也玩了一段時間，所以我自認我對這款遊戲的掌握度很高，而在製作成遊戲時也會比較好上手。

2. 分工

陳思齊、謝宗翰：美術、音樂、圖形、程式，各個方面都有參與。我們的方工方式是把星期一上課訂的目標，經過討論後分配好，這樣在各個方面兩人都有參與到，也不會有人完全不會某個部分。

二、 遊戲介紹

1. 遊戲說明

a. 遊戲背景

可愛的薑餅人被送入烤箱了，玩家必須操控薑餅人躲避各種障礙物，並通過吃到各種道具和果實來到達終點，幫助薑餅人逃出烤箱!!!

b. 遊戲玩法

一共有兩個關卡，選擇好關卡後馬上就開始，玩家使用鍵盤上、下鍵來操控薑餅人閃過各種障礙物，並得到果實和道具(詳細內容在 2. 遊戲圖形裡)，遊戲途中，撞到障礙物的時間愈長，扣的血量愈多，如果玩家掉到洞裡或血量歸零則遊戲結束，代表薑餅人被烤成餅乾了，如果成功通過兩關，則恭喜玩家成功幫助薑餅人逃出烤箱，通過遊戲。

2. 遊戲圖形

a. 人物

通過動畫來生動表現出薑餅人的動作，也經過精細的修改，讓玩家不會在動畫間感覺到違和或突兀感。

b. 地圖

選擇優美的背景來搭配逃出的緊張感，撞到礙障物時也通過感變色調來提醒玩家，因撞到障礙物而扣血。

c. 果實與道具

變大果實，可以使薑餅人在一定內巨大化，且無視障礙物。（密技為Q鍵）

衝刺果實，可以使薑餅人在一定內加速衝刺，且無視障礙物。(密技為 W 鍵)

磁鐵,可以使薑餅人在一定內吸取周圍的分數,包括其他果實或道具。(密技為 E 鍵)

回復果實，可以使薑餅人回覆一定生命的道具。(密技為 R 鍵)

BonusTime，可以使薑餅人飛上天空，並得到衝刺和磁鐵功能，在加分地圖裡快速獲取分數。(無密技)

一般的果實，可以使薑餅人獲得分數。(無密技)

d. 障礙物

前兩種為地面障礙物，第一種必須通過兩段跳來進行閃躲，第二種只需要一段跳就可以躲避；第三種為天空的障礙物，玩家必須通過蹲下來進行閃躲。我們也製作動畫來讓玩家因衝刺果實或變大果實而破壞障礙物時有實際的感覺。

e. 說明畫面

通過詳細的文字說明，搭配圖片讓玩家印象深刻，能馬上熟習操作且看到道具就馬上知道它的功能。

f. About

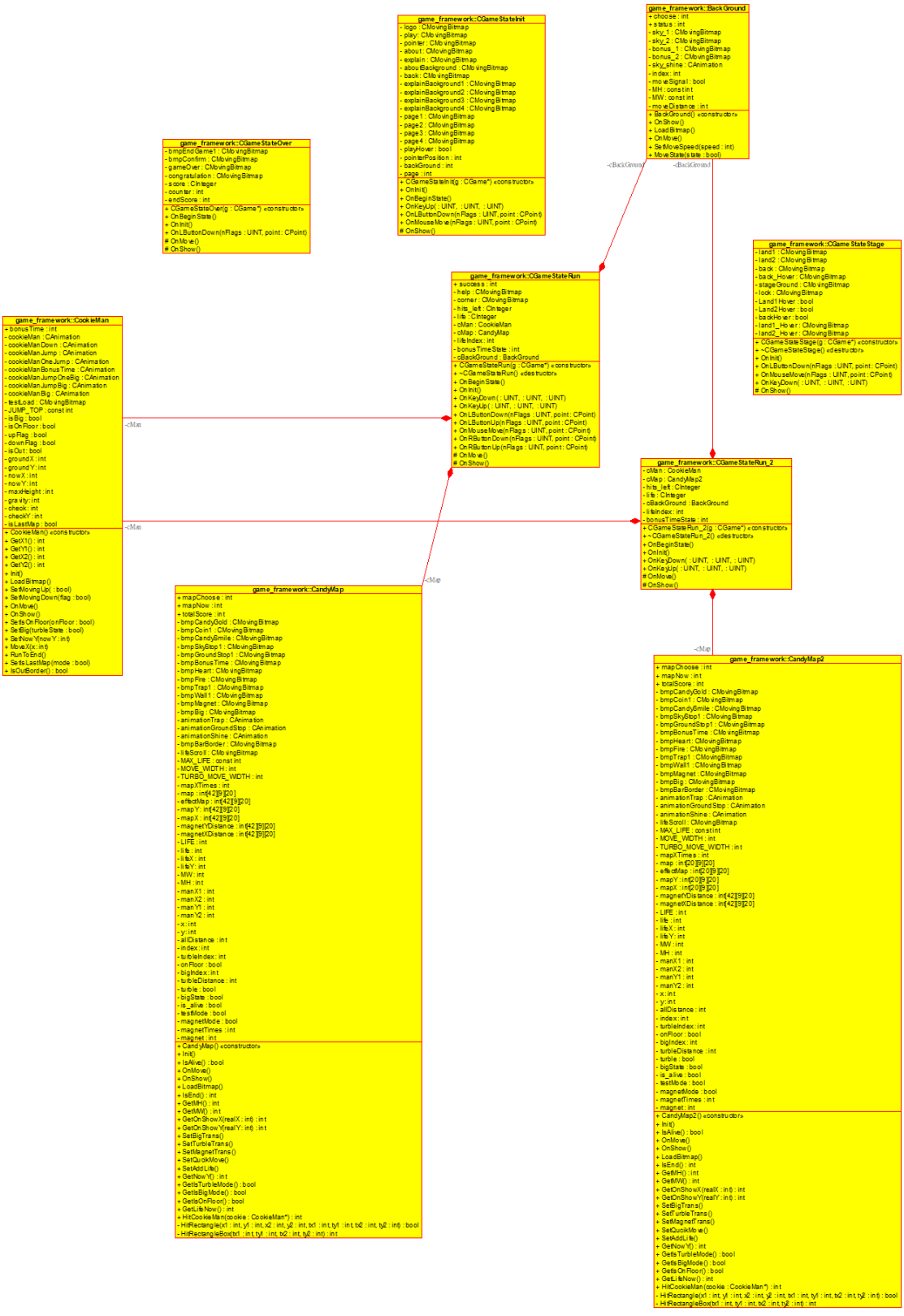
以白、黑色來襯托，讓玩家一眼可以看出小組成員和授課老師。

3. 遊戲音效

音效名稱	音效使用時機	音效說明
coin01	吃到分數時	「叮」的一聲，表示吃到分數
background2	開始進行關卡時	配合關卡放出 bgm，以不會讓玩家感到沉悶

三、 程式設計

1. 程式架構


game_framework::CGameStateInit
- logo : CMovingBitmap
- play: CMovingBitmap
- pointer : CMovingBitmap
- about : CMovingBitmap
- explain : CMovingBitmap
- aboutBackground : CMovingBitmap
- back: CMovingBitmap
- explainBackground1 : CMovingBitmap
- explainBackground2 : CMovingBitmap
- explainBackground3 : CMovingBitmap
- explainBackground4 : CMovingBitmap
- page 1 : CMovingBitmap
- page 2 : CMovingBitmap
- page 3 : CMovingBitmap
- page 4 : CMovingBitmap
- playHover : bool
- pointerPosition : int
- backGround : int
- page : int
+ CGameStateInit(g : CGame*) «constructor»
+ OnInit()
+ OnBeginState()
+ OnKeyUp(: UINT, : UINT, : UINT)
+ OnLButtonDown(nFlags : UINT, point : CPoint)
+ OnMouseMove(nFlags : UINT, point : CPoint)
# OnShow()
game_framework::BackGround
+ choose : int
+ status : int
- sky_1 : CMovingBitmap
- sky_2 : CMovingBitmap
- bonus_1 : CMovingBitmap
- bonus_2 : CMovingBitmap
- sky_shine : CAnimation
- index: int
- moveSignal : bool
- MH : const int
- MW: const int
- moveDistance : int
+ BackGround() «constructor»
+ OnShow()
+ LoadBitmap()
+ OnMove()
+ SetMoveSpeed(speed : int)
+ MoveState(state : bool)
game_framework::CGameStateOver
- bmpEndGame1 : CMovingBitmap
- bmpConfirm : CMovingBitmap
- gameOver : CMovingBitmap
- congratulation : CMovingBitmap
- score : CInteger
- counter : int
- endScore : int
+ CGameStateOver(g : CGame*) «constructor»
+ OnBeginState()
+ OnInit()
+ OnLButtonDown(nFlags : UINT, point : CPoint)
# OnMove()
# OnShow()
game_framework::CGameStateRun
+ success : int
- help : CMovingBitmap
- corner : CMovingBitmap
- hits_left : CInteger
- life : CInteger
- cMan : CookieMan
- cMap : CandyMap
- lifeIndex : int
- bonusTimeState : int
- cBackGround : BackGround
+ CGameStateRun(g : CGame*) «constructor»
+ ~CGameStateRun() «destructor»
+ OnBeginState()
+ OnInit()
+ OnKeyDown(: UINT, : UINT, : UINT)
+ OnKeyUp(: UINT, : UINT, : UINT)
+ OnLButtonDown(nFlags : UINT, point : CPoint)
+ OnLButtonUp(nFlags : UINT, point : CPoint)
+ OnMouseMove(nFlags : UINT, point : CPoint)
+ OnRButtonDown(nFlags : UINT, point : CPoint)
+ OnRButtonUp(nFlags : UINT, point : CPoint)
# OnMove()
# OnShow()
game_framework::CGame StateStage
- land1 : CMovingBitmap
- land2 : CMovingBitmap
- back : CMovingBitmap
- back_Hover : CMovingBitmap
- stageGround : CMovingBitmap
- lock : CMovingBitmap
- Land1Hover : bool
- Land2Hover : bool
- backHover : bool
- land1_Hover : CMovingBitmap
- land2_Hover : CMovingBitmap
+ CGameStateStage(g : CGame*) «constructor»
+ ~CGameStateStage() «destructor»
+ OnInit()
+ OnLButtonDown(nFlags : UINT, point : CPoint)
+ OnMouseMove(nFlags : UINT, point : CPoint)
+ OnKeyDown(: UINT, : UINT, : UINT)
# OnShow()
game_framework::CookieMan
+ bonusTime : int
- cookieMan : CAnimation
- cookieManDown : CAnimation
- cookieManJump : CAnimation
- cookieManOneJump : CAnimation
- cookieManBonusTime : CAnimation
- cookieManJumpOneBig : CAnimation
- cookieManJumpBig : CAnimation
- cookieManBig : CAnimation
- testLoad : CMovingBitmap
- JUMP_TOP : const int
- isBig : bool
- isOnFloor : bool
- upFlag : bool
- downFlag : bool
- isOut : bool
- groundX : int
- groundY : int
- nowX : int
- nowY: int
- maxHeight : int
- gravity : int
- check : int
- checkY : int
- isLastMap : bool
+ CookieMan() «constructor»
+ GetX1() : int
+ GetY1() : int
+ GetX2() : int
+ GetY2() : int
+ Init()
+ LoadBitmap()
+ SetMovingUp(: bool)
+ SetMovingDown(flag : bool)
+ OnMove()
+ OnShow()
+ SetIsOnFloor(onFloor : bool)
+ SetBig(turbleState : bool)
+ SetNowY(nowY : int)
+ MoveX(x : int)
+ RunToEnd()
+ SetIsLastMap(mode : bool)
+ IsOutBorder() : bool
game_framework::CGameStateRun_2
- cMan : CookieMan
- cMap : CandyMap2
- hits_left : CInteger
- life : CInteger
- cBackGround : BackGround
- lifeIndex : int
- bonusTimeState : int
+ CGameStateRun_2(g : CGame*) «constructor»
+ ~CGameStateRun_2() «destructor»
+ OnBeginState()
+ OnInit()
+ OnKeyDown(: UINT, : UINT, : UINT)
+ OnKeyUp(: UINT, : UINT, : UINT)
# OnMove()
# OnShow()
game_framework::CandyMap
+ mapChoose : int
+ mapNow : int
+ totalScore : int
- bmpCandyGold : CMovingBitmap
- bmpCoin1 : CMovingBitmap
- bmpCandySmile : CMovingBitmap
- bmpSkyStop1 : CMovingBitmap
- bmpGroundStop1 : CMovingBitmap
- bmpBonusTime : CMovingBitmap
- bmpHeart : CMovingBitmap
- bmpFire : CMovingBitmap
- bmpTrap1 : CMovingBitmap
- bmpWall1 : CMovingBitmap
- bmpMagnet : CMovingBitmap
- bmpBig : CMovingBitmap
- animationTrap : CAnimation
- animationGroundStop : CAnimation
- animationShine : CAnimation
- bmpBarBorder : CMovingBitmap
- lifeScroll : CMovingBitmap
- MAX_LIFE : const int
- MOVE_WIDTH : int
- TURBO_MOVE_WIDTH : int
- mapXTimes : int
- map : int[42][9][20]
- effectMap : int[42][9][20]
- mapY: int[42][9][20]
- mapX : int[42][9][20]
- magnetYDistance : int[42][9][20]
- magnetXDistance : int[42][9][20]
- LIFE : int
- life : int
- lifeX : int
- lifeY: int
- MW : int
- MH : int
- manX1 : int
- manX2 : int
- manY1 : int
- manY2 : int
- x: int
- y: int
- allDistance : int
- index: int
- turbleIndex: int
- onFloor : bool
- bigIndex: int
- turbleDistance : int
- turble : bool
- bigState : bool
- is_alive : bool
- testMode : bool
- magnetMode : bool
- magnetTimes : int
- magnet : int
+ CandyMap() «constructor»
+ Init()
+ IsAlive() : bool
+ OnMove()
+ OnShow()
+ LoadBitmap()
+ IsEnd() : int
+ GetMH() : int
+ GetMW() : int
+ GetOnShowX(realX : int) : int
+ GetOnShowY(realY : int) : int
+ SetBigTrans()
+ SetTurbleTrans()
+ SetMagnetTrans()
+ SetQuickMove()
+ SetAddLife()
+ GetNowY() : int
+ GetIsTurbleMode() : bool
+ GetIsBigMode() : bool
+ GetIsOnFloor() : bool
+ GetLifeNow() : int
+ HitCookieMan(cookie : CookieMan*) : int
- HitRectangle(x1 : int, y1 : int, x2 : int, y2 : int, tx1 : int, ty1 : int, tx2 : int, ty2 : int) : bool
- HitRectangleBox(tx1 : int, ty1 : int, tx2 : int, ty2 : int) : int
game_framework::CandyMap2
+ mapChoose : int
+ mapNow : int
+ totalScore : int
- bmpCandyGold : CMovingBitmap
- bmpCoin1 : CMovingBitmap
- bmpCandySmile : CMovingBitmap
- bmpSkyStop1 : CMovingBitmap
- bmpGroundStop1 : CMovingBitmap
- bmpBonusTime : CMovingBitmap
- bmpHeart : CMovingBitmap
- bmpFire : CMovingBitmap
- bmpTrap1 : CMovingBitmap
- bmpWall1 : CMovingBitmap
- bmpMagnet : CMovingBitmap
- bmpBig : CMovingBitmap
- bmpBarBorder : CMovingBitmap
- animationTrap : CAnimation
- animationGroundStop : CAnimation
- animationShine : CAnimation
- lifeScroll : CMovingBitmap
- MAX_LIFE : const int
- MOVE_WIDTH : int
- TURBO_MOVE_WIDTH : int
- mapXTimes : int
- map : int[20][9][20]
- effectMap : int[20][9][20]
- mapY: int[20][9][20]
- mapX : int[20][9][20]
- magnetYDistance : int[42][9][20]
- magnetXDistance : int[42][9][20]
- LIFE : int
- life : int
- lifeX : int
- lifeY: int
- MW : int
- MH : int
- manX1 : int
- manX2 : int
- manY1 : int
- manY2 : int
- x: int
- y: int
- allDistance : int
- index: int
- turbleIndex: int
- onFloor : bool
- bigIndex: int
- turbleDistance : int
- turble : bool
- bigState : bool
- is_alive : bool
- testMode : bool
- magnetMode : bool
- magnetTimes : int
- magnet : int
+ CandyMap2() «constructor»
+ Init()
+ IsAlive() : bool
+ OnMove()
+ OnShow()
+ LoadBitmap()
+ IsEnd() : int
+ GetMH() : int
+ GetMW() : int
+ GetOnShowX(realX : int) : int
+ GetOnShowY(realY : int) : int
+ SetBigTrans()
+ SetTurbleTrans()
+ SetMagnetTrans()
+ SetQuickMove()
+ SetAddLife()
+ GetNowY() : int
+ GetIsTurbleMode() : bool
+ GetIsBigMode() : bool
+ GetIsOnFloor() : bool
+ GetLifeNow() : int
+ HitCookieMan(cookie : CookieMan*) : int
- HitRectangle(x1 : int, y1 : int, x2 : int, y2 : int, tx1 : int, ty1 : int, tx2 : int, ty2 : int) : bool
- HitRectangleBox(tx1 : int, ty1 : int, tx2 : int, ty2 : int) : int
-cBackGround
-cBackGround
-cMan
-cMan
-cMap
-cMap

2. 程式類別

類別名稱	.h 檔行數	.cpp 檔行數	說明
mygame	188	790	遊戲操作流程框架執行
CookieMan	48	287	針對人物的移動/特殊模式的人物動畫改變
CandyMap1	68	1141	針對地圖 1 的所有地圖物件做偵測顯示，以及物件的動畫
CandyMap2	68	881	針對地圖 2 的所有地圖物件做偵測顯示，以及物件的動畫
Background	23	95	判斷執行遊戲內應該使用的地圖
總行數	335	2096	

四、 結語

1. 問題及解決方法

   a. 在一開始時，因為過了個暑假而有點忘記程式一些基本的東西，如：繼承、指標、多型。

      SOL: 在第一周時上網或請教同學快速地複習，避免開始做遊戲時，寫得不順或動不動就卡住。

   b. 在建置地圖時，不能成功的建成地圖

      SOL :通過詢問老師和看老師網頁上的講義來了解地圖是如何建成的，並實際建置一個大陣列來存放地圖。

   c. 人物不能順利踩上地圖上較高的地板，會直接穿過去。

      SOL :原因是發現我們一開始沒有把人物真正的讓他踩下地板上，才會導致這個問題，通過詢問同學的想法後，成功讓人物踩在地板上。

   d. 卷軸地圖和人物座標之間無法取得對方位址。

      SOL: 將原本獨立計算移動距離的方式換成以卷軸地圖為主，當人物有座標需求時，用地圖座標去換算，同時減少負擔和誤差。

   e. 由於使用大量地圖物件(各種糖果、各式障礙物)，因此沒有各別創建物件 class 來操作顯示，直接將物件貼在視窗上，但卻造成這些物件無法被指定操作。

      SOL: 另外創建一個 MapArray，存取 MapArray 裏頭的功能/特殊狀態。

   f. 銜接問題 e，製作磁鐵模式時，由於地圖上的物件都是用地圖 Map 裡矩陣的物品(他們用 switch case 顯示)，無法獨立給他們磁鐵吸引的 Method，所以不知道如何下手操作。

SOL: 吸引需牽扯到上一個狀態的位置和下一個狀態的位置，因此創建一個 Array 紀錄舊位置，並將經過計算得到的下一次的顯示位置也存起來，以便 OnShow 可以使用。

g. 銜接問題 f，磁鐵模式下無法讓每個物件吸引到人物身上

SOL: 藉由上面說的儲存資料，算出人物座標和每個物件的距離差用數學公式算出 X,Y 分別移動的量，合併到 onShow 裡顯示。

h. 地圖本身>>>視窗大小，顯示時產生重疊/沒在預期的地方出現

SOL: 為了防止這種不自然的情況，我們每次更新 Map 是同時更新兩個，一個是視窗正在顯示的 Map，另一個則是預備準備要 show 出來的 map，當兩個隨著移動距離切換時，才能顯得自然。

i. 在 MyGame 裡的各種 GameState 之間需要傳值卻礙於不同 class 無法將分數傳到別的 Class

SOL: 我們用的方式不是最完美的，我們選擇在 MyGame 的 Cpp 裡使用全域變數，讓不同的 class 都可以同時存取，雖然明白這不是一個好的想法，但還找不到其他能解決的方法。

j. 原版遊戲是手遊，地圖素材資源在網路上十分稀少，多半為影片或是手機截圖，畫面使用起來十分粗糙。

SOL: 手機截的圖片出來大部分畫質都不好，再經過上傳到網路，多半被壓縮得十分嚴重，因此改用手機模擬器一張一張的手截，並把人物單獨裁剪出來，進行後製修圖。

k. 人物變大的蹲下無法準確的在指定位置

SOL :這源自於我們寫這人物操作時的沒有溝通好，兩個人同時針對人物新添共能時，寫法完全不同，因此變得複雜，交雜一些邏輯錯誤，進而產生的 bug，解決方法就是各種設中斷點回推問題找錯誤點。

l. 銜接問題 k，如果短時間內狂按蹲下鍵讓人物趴下，會意外觸發 bug，直接穿過地板，跌出地圖界線外。

SOL: 當按下蹲下鍵時人物會下降，當快速按蹲下鍵時人物還沒被返回原本的位置，又下降，讓人物逐漸超過地板能承受的位置，因此我們設定一個界線，小於一定值後會阻擋任何下降的移動，來暫時修復這個功能。

m. 地圖裡有些障礙物面積蠻大的，要把區域內的每個點都設定成為障礙物範圍的話，會使得產生很多障礙物交錯再一起。

SOL: 我們把每個將障礙物的點都先收集好，從中算出一個最適合的位置做顯示，其餘的都不顯示。

2. 時間表

陳思齊：

周次	陳思齊(小時)	說明
1	4.5	熟悉遊戲,糖果擺放程式，討論細節,做出會跳動人物
2	10	盡量完成第一章地圖畫面,卷軸可以"銜接"移動,人物碰撞果凍後果凍會消失。
3	9.5	完成第一章地圖和規劃第二章地圖
4	12	新增開始畫面和結束畫面，新增生命條和完成第一關。優化碰撞
5	9	完成選擇關卡和設定畫面，規劃第二關，完成角色動畫，完成衝刺。
6	7.5	角色巨大化功能，結束畫面可點選，美化角色動畫，討論第二關。

7	1.5	角色巨大化動畫，生命條動畫，優化動畫，設計磁鐵架構。
8	7	設計第二關功能: bonus time 功能，地板偵測
9	8.5	完成 bonusTime 動畫功能，區分二段跳動畫，turble 不見 ，撞障礙物扣血，bigState 碰撞後的彈飛動畫，掉入地板下會死掉
10	11	第二/三關地圖完成，修復天花板 bug，bonusTime 後變成飛的，磁鐵功能，加入音效，趴下的動畫，photoshop 白邊，顯示結算分數，成功畫面，裝飛障礙物會旋轉，新增變大&磁鐵圖示
11	2	美化圖片，變大後的動畫，修改地圖 bug
12	1	調整動畫 2.修正地圖小 bug
13	2.5	調整動畫，修正地圖 load 圖片的 bug，結束動畫
14	12.5	選關/人物/LifeBar/Sky Background/美觀 2.人物放大後的蹲下 bug 3.音樂可調整 4.遊戲 icon 5.遊戲 about 6.說明
15	9.5	遊戲 icon，調音樂大小，遊戲得分顯示，人物美化，書面報告。
合計	108	

謝宗翰：

周次	謝宗翰(小時)	說明
1	11	熟悉遊戲,糖果擺放程式，討論細節,做出會跳動人物
2	9.5	盡量完成第一章地圖畫面,卷軸可以"銜接"移動,人物碰撞果凍後果凍會消失。
3	11	完成第一章地圖和規劃第二章地圖
4	11	新增開始畫面和結束畫面，新增生命條和完成第一關。優化碰撞
5	9	完成選擇關卡和設定畫面，規劃第二關，完成角色動畫，完成衝刺。
6	4	角色巨大化功能，結束畫面可點選，美化角色動畫，討論第二關。
7	1	角色巨大化動畫，生命條動畫，優化動畫，設計磁鐵架構。
8	4.5	設計第二關功能: bonus time 功能,地板偵測
9	8.5	完成 bonusTime 動畫功能,區分二段跳動畫,turble 不見 ，撞障礙物扣血，bigState 碰撞後的彈飛動畫，掉入地板下會死掉
10	17	第二/三關地圖完成，修復天花板 bug，bonusTime 後變成飛的，磁鐵功能，加入音效，趴下的動畫，photoshop 白邊，顯示結算分數，成功畫面，裝飛障礙物會旋轉，新增變大&磁鐵圖示
11	1	美化圖片，變大後的動畫，修改地圖 bug

12	1	調整動畫 2.修正地圖小 bug
13	3.5	調整動畫，修正地圖 load 圖片的 bug，結束動畫
14	15	選關/人物/LifeBar/Sky Background/美觀 2.人物放大後的蹲下 bug 3.音樂可調整 4.遊戲 icon 5.遊戲 about 6.說明
15	18	遊戲 icon，調音樂大小，遊戲得分顯示，人物美化，書面報告。
合計	125	

3. 貢獻比例

陳思齊：50%

謝宗翰：50%

4. 自我檢核表

	項目	完成否	無法完成的原因
1	解決 Memory leak	已完成	
2	自訂遊戲 Icon	已完成	
3	全螢幕啟動	已完成	
4	有 About 畫面	已完成	
5	初始畫面說明按鍵及滑鼠之用法與密技	已完成	
6	上傳 setup/apk/source 檔	已完成	
7	setup 檔可正確執行	已完成	
8	報告字型、點數、對齊、行距、頁碼、等格式正確	已完成	
9	報告封面、側邊格式正確	已完成	
10	報告附錄程式格式正確	已完成	

1. 收穫

陳思齊：

把上學期的物件導向程式設計所學習的繼承、多型……等功能，在這邊進一步了解、應用，也因為程式的巨大化，要進行長時間的偵錯，進而學習到如何偵錯，為了讓下次更好看懂自己在寫什麼，而學習到註解的重要性，為了讓程式更方便的擴充功能，而學習到程式的維護性，這些都是上學期知道，但卻不怎麼做過的東西；也通過上網查詢或和同

學討論程式，來了解到他人對程式不同的見解，以讓自身可以更上一層樓。

謝宗翰：

經過這次的專題消化整理了大一大二的程式概念，並學習了框架的概念。用框架的結構，讓程式更多樣性，不一樣的變化。這次使用的是老師的自己寫的框架，一開始花了不少時間上手摸透框架的邏輯，間接學習了要如何閱讀程式碼,看到好的註解和程式架構會讓後續的維運、擴展更輕鬆。每週一就像是一個考核日，助教或老師會笑笑的走過來問說，這週進度怎麼樣，簡簡單單的一句話就可以使我們充滿壓力，有著時間限制/進度壓力讓我們更明白 Deadline 的重要性。獨立/邏輯思考是這堂課累積起來的軟技能，遇到問題回推問自己為什麼出錯了，哪出錯了，影響這個點的東西是什麼，幫助我釐清很多解決不了的問題。還有就是分享跟溝通，在跟同學聊天交流抱怨時，常常會收穫很多不同面向的知識，即使那些東西沒接觸過，也能知道一二。

2. 心得

陳思齊：

一開始聽完老師講解本課程主要的目的時,我心中感到非常不安,因為我的程式不是非常厲害,上學期程作業也都花非常久的時間才完成,突然就要做出一款遊戲，心中難免有些不安，但一周一周的過去，當然做出一個遊戲真的不簡單，但是過程我沒有想像中的不安、痛苦，每個禮拜都跟組員討論自己的遊戲進度，跟其他同學討論如何寫出這個、那個功能，成功寫出一個功能時，自己心中總有種成就感，而在修改圖片的部分覺得有些乏味，加上自己的審美有點問題，修好的圖總是不是很滿意，不過這個部分還是通過跟組員的合作，成功的完成了。整個學期

下來，我們成功的完成了一款有頭有尾的遊戲，雖然說不上是個完美的作品，但我每次玩著這個遊戲時，都讓我產生一種我可以、我做得到的感覺，希望我可以記住這份心情，並在未來不管是在做遊戲設計或是其他的軟體應用上都可以回想起自己當初的那份初衷。

謝宗翰：

很開心我完成了第一個/最後一個遊戲了，打破了說過不做遊戲的這種話。之前都沒有很多玩遊戲的經驗，二上一開始聽到有這門課時就有點緊張，雖然後來有點期待，不過更多的是害怕自己達不到這個目標。真的非常非常欣慰和開心能和我的組員完成這個專題。這算是第一次和別人一起協同寫程式，分擔了很多不同的功能實作。雖然每次看到組員完成一個功能時都十分開心，但過了幾週要加新的功能，去找到他寫的程式時，心裡就產生很多問號，想說這倒底是什麼幹嘛的。因為不瞭解對方的思考方式跟邏輯，有時寫的時候就會不小心誤解並埋下 bug。規劃和溝通真的蠻重要的，不僅僅是目標的討論，能在一開始就對程式做好架構框架圖，討論要用什麼方法實作，在後續的寫扣上能減少很多心力。因為第一次寫遊戲的程式，很多東西都是邊做邊想，埋下了許多技術債，後面幾週需要增添新功能時卻發現因為前面的程式框架寫的不好，導致得花更多 code 彌補前面沒想好導致的問題。寫遊戲的過程中常常會懷疑自己真的做得到嗎，看到別人的遊戲那麼厲害，心裏就覺得自己的遊戲怎麼就很普通，有時寫扣寫到半夜，為了把圖能夠弄的更好看、動畫更流暢，可以調個 1.2 個小時，走在路上突然想到 bug 怎麼解時，又覺得能一步步完成這個遊戲，已經盡力了，看著這遊戲從 0 到現在有個完整的架構，十分有成就感。這個程式是到目前為止寫過最大的程式，和組員一起弄出完整的專案，很滿足，一來一往的過程，提升對程式的敏銳度，知道完成一個專案時需要如何規劃，還有對於時間、進度的掌

控度。希望未來能夠以這個當起點，做的東西可以更完善，補足這次沒做好的地方，寫得大的同時程式也是簡潔清楚的。

五、 附錄

Mygame.h

```
#include "CookieMan.h"
#include "CandyMap.h"
#include "CandyMap2.h"
#include "BackGround.h"
#include "Sound.h"
extern bool stage[3];//關卡破關狀況
extern int stageScore;
namespace game_framework
{
	/////////////////////////////////////////////////////////////////////////////
	// 這個 class 乜為遊戲的遊戲開頭畫面物件
	// 每個 Member function 的 Implementation 都要弄懂
	/////////////////////////////////////////////////////////////////////////////
	class CGameStateInit : public CGameState
	{
	public:
		CGameStateInit(CGame* g);
		void OnInit();  // 遊戲的初值及圖形設定
		void OnBeginState();  // 設定每次重玩所需的變數
		void OnKeyUp(UINT, UINT, UINT);  // 處理鍵盤 Up 的動作
		void OnLButtonDown(UINT nFlags, CPoint point);  // 處理滑鼠的動作
		void OnMouseMove(UINT nFlags, CPoint point);  // 處理滑鼠的動作
	protected:
		void OnShow();  // 顯示這個狀態的遊戲畫面
	private:
		CMovingBitmap logo, play, pointer, about, explain, aboutBackground, back;
			// csie 的 logo
		CMovingBitmap explainBackground1, explainBackground2, explainBackground3, explainBackground4;
		CMovingBitmap page1, page2, page3, page4;
		bool playHover;
		int pointerPosition, backGround, page;
	};
	/////////////////////////////////////////////////////////////////////////////
	// 這個 class 為遊戲的選擇畫面
	// 每個 Member function 的 Implementation 都要弄懂
```

```
/////////////////////////////////////////////////////////////////////////////
class CGameStateStage : public CGameState
{
public:
	CGameStateStage(CGame* g);
	~CGameStateStage();
	void OnInit();
	void OnLButtonDown(UINT nFlags, CPoint point);
	void OnMouseMove(UINT nFlags, CPoint point);
	void OnKeyDown(UINT, UINT, UINT);
protected:
	void OnShow();
private:
	CMovingBitmap land1, land2, back, back_Hover, stageGround, lock;
	bool Land1Hover, Land2Hover, backHover;
	CMovingBitmap land1_Hover, land2_Hover;
};
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
// 每個 Member function 的 Implementation 都要弄懂
/////////////////////////////////////////////////////////////////////////////
class CGameStateRun : public CGameState
{
public:
	CGameStateRun(CGame* g);
	~CGameStateRun();
	void OnBeginState();                                    // 設定每次重玩所需的變數
	void OnInit();                                          // 遊戲的初值及圖形設定
	void OnKeyDown(UINT, UINT, UINT);
	void OnKeyUp(UINT, UINT, UINT);
	void OnLButtonDown(UINT nFlags, CPoint point);          // 處理滑鼠的動作
	void OnLButtonUp(UINT nFlags, CPoint point);            // 處理滑鼠的動作
	void OnMouseMove(UINT nFlags, CPoint point);            // 處理滑鼠的動作
	void OnRButtonDown(UINT nFlags, CPoint point);          // 處理滑鼠的動作
	void OnRButtonUp(UINT nFlags, CPoint point);            // 處理滑鼠的動作
	int success;
protected:
	void OnMove();
```

```
	void OnShow();                                          // 顯示這個狀態的遊戲畫面
private:
	CMovingBitmap	help;                                   // 說明圖
	CMovingBitmap	corner;                                 // 角落圖
	CInteger		hits_left;                              // 剩下的撞擊
	CInteger		life;                                   // 剩下的撞擊
	CookieMan		cMan;
	CandyMap		cMap;
	int lifeIndex;
	int bonusTimeState;
	BackGround		cBackGround;
};
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
// 每個 Member function 的 Implementation 都要弄懂
/////////////////////////////////////////////////////////////////////////////
class CGameStateRun_2 : public CGameState
{
public:
	CGameStateRun_2(CGame* g);
	~CGameStateRun_2();
	void OnBeginState();                                    // 設定每次重玩所需的變數
	void OnInit();                                          // 遊戲的初值及圖形設定
	void OnKeyDown(UINT, UINT, UINT);
	void OnKeyUp(UINT, UINT, UINT);
protected:
	void OnMove();
	void OnShow();
private:
	CookieMan	cMan;
	CandyMap2 cMap;
	CInteger		hits_left;                              // 剩下的撞擊
	CInteger		life;                                   // 剩下的撞擊
	BackGround		cBackGround;
	int lifeIndex;
	int bonusTimeState;
};
/////////////////////////////////////////////////////////////////////////////
```

```
    // 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
    // 每個 Member function 的 Implementation 都要弄懂
    /////////////////////////////////////////////////////////////////////////////
    class CGameStateBonus : public CGameState
    {
    public:
        CGameStateBonus(CGame* g);
        ~CGameStateBonus();
        void OnBeginState();                                    // 設定每次重玩所需的變數
        void OnInit();                                          // 遊戲的初值及圖形設定
        void OnKeyDown(UINT, UINT, UINT);
        void OnKeyUp(UINT, UINT, UINT);
    protected:
        void OnShow();
    private:
        CookieMan     cMan;
    };
    /////////////////////////////////////////////////////////////////////////////
    // 這個 class 為遊戲的結束狀態(Game Over)
    // 每個 Member function 的 Implementation 都要弄懂
    /////////////////////////////////////////////////////////////////////////////
    class CGameStateOver : public CGameState
    {
    public:
        CGameStateOver(CGame* g);
        //CGameStateOver(int score);
        void OnBeginState();                                    // 設定每次重玩所需的變數
        void OnInit();
        void OnLButtonDown(UINT nFlags, CPoint point);          // 處理滑鼠的動作
    protected:
        void OnMove();                                          // 移動遊戲元素
        void OnShow();                                          // 顯示這個狀態的遊戲畫面
    private:
        CMovingBitmap bmpEndGame1, bmpConfirm, gameOver, congratulation;
        CInteger score;
        int counter, endScore; // 倒數之計數器
    };
}
```

Mygame.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "mygame.h"
#include <stdio.h>
#define ABOUT_PAGE 1
#define SKILL_PAGE 2
bool stage[3] = { false,false,false };                                  //遊戲關卡破關狀態
int stageScore = 0;                                                     //分數
namespace game_framework {
    /////////////////////////////////////////////////////////////////////////////
    // 這個 class 為遊戲的遊戲開頭畫面物件
    /////////////////////////////////////////////////////////////////////////////
    CGameStateInit::CGameStateInit(CGame *g)
        : CGameState(g)
    {
        backGround = 0;
        page = 1;
    }
    void CGameStateInit::OnInit()
    {
    ///////////////////////////////////////////////////////////////////////////////////////////////////////
    // 當圖很多時，OnInit 載入所有的圖要花很多時間。為避免玩遊戲的人
    //       等的不耐煩，遊戲會出現「Loading ...」，顯示 Loading 的進度。
    ///////////////////////////////////////////////////////////////////////////////////////////////////////
        ShowInitProgress(0);                                            // 一開始的 loading 進度為 0%
                                                                        // 開始載入資料
        logo.LoadBitmap(IDB_start);
        about.LoadBitmap(IDB_about, RGB(255, 255, 255));
        explain.LoadBitmap(IDB_explain, RGB(255, 255, 255));
        aboutBackground.LoadBitmap(IDB_aboutBackground);
        explainBackground1.LoadBitmap(IDB_explainBackground1);
        explainBackground2.LoadBitmap(IDB_explainBackground2);
        explainBackground3.LoadBitmap(IDB_explainBackground3);
```

```
	explainBackground4.LoadBitmap(IDB_explainBackground4);
	page1.LoadBitmap(IDB_page1);
	page2.LoadBitmap(IDB_page2);
	page3.LoadBitmap(IDB_page3);
	page4.LoadBitmap(IDB_page4);
	back.LoadBitmap(IDB_BACK);
	playHover = false;
	play.LoadBitmap(IDB_PLAY, RGB(255, 255, 255));
	pointer.LoadBitmap(IDB_pointer, RGB(255, 255, 255));
	Sleep(300);                                          // 放慢，以便看清楚進度，實際遊戲請刪除此 Sleep
	//
	// 此 OnInit 動作會接到 CGameStaterRun::OnInit()，所以進度還沒到 100%
	//
}
void CGameStateInit::OnBeginState()
{
}
void CGameStateInit::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_ESC = 27;
	const char KEY_SPACE = ' ';
	if (nChar == KEY_SPACE)
		GotoGameState(GAME_STATE_RUN);                   // 切換至 GAME_STATE_RUN
	else if (nChar == KEY_ESC)                           // Demo 關閉遊戲的方法
		PostMessage(AfxGetMainWnd()->m_hWnd, WM_CLOSE, 0, 0); // 關閉遊戲
}
/////////////////////////////////////////////////////////////////////////////
//這個地方可以改成偵測滑鼠的位址選擇指定關卡
/////////////////////////////////////////////////////////////////////////////
void CGameStateInit::OnLButtonDown(UINT nFlags, CPoint point)
{
	if (point.x>195 && point.x < 445 && point.y>235 && point.y<301)
	{
		GotoGameState(GAME_STATE_STAGE);
	}
	if (point.x > 195 && point.x < 445 && point.y>315 && point.y < 381)
	{
		backGround = ABOUT_PAGE;
```

```
	}
	if (backGround == ABOUT_PAGE)
	{
		if (point.x > 500 && point.x < 597 && point.y>420 && point.y < 447) {
			backGround = 0;
		}
	}
	if (backGround == SKILL_PAGE)
	{
		if (point.x > 500 && point.x < 597 && point.y>420 && point.y < 447) {
			backGround = 0;
		}
		if (point.x > 450 && point.x < 485 && point.y>420 && point.y < 458) page = 4;
		if (point.x > 400 && point.x < 435 && point.y>420 && point.y < 458) page = 3;
		if (point.x > 350 && point.x < 385 && point.y>420 && point.y < 458) page = 2;
		if (point.x > 300 && point.x < 335 && point.y>420 && point.y < 458) page = 1;
	}
	else if (point.x > 195 && point.x < 445 && point.y>395 && point.y < 461)
	{
		backGround = SKILL_PAGE;
	}
}
void CGameStateInit::OnMouseMove(UINT nFlags, CPoint point)		// 處理滑鼠的動作
{
	if (point.x > 195 && point.x < 445 && point.y>235 && point.y < 301) {		//第一關
		playHover = true;
		pointerPosition = 240;
	}
	else if (point.x > 195 && point.x <445 && point.y>315 && point.y<381) {		//第二關
		playHover = true;
		pointerPosition = 320;
	}
	else if (point.x > 195 && point.x < 445 && point.y>395 && point.y < 461) {	//Back 鍵
		playHover = true;
		pointerPosition = 400;
	}
	else playHover = false;
}
```

```
void CGameStateInit::OnShow()
{
	CDC *pDC = CDDraw::GetBackCDC();		// 取得 Back Plain 的 CDC
	CFont f, *fp;
	f.CreatePointFont(160, "Times New Roman");		// 產生 font f; 160 表示 16 point 的字
	fp = pDC->SelectObject(&f);		// 選用 font f
	pDC->SetBkColor(RGB(0, 0, 0));
	pDC->SetTextColor(RGB(255, 255, 0));
	pDC->TextOut(120, 220, "Please click mouse or press SPACE to begin.");
	pDC->TextOut(5, 395, "Press Ctrl-F to switch in between window mode and full screen mode.");
	if (ENABLE_GAME_PAUSE)
		pDC->TextOut(5, 425, "Press Ctrl-Q to pause the Game.");
	pDC->TextOut(5, 455, "Press Alt-F4 or ESC to Quit.");
	pDC->SelectObject(fp);		// 放掉 font f (千萬不要漏了放掉)
	CDDraw::ReleaseBackCDC();		// 放掉 Back Plain 的 CDC
	if (backGround == 0) {
		page = 1;
		logo.SetTopLeft(0, 0);
		logo.ShowBitmap();
		explain.SetTopLeft(195, 395);
		explain.ShowBitmap();
		about.SetTopLeft(195, 315);
		about.ShowBitmap();
		play.SetTopLeft(195, 235);
		play.ShowBitmap();
		if (playHover == true) {
			pointer.SetTopLeft(80, pointerPosition);
			pointer.ShowBitmap();
		}
	}
	if (backGround == ABOUT_PAGE)//about
	{
		aboutBackground.SetTopLeft(0, 0);
		aboutBackground.ShowBitmap();
		back.SetTopLeft(500, 420);
		back.ShowBitmap();
	}
	if (backGround == SKILL_PAGE)//說明密技頁面
```

```
	{
		if (page == 1)
		{
			explainBackground1.SetTopLeft(0, 0);
			explainBackground1.ShowBitmap();
		}
		if (page == 2)
		{
			explainBackground2.SetTopLeft(0, 0);
			explainBackground2.ShowBitmap();
		}
		if (page == 3)
		{
			explainBackground3.SetTopLeft(0, 0);
			explainBackground3.ShowBitmap();
		}
		if (page == 4)
		{
			explainBackground4.SetTopLeft(0, 0);
			explainBackground4.ShowBitmap();
		}
		back.SetTopLeft(500, 420);
		back.ShowBitmap();
		page4.SetTopLeft(450, 420);
		page4.ShowBitmap();
		page3.SetTopLeft(400, 420);
		page3.ShowBitmap();
		page2.SetTopLeft(350, 420);
		page2.ShowBitmap();
		page1.SetTopLeft(300, 420);
		page1.ShowBitmap();
	}
}
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的選擇狀態(Stage)
/////////////////////////////////////////////////////////////////////////////
CGameStateStage::CGameStateStage(CGame *g) : CGameState(g)
{
```

```
}
CGameStateStage::~CGameStateStage()
{
}
void CGameStateStage::OnInit()
{
	land1.LoadBitmap(IDB_LAND_1);
	land2.LoadBitmap(IDB_LAND_2);
	land1_Hover.LoadBitmap(IDB_LAND_1_HOVER);
	land2_Hover.LoadBitmap(IDB_LAND_2_HOVER);
	back.LoadBitmap(IDB_BACK);
	lock.LoadBitmap(IDB_lock, RGB(255, 255, 255));
	//lock.LoadBitmap(IDB_lock, RGB(255, 255, 255));
	back_Hover.LoadBitmap(IDB_back_hover);
	stageGround.LoadBitmap(IDB_STAGE_GROUND);
}
/////////////////////////////////////////////////////////////////////////////
//這個地方可以改成偵測滑鼠的位址選擇指定關卡
/////////////////////////////////////////////////////////////////////////////
void CGameStateStage::OnLButtonDown(UINT nFlags, CPoint point)
{
	if (point.x > 500 && point.x < 600 && point.y>420 && point.y < 449)
	{
		GotoGameState(GAME_STATE_INIT);
	}
	if (point.x > 70 && point.x < 289 && point.y>102 && point.y <  322)
	{
		GotoGameState(GAME_STATE_RUN);
	}
	if (point.x >350 && point.x < 579 && point.y>102 && point.y < 322 && stage[0] == true)
	{
		GotoGameState(GAME_STATE_RUN_2);
	}
}
void CGameStateStage::OnMouseMove(UINT nFlags, CPoint point)                    // 處理滑鼠的動作
{
	if (point.x > 70 && point.x < 289 && point.y>102 && point.y <  322)
		Land1Hover = true;
```

```
	else Land1Hover = false;
	if (point.x > 350 && point.x < 579 && point.y>102 && point.y < 322)	Land2Hover = true;
	else Land2Hover = false;
	if (point.x > 500 && point.x < 597 && point.y>420 && point.y < 447)
		backHover = true;
	else backHover = false;
}
void CGameStateStage::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags) {
	const char KEY_A = 65;
	const char KEY_S = 83;
	if (nChar == KEY_A) {
		stage[0] = true;
		stage[1] = true;
	}
	if (nChar == KEY_S) {
		stage[0] = false;
		stage[1] = false;
	}
}
void CGameStateStage::OnShow()
{
	stageGround.SetTopLeft(0, 0);
	stageGround.ShowBitmap();
	if (Land1Hover == true) {
		land1_Hover.SetTopLeft(70, 102);
		land1_Hover.ShowBitmap();
	}
	else
	{
		land1.SetTopLeft(70, 87);
		land1.ShowBitmap();
	}
	if (stage[0] == false) {
		land2_Hover.SetTopLeft(350, 87);
		land2_Hover.ShowBitmap();
	}
	else if (Land2Hover == true) {
		land2_Hover.SetTopLeft(350, 102);
```

```
		land2_Hover.ShowBitmap();
	}
	else {
		land2.SetTopLeft(350, 87);
		land2.ShowBitmap();
	}
	if (backHover == true) {
		back_Hover.SetTopLeft(500, 420);
		back_Hover.ShowBitmap();
	}
	else {
		back.SetTopLeft(500, 420);
		back.ShowBitmap();
	}
	if (stage[0] == false) {
		lock.SetTopLeft(156, 25);
		lock.ShowBitmap();
	}
	if (stage[1] == false) {
		lock.SetTopLeft(436, 25);
		lock.ShowBitmap();
	}
}
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的結束狀態(Game Over)
/////////////////////////////////////////////////////////////////////////////
CGameStateOver::CGameStateOver(CGame *g)
	:CGameState(g)
{
}
void CGameStateOver::OnBeginState()
{
	counter = 30 * 5; // 5 seconds
	score.SetInteger(0);
	score.SetTopLeft(230, 250);
}
void CGameStateOver::OnLButtonDown(UINT nFlags, CPoint point) {
	if (point.x > 205 && point.x < 406 && point.y>406 && point.y < 473)
```

```
	{
		GotoGameState(GAME_STATE_INIT);
	}
}
void CGameStateOver::OnInit()
{
	///////////////////////////////////////////////////////////////////////////
	// 當圖很多時，OnInit 載入所有的圖要花很多時間。為避免玩遊戲的人
	//	等的不耐煩，遊戲會出現「Loading ...」，顯示 Loading 的進度。
	///////////////////////////////////////////////////////////////////////////
	bmpEndGame1.LoadBitmap(IDB_endGame1);
	bmpConfirm.LoadBitmap(IDB_CONFIRM, RGB(255, 255, 255));
	gameOver.LoadBitmap(IDB_gameover, RGB(255, 255, 255));
	congratulation.LoadBitmap(IDB_congratulation, RGB(255, 255, 255));
	score.LoadBitmap();
	ShowInitProgress(66);				// 接個前一個狀態的進度，此處進度視為 66%
	///////////////////////////////////////////////////////////////////////////
	// 開始載入資料
	///////////////////////////////////////////////////////////////////////////
	Sleep(300);				// 放慢，以便看清楚進度，實際遊戲請刪除此 Sleep
	ShowInitProgress(100);
	///////////////////////////////////////////////////////////////////////////
	// 最終進度為 100%
	///////////////////////////////////////////////////////////////////////////
}
void CGameStateOver::OnMove()
{
	//counter--;
	//if (counter < 0)
	//	GotoGameState(GAME_STATE_INIT);
}
void CGameStateOver::OnShow()
{
	if (score.GetInteger() != stageScore) {
		score.Add(stageScore);
	}
	//GAMEOVER 畫面
	bmpEndGame1.SetTopLeft(0, 0);
```

```
		bmpEndGame1.ShowBitmap();
		bmpConfirm.SetTopLeft(205, 406);
		bmpConfirm.ShowBitmap();
		if (stage[2] == 0) {                                        //GAMEOVER 死亡狀態
			gameOver.SetTopLeft(100, 100);
			gameOver.ShowBitmap();
		}
		else if (stage[2] == 1) {//勝利狀態
			congratulation.SetTopLeft(100, 100);
			congratulation.ShowBitmap();
		}
		score.ShowBitmap();
}
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
/////////////////////////////////////////////////////////////////////////////
CGameStateRun::CGameStateRun(CGame *g)
		: CGameState(g)
{
		bonusTimeState = 0;
}
CGameStateRun::~CGameStateRun()
{
		//delete [] ball;
}
void CGameStateRun::OnBeginState()
{
		const int BALL_GAP = 90;
		const int BALL_XY_OFFSET = 45;
		const int BALL_PER_ROW = 7;
		const int HITS_LEFT = 0;
		const int HITS_LEFT_X = 590;
		const int HITS_LEFT_Y = 0;
		const int BACKGROUND_X = 60;
		const int ANIMATION_SPEED = 15;
		const int CANDY_PER_ROW = 15;
		const int CANDY_GAP = 10;
		cMan.Init();
```

```
	cMap.Init();
	cMap.LoadBitmap();
	cBackGround.LoadBitmap();
	stage[2] = 0;
	help.SetTopLeft(0, SIZE_Y - help.Height());                    // 設定說明圖的起始座標
	hits_left.SetInteger(HITS_LEFT);                               // 指定剩下的撞擊數
	hits_left.SetTopLeft(HITS_LEFT_X, HITS_LEFT_Y);                // 指定剩下撞擊數的座標
	CAudio::Instance()->Play(AUDIO_LAKE, true);                    // 撥放 WAVE
	CAudio::Instance()->Play(AUDIO_DING, false);                   // 撥放 WAVE
	CAudio::Instance()->Play(AUDIO_NTUT, true);                    // 撥放 MIDI
}
void CGameStateRun::OnInit()                                       // 遊戲的初值及圖形設定
{
	/////////////////////////////////////////////////////////////////////////////
	// 當圖很多時，OnInit 載入所有的圖要花很多時間。為避免玩遊戲的人
	//       等的不耐煩，遊戲會出現「Loading ...」，顯示 Loading 的進度。
	/////////////////////////////////////////////////////////////////////////////
	ShowInitProgress(33);                          // 接個前一個狀態的進度，此處進度視為 33%
	/////////////////////////////////////////////////////////////////////////////
	// 開始載入資料
	/////////////////////////////////////////////////////////////////////////////
	cMan.LoadBitmap();
	cBackGround.LoadBitmap();
	//background.LoadBitmap(IDB_BACKGROUND);                       // 載入背景的圖形
	//
	// 完成部分 Loading 動作，提高進度
	//
	ShowInitProgress(50);
	Sleep(300);                                    // 放慢，以便看清楚進度，實際遊戲請刪除此 Sleep
	/////////////////////////////////////////////////////////////////////////////
	// 繼續載入其他資料
	/////////////////////////////////////////////////////////////////////////////
	help.LoadBitmap(IDB_HELP, RGB(255, 255, 255));                 // 載入說明的圖形
	corner.LoadBitmap(IDB_CORNER);                                 // 載入角落圖形
	hits_left.LoadBitmap();
}
void CGameStateRun::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
```

```
	const char KEY_LEFT = 0x25;	// keyboard 左箭頭
	const char KEY_UP = 0x26;	// keyboard 上箭頭
	const char KEY_RIGHT = 0x27;	// keyboard 右箭頭
	const char KEY_DOWN = 0x28;	// keyboard 下箭頭
	const char KEY_Q = 81;
	const char KEY_W = 87;
	const char KEY_E = 69;
	const char KEY_R = 82;
	const char KEY_SPACE = ' ';
	if (nChar == KEY_Q)cMap.SetBigTrans();
	if (nChar == KEY_W) cMap.SetTurbleTrans();
	if (nChar == KEY_E) cMap.SetMagnetTrans();
	if (nChar == KEY_R) cMap.SetAddLife();
	if (nChar == KEY_SPACE) cMap.SetQucikMove();
	if (nChar == KEY_UP) cMan.SetMovingUp(true);
	if (nChar == KEY_DOWN)cMan.SetMovingDown(true);
}
void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25;	// keyboard 左箭頭
	const char KEY_UP = 0x26;	// keyboard 上箭頭
	const char KEY_RIGHT = 0x27;	// keyboard 右箭頭
	const char KEY_DOWN = 0x28;	// keyboard 下箭頭
	if (nChar == KEY_UP) {
		cMan.SetMovingUp(false);
	}
	if (nChar == KEY_DOWN) {
		cMan.SetMovingDown(false);
	}
}
void CGameStateRun::OnLButtonDown(UINT nFlags, CPoint point)	// 處理滑鼠的動作
{
	//eraser.SetMovingLeft(true);
}
void CGameStateRun::OnLButtonUp(UINT nFlags, CPoint point)	// 處理滑鼠的動作
{
	//eraser.SetMovingLeft(false);
}
```

```
void CGameStateRun::OnMouseMove(UINT nFlags, CPoint point)                    // 處理滑鼠的動作
{
    // 沒事。如果需要處理滑鼠移動的話，寫 code 在這裡
}
void CGameStateRun::OnRButtonDown(UINT nFlags, CPoint point)                  // 處理滑鼠的動作
{
    //eraser.SetMovingRight(true);
}
void CGameStateRun::OnRButtonUp(UINT nFlags, CPoint point)                    // 處理滑鼠的動作
{
    //eraser.SetMovingRight(false);
}
void CGameStateRun::OnMove()                                                  // 移動遊戲元素
{
    ///////////////////////////////////////////////////////////////////////////////////////////
    // 如果希望修改 cursor 的樣式，則將下面程式的 commment 取消即可
    ///////////////////////////////////////////////////////////////////////////////////////////
    if (cMap.IsEnd()) {
        cMan.SetIsLastMap(true);
        if (cMan.GetX1() > SIZE_X) {
            CAudio::Instance()->Stop(AUDIO_LAKE);                             // 停止 WAVE
            CAudio::Instance()->Stop(AUDIO_NTUT);                             // 停止 MIDI
            success = 1;
            stageScore = hits_left.GetInteger();
            GotoGameState(GAME_STATE_OVER);
        }
    }
    else {
        cMan.SetIsLastMap(false);
    }
    cMan.OnMove();
    cMap.OnMove();
    cBackGround.OnMove();
    int score = cMap.HitCookieMan(&cMan);
    ///////////////////////////////////////////////////////////////////////////////////////////
    //判斷每次移動 index 會撞到的分數或是不適障礙物 score=-1 表示撞到障礙物
    ///////////////////////////////////////////////////////////////////////////////////////////
    if (score >= 0) {
```

```
        hits_left.Add(score);                                   //碰撞後加分
}
if (score == -1) {                                              //初始得分狀態
        cBackGround.status = 1;
}
cMan.SetBig(cMap.GetIsBigMode());
cMan.SetIsOnFloor(cMap.GetIsOnFloor());
cMan.SetNowY(cMap.GetNowY());
if (cMap.GetLifeNow() <= 0 || cMan.IsOutBorder() == 1) //cMap.IsEnd() == 1
{
        CAudio::Instance()->Stop(AUDIO_LAKE);                   // 停止 WAVE
        CAudio::Instance()->Stop(AUDIO_NTUT);                   // 停止 MIDI
        success = 0;
        stageScore = hits_left.GetInteger();
        GotoGameState(GAME_STATE_OVER);
}
if (cMap.IsEnd() == 1)                                          //跑完關卡
{
        cBackGround.MoveState(false);
        stage[0] = true;
        stage[2] = true;
}
if (score == -2)
{
        cMan.bonusTime = 1;
        score = 0;
}
if (cMan.bonusTime == 2 && bonusTimeState != 2)                 //進入 bonusTime 狀態
{
        bonusTimeState = 1;
}
if (bonusTimeState == 1)                                        //預備 bonusTime 的動畫
{
        cBackGround.choose = 1;
        cMap.mapNow = cMap.mapChoose;
        cMap.mapChoose = 35;
        cMap.SetTurbleTrans();
        cMap.SetMagnetTrans();
```

```
            bonusTimeState = 2;
        }
        if (cMap.mapChoose == 40)                                   //BonusTime 走道終點，結束回到原本地圖
        {
            cBackGround.choose = 0;
            cMap.mapChoose = cMap.mapNow;
            cMan.SetNowY(0);
            cMap.SetTurbleTrans();
            cMap.SetMagnetTrans();
            cMan.bonusTime = 0;
            bonusTimeState = 0;
        }
}
void CGameStateRun::OnShow()
{
        cBackGround.OnShow();
        //
        //  注意：Show 裡面千萬不要移動任何物件的座標，移動座標的工作應由 Move 做才對，
        //        否則當視窗重新繪圖時(OnDraw)，物件就會移動，看起來會很怪。換個術語
        //        說，Move 負責 MVC 中的 Model，Show 負責 View，而 View 不應更動 Model。
        //  貼上背景圖、撞擊數、球、擦子、彈跳的球
        //
        help.ShowBitmap();                                                  // 貼上說明圖
        hits_left.ShowBitmap();
        //
        //  貼上左上及右下角落的圖
        //
        corner.SetTopLeft(0, 0);
        corner.ShowBitmap();
        corner.SetTopLeft(SIZE_X - corner.Width(), SIZE_Y - corner.Height());
        corner.ShowBitmap();
        cMap.OnShow();
        cMan.OnShow();                                                      // 貼上人物
}
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
/////////////////////////////////////////////////////////////////////////////
CGameStateRun_2::CGameStateRun_2(CGame *g) : CGameState(g)
```

```
{
	bonusTimeState = 0;
}
CGameStateRun_2::~CGameStateRun_2()
{
	//delete [] ball;
}
void CGameStateRun_2::OnBeginState()
{
	const int BALL_GAP = 90;
	const int BALL_XY_OFFSET = 45;
	const int BALL_PER_ROW = 7;
	const int HITS_LEFT = 0;
	const int HITS_LEFT_X = 590;
	const int HITS_LEFT_Y = 0;
	const int BACKGROUND_X = 60;
	const int ANIMATION_SPEED = 15;
	const int CANDY_PER_ROW = 15;
	const int CANDY_GAP = 10;
	cMan.Init();
	cMap.Init();
	cMap.LoadBitmap();
	cBackGround.LoadBitmap();
	stage[2] = 0;
	hits_left.SetInteger(HITS_LEFT);                                    // 指定剩下的撞擊數
	hits_left.SetTopLeft(HITS_LEFT_X, HITS_LEFT_Y);                     // 指定剩下撞擊數的座標
	CAudio::Instance()->Play(AUDIO_LAKE, true);                         // 撥放 WAVE
	CAudio::Instance()->Play(AUDIO_DING, false);                        // 撥放 WAVE
	CAudio::Instance()->Play(AUDIO_NTUT, true);                         // 撥放 MIDI
}
void CGameStateRun_2::OnInit()
{
	cMan.LoadBitmap();
	cBackGround.LoadBitmap();
	hits_left.LoadBitmap();
	CAudio::Instance()->Load(AUDIO_DING, "sounds\\ding.wav");           // 載入編號 0 的聲音 ding.wav
	CAudio::Instance()->Load(AUDIO_LAKE, "sounds\\lake.mp3");           // 載入編號 1 的聲音 lake.mp3
	CAudio::Instance()->Load(AUDIO_NTUT, "sounds\\background2.mp3");    // 載入編號 2 的聲音 ntut.mid
```

```
}
void CGameStateRun_2::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25;	// keyboard 左箭頭
	const char KEY_UP = 0x26;	// keyboard 上箭頭
	const char KEY_RIGHT = 0x27;	// keyboard 右箭頭
	const char KEY_DOWN = 0x28;	// keyboard 下箭頭
	const char KEY_Q = 81;
	const char KEY_W = 87;
	const char KEY_E = 69;
	const char KEY_R = 82;
	const char KEY_SPACE = ' ';
	if (nChar == KEY_Q) cMap.SetBigTrans();
	if (nChar == KEY_W) cMap.SetTurbleTrans();
	if (nChar == KEY_E) cMap.SetMagnetTrans();
	if (nChar == KEY_R) cMap.SetAddLife();
	if (nChar == KEY_SPACE) cMap.SetQucikMove();
	if (nChar == KEY_UP) {
		cMan.SetMovingUp(true);
	}
	if (nChar == KEY_DOWN)
		cMan.SetMovingDown(true);
}
void CGameStateRun_2::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	const char KEY_LEFT = 0x25;	// keyboard 左箭頭
	const char KEY_UP = 0x26;	// keyboard 上箭頭
	const char KEY_RIGHT = 0x27;	// keyboard 右箭頭
	const char KEY_DOWN = 0x28;	// keyboard 下箭頭
	if (nChar == KEY_UP) {
		cMan.SetMovingUp(false);
	}
	if (nChar == KEY_DOWN) {
		cMan.SetMovingDown(false);
	}
}
void CGameStateRun_2::OnMove()	// 移動遊戲元素
{
```

```
if (cMap.IsEnd()) {
	cMan.SetIsLastMap(true);
	if (cMan.GetX1() > SIZE_X) {
		CAudio::Instance()->Stop(AUDIO_LAKE);		// 停止 WAVE
		CAudio::Instance()->Stop(AUDIO_NTUT);		// 停止 MIDI
		stageScore = hits_left.GetInteger();
		GotoGameState(GAME_STATE_OVER);
	}
}
else {
	cMan.SetIsLastMap(false);
}
cMan.OnMove();
cMap.OnMove();
cBackGround.OnMove();
//////////////////////////////////////////////////////////////////////////////
//判斷每次移動 index 會撞到的分數或是不適障礙物 score=-1 表示撞到障礙物
//////////////////////////////////////////////////////////////////////////////
int score = cMap.HitCookieMan(&cMan);
if (score >= 0) {
	hits_left.Add(score);//碰撞後加分
}
if (score == -1) {
	cBackGround.status = 1;
}
cMan.SetBig(cMap.GetIsBigMode());
cMan.SetIsOnFloor(cMap.GetIsOnFloor());
cMan.SetNowY(cMap.GetNowY());
if (cMap.GetLifeNow() <= 0 || cMan.IsOutBorder() == 1)//cMap.IsEnd() == 1 ||
{
	CAudio::Instance()->Stop(AUDIO_LAKE); // 停止 WAVE
	CAudio::Instance()->Stop(AUDIO_NTUT); // 停止 MIDI
	stageScore = hits_left.GetInteger();
	GotoGameState(GAME_STATE_OVER);
}
if (cMap.IsEnd() == 1)
{
	cBackGround.MoveState(false);
```

```
			stage[1] = true;
			stage[2] = true;
		}
		if (score == -2)
		{
			cMan.bonusTime = 1;
			score = 0;
		}
		if (cMan.bonusTime == 2 && bonusTimeState != 2)                    //進入 bonusTime 狀態
		{
			bonusTimeState = 1;
		}
		if (bonusTimeState == 1)                                           //預備 bonusTime 動畫
		{
			cBackGround.choose = 1;
			cMap.mapNow = cMap.mapChoose;
			cMap.mapChoose = 15;
			cMap.SetTurbleTrans();
			cMap.SetMagnetTrans();
			bonusTimeState = 2;
		}
		if (cMap.mapChoose == 19) {
			cBackGround.choose = 0;
			cMap.mapChoose = cMap.mapNow;
			cMan.bonusTime = 0;
			cMap.SetTurbleTrans();
			cMap.SetMagnetTrans();
			cMan.SetNowY(0);
			bonusTimeState = 0;
		}
}
void CGameStateRun_2::OnShow()
{
		cBackGround.OnShow();
		hits_left.ShowBitmap();
		cMap.OnShow();
		cMan.OnShow();                                                     // 貼上人物
}
```

```
	///////////////////////////////////////////////////////////////////////////
	// 這個 class 為遊戲的遊戲執行物件，主要的遊戲程式都在這裡
	///////////////////////////////////////////////////////////////////////////
	CGameStateBonus::CGameStateBonus(CGame *g) :CGameState(g)
	{
	}
	CGameStateBonus::~CGameStateBonus()
	{
	}
	void CGameStateBonus::OnBeginState()
	{
	}
	void CGameStateBonus::OnInit()
	{
	}
	void CGameStateBonus::OnKeyDown(UINT, UINT, UINT)
	{
	}
	void CGameStateBonus::OnKeyUp(UINT, UINT, UINT)
	{
	}
	void CGameStateBonus::OnShow()
	{
	}
}
```

CookieMan.h

```
#ifndef COOKIE_MAN
#define COOKIE_MAN
#include "Sound.h"
namespace game_framework {
	///////////////////////////////////////////////////////////////////////////
	// 這個 class 為遊戲人物:cookieMan
	///////////////////////////////////////////////////////////////////////////
	class CookieMan {
	public:
		CookieMan();
		int	GetX1();				// 擦子左上角 x 座標
		int	GetY1();				// 擦子左上角 y 座標
```

```
        int    GetX2();                                                    // 擦子右下角 x 座標
        int    GetY2();                                                    // 擦子右下角 y 座標
        void Init();
        void LoadBitmap();
        void SetMovingUp(bool);                                            //跳起來
        void SetMovingDown(bool flag);                                     //趴下
        void OnMove();
        void OnShow();
        void SetIsOnFloor(bool onFloor);
        void SetBig(bool turbleState);
        void SetNowY(int nowY);
        void MoveX(int x);
        void RunToEnd();
        void SetIsLastMap(bool mode);
        bool IsOutBorder();
        int bonusTime;
    private:
        CAnimation cookieMan, cookieManDown, cookieManJump, cookieManOneJump, cookieManBonusTime,
cookieManJumpOneBig, cookieManJumpBig;                                     // 擦子的動畫
        CAnimation cookieManBig;
        CMovingBitmap testLoad;
        const int JUMP_TOP;                                                //跳的最高點
        bool isBig;
        bool isOnFloor;
        bool upFlag;//跳的狀態
        bool downFlag;
        bool isOut;
        int groundX, groundY;
        int nowX, nowY;
        int maxHeight;
        int gravity;                                                       //重力
        int check;
        int checkY;                                                        //跳躍的頂點
        bool isLastMap;
    };
}
#endif
```

CookieMan.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "CookieMan.h"
#include "CandyMap.h"
#define PAN_Y                   30                                      //整張 Map 矩陣 y 軸向下平移距離->因為
namespace game_framework {
        ////////////////////////////////////////////////////////////////////////////
        // 這個 class 為 CookieMan
        ////////////////////////////////////////////////////////////////////////////
        CookieMan::CookieMan() :JUMP_TOP(100)
        {
                Init();
        }
        void CookieMan::Init() {
                const int X_POS = 100;
                const int Y_POS = 250 + PAN_Y;
                const int JUMP_TIME = 2;
                const int MAX_HEIGHT = 100 + PAN_Y;
                nowX = groundX = X_POS;
                groundY = Y_POS;
                nowY = 200;
                maxHeight = MAX_HEIGHT;
                upFlag = downFlag = false;
                isBig = false;
                isOut = false;
                isOnFloor = false;
                check = checkY = 0;
                bonusTime = 0;
                isLastMap = false;
        }
        void CookieMan::LoadBitmap() {
                if (testLoad.isBitmapLoad() == false) {
                        testLoad.LoadBitmap(IDB_testLoad);
                        cookieMan.AddBitmap(IDB_cookieMan1, RGB(255, 255, 255));
```

```
		cookieMan.AddBitmap(IDB_cookieMan1_run1, RGB(255, 255, 255));
		cookieMan.AddBitmap(IDB_cookieMan1_run2, RGB(255, 255, 255));
		cookieMan.AddBitmap(IDB_cookieMan1_run3, RGB(255, 255, 255));
		cookieManDown.AddBitmap(IDB_cookieMan1_down4, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_1, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_2, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_3, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_4, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_3, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_4, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_3, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_4, RGB(255, 255, 255));
		cookieManBonusTime.AddBitmap(IDB_bonus_3, RGB(255, 255, 255));
		cookieManOneJump.AddBitmap(IDB_JUMP_0, RGB(255, 255, 255));
		cookieManJump.AddBitmap(IDB_jump_3, RGB(255, 255, 255));
		cookieManJump.AddBitmap(IDB_jump_4, RGB(255, 255, 255));
		cookieManJump.AddBitmap(IDB_jump_5, RGB(255, 255, 255));
		cookieManJump.AddBitmap(IDB_jump_7, RGB(255, 255, 255));
		cookieManBig.AddBitmap(IDB_cookieManBig, RGB(255, 255, 255));		//人物變大
		cookieManBig.AddBitmap(IDB_cookieManBig_run1, RGB(255, 255, 255));
		cookieManBig.AddBitmap(IDB_cookieManBig_run2, RGB(255, 255, 255));
		cookieManBig.AddBitmap(IDB_cookieManBig_run3, RGB(255, 255, 255));
		cookieManJumpOneBig.AddBitmap(IDB_jumpBig0, RGB(255, 255, 255));
		cookieManJumpBig.AddBitmap(IDB_jumpBig2, RGB(255, 255, 255));
		cookieManJumpBig.AddBitmap(IDB_jumpBig3, RGB(255, 255, 255));
		cookieManJumpBig.AddBitmap(IDB_jumpBig4, RGB(255, 255, 255));
		cookieManJumpBig.AddBitmap(IDB_jumpBig5, RGB(255, 255, 255));
	}
}
void CookieMan::SetMovingUp(bool flag) {
	if (bonusTime == 2) {
		upFlag = flag;
		return;
	}
	if (flag == true) {
		//////////////////////////////////////////////////////////////////
		//現在人物踩在地板
		//////////////////////////////////////////////////////////////////
```

```
		if (isOnFloor == true) {
			check = 0;
			checkY = nowY;
			upFlag = flag;
		}
		//////////////////////////////////////////////////////////////////
		//人物狀態為空中跳躍且執行二段跳
		//////////////////////////////////////////////////////////////////
		if (check == 1) {
			upFlag = true;
		}
		//////////////////////////////////////////////////////////////////
		//人物狀態為空中跳躍一次爾已
		//////////////////////////////////////////////////////////////////
		if (isOnFloor == false && check == 0) {
			check = 1;
			upFlag = flag;
			checkY = nowY;
		}
	}
}
void CookieMan::SetMovingDown(bool flag) {
	//當踩在地板時才可以蹲下
	//沒有蹲下時 downFlag set False
	cookieManDown.Reset();                                          //蹲下動畫 重製
	if (flag == false) {//蹲下鍵為 false
		downFlag = flag;
	}
	//////////////////////////////////////////////////////////////////
	//踩在地板時 and 蹲下鍵為 true
	//////////////////////////////////////////////////////////////////
	if (isOnFloor&& flag == true) {
		downFlag = flag;
		nowY = nowY;
	}
}
```

```
//////////////////////////////////////////////////////////////////
//兩段跳最高的距離是 150*2==300
//////////////////////////////////////////////////////////////////
void CookieMan::OnMove()
{
	const int JUMP_DISTANCE = 30;                                   //向上加速度
	const int GRAVITY = 15;                                         //向下加速度
	gravity = GRAVITY;
	//判斷跳的最高上限
	cookieManDown.SetDelayCount(1);                                 //蹲下動畫
	cookieManDown.OnMove();
	cookieManJump.SetDelayCount(4);                                 //跳起來動畫
	cookieManJump.OnMove();
	cookieManOneJump.SetDelayCount(4);
	cookieManJump.OnMove();
	cookieManJumpBig.SetDelayCount(4);
	cookieManJumpBig.OnMove();
	cookieManJumpOneBig.SetDelayCount(4);
	cookieManJumpOneBig.OnMove();
	cookieMan.SetDelayCount(2);                                     //普通動畫
	cookieMan.OnMove();
	cookieManBig.SetDelayCount(2);
	cookieManBig.OnMove();
	if (bonusTime == 1) {
		cookieManBonusTime.SetDelayCount(25);
		cookieManBonusTime.OnMove();
		if (cookieManBonusTime.GetCurrentBitmapNumber() == 8)
		{
			upFlag = 0;
			bonusTime = 2;
			cookieManBonusTime.Reset();
		}
		if (nowY >= 20) {
			nowY -= 2;
		}
		return;
	}
	if (bonusTime == 2)
```

```
{
	if (upFlag == 1 && nowY >= 0)
	{
		nowY -= 12;
	}
	if (nowY <= 380) {
		nowY += 6;
	}
	return;
}
if (check == 1) {
	upFlag = 1;                                                    //跳的狀態
	if (nowY <= checkY - 150 || nowY <= 0) {
		upFlag = 0;
		check = 2;
	}
}
if (upFlag == 1) {
	if (nowY >= 0)
	{
		//CAudio::Instance()->Play(JUMP_1, false);
		nowY -= JUMP_DISTANCE;
	}
	if ((nowY <= checkY - 200 && check == 0)) {
		upFlag = 0;
	}
	if (nowY <= 0)
	{
		upFlag = 0;
	}
}
if (isOnFloor == false)                                            //跳起來狀態,地板上面
{
	nowY += gravity;
}
else if (nowY >= groundY &&downFlag == false)                      //踩在地板上
{
	nowY = groundY;
```

```
		upFlag = false;
	}
	if (nowY > SIZE_Y)//超出邊界 Y
	{
		isOut = true; //掉入地板下
	}
	if (isLastMap == true && nowX <= SIZE_X) {                    //走完整張地圖的最後一章了
		this->RunToEnd();
	}
}
void CookieMan::OnShow()
{
	if (bonusTime == 1) {
		cookieManBonusTime.SetTopLeft(nowX, nowY);
		cookieManBonusTime.OnShow();
	}
	else if (bonusTime == 2)
	{
		cookieManOneJump.SetTopLeft(nowX, nowY);
		cookieManOneJump.OnShow();
	}
	else if (downFlag == 1) {
		cookieManDown.SetTopLeft(nowX, nowY);
		cookieManDown.OnShow();
	}
	else if (isBig == true && check == 0 && isOnFloor == false) {
		cookieManJumpOneBig.SetTopLeft(nowX, nowY);
		cookieManJumpOneBig.OnShow();
	}
	else if (isBig == true && check != 0 && isOnFloor == false)
	{
		cookieManJumpBig.SetTopLeft(nowX, nowY);
		cookieManJumpBig.OnShow();
	}
	else if (isBig == true) {
		cookieManBig.SetTopLeft(nowX, nowY);
		cookieManBig.OnShow();
	}
```

```
	else if (check == 0 && isOnFloor == false)//||isOnFloor==false)
	{
		cookieManOneJump.SetTopLeft(nowX, nowY);
		cookieManOneJump.OnShow();
	}
	else if (check != 0 && isOnFloor == false)//||isOnFloor==false)
	{
		cookieManJump.SetTopLeft(nowX, nowY);
		cookieManJump.OnShow();
	}
	else {
		cookieMan.SetTopLeft(nowX, nowY);
		cookieMan.OnShow();
	}
}
void CookieMan::SetBig(bool bigState) {
	if (isBig == true && bigState == false) {			//改變狀態: 變大狀態即將結束，變回初始狀態
		isBig = bigState;
		nowY = SIZE_Y - cookieMan.Height() - 2 * 50;
		groundY = nowY;
	}
	else if (isBig == false && bigState == true) {			// 改變狀態:  從小變成大狀態 1
		isBig = bigState;
		nowY = SIZE_Y - cookieManBig.Height() - 2 * 50;
		groundY = nowY;
	}
	else if (isBig == false) {					//維持原狀態
		isBig = bigState;
	}
}
bool CookieMan::IsOutBorder() {
	return isOut;
}
void CookieMan::SetIsOnFloor(bool onFloor) {
	isOnFloor = onFloor;
}
void CookieMan::MoveX(int x) {
	nowX += x;
```

```
	}
	void CookieMan::RunToEnd() {
		if (nowX <= SIZE_X) {
			this->MoveX(10);
		}
	}
	void CookieMan::SetNowY(int y)
	{
		nowY = y;
	}
	void CookieMan::SetIsLastMap(bool mode) {
		isLastMap = mode;
	}
	int CookieMan::GetX1()
	{
		return nowX;
	}
	int CookieMan::GetY1()
	{
		return nowY;
	}
	int CookieMan::GetX2()
	{
		if (isBig) return nowX + cookieManBig.Width();
		else if (downFlag)return nowX + cookieManDown.Width();
		else return nowX + cookieMan.Width();
	}
	int CookieMan::GetY2()
	{
		if (isBig&&downFlag == false)return nowY + cookieManBig.Height();
		else if (downFlag)
			return nowY + cookieManDown.Height();
		else return nowY + cookieMan.Height();
	}
}
```

CandyMap.h

```
#include "Sound.h"
namespace game_framework {
```

```
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為糖果地圖:CandyMap
/////////////////////////////////////////////////////////////////////////////
class CandyMap {
public:
	CandyMap();
	void Init();
	bool IsAlive();                                          // 是否活著
	void OnMove();                                           // 移動
	void OnShow();
	void LoadBitmap();
	int IsEnd();                                             //地圖播放完畢
	int GetMH();
	int GetMW();
	int GetOnShowX(int realX);                               //在視窗畫面上相對的 X
	int GetOnShowY(int realY);                               //在視窗畫面上相對的 Y
	void SetBigTrans();
	void SetTurbleTrans();
	void SetMagnetTrans();
	void SetQucikMove();
	void SetAddLife();
	int GetNowY();
	bool GetIsTurbleMode();
	bool GetIsBigMode();
	bool GetIsOnFloor();
	int GetLifeNow();
	int HitCookieMan(CookieMan * cookie);                    // 是否碰到擦子
	int mapChoose, mapNow, totalScore;
private:
	CMovingBitmap bmpCandyGold, bmpCoin1, bmpCandySmile, bmpSkyStop1, bmpGroundStop1, bmpBonusTime;
	/////////////////////////////////////////////////////////////////////////
	// 球的圖
	/////////////////////////////////////////////////////////////////////////
	CMovingBitmap bmpHeart, bmpFire, bmpTrap1, bmpWall1, bmpMagnet, bmpBig;
	CAnimation animationTrap, animationGroundStop, animationShine;
	CMovingBitmap bmpBarBorder;
	CMovingBitmap lifeScroll;
	const int MAX_LIFE;
```

```
        int MOVE_WIDTH;
        int TURBO_MOVE_WIDTH;
        int mapXTimes;
        int map[42][9][20];
        int effectMap[42][9][20];                                   //物件的特殊效果
        int mapY[42][9][20];                                        //物件的 y 高度
        int mapX[42][9][20];                                        //物件的 X 位址
        int magnetYDistance[42][9][20];
        int magnetXDistance[42][9][20];
        int LIFE;
        int life, lifeX, lifeY;
        int MW, MH;                                                 //圖的寬高
        int manX1, manX2, manY1, manY2;
        int x, y;                                                   // 圓心的座標
        int allDistance;
        int index;
        int turbleIndex;
        bool onFloor;
        int bigIndex;
        int turbleDistance;
        bool turble;                                                //是否加速模式
        bool bigState;
        bool is_alive;                                              // 是否活著
        bool testMode;                                              //測試功能模式
        bool magnetMode;                                            //磁鐵模式
        int magnetTimes;
        int magnet;
        bool HitRectangle(int x1, int y1, int x2, int y2, int tx1, int ty1, int tx2, int ty2);     // 是否碰到參數範圍的矩形
        int HitRectangleBox(int tx1, int ty1, int tx2, int ty2);
    };
}
```

CookieMap.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
```

```
#include "CookieMan.h"
#include "CandyMap.h"
#define MAP_SIZE_X              1000
#defineMAP_TIMES                42                      //地圖 map 矩陣張數
#define PAN_Y                   30                      //整張 Map 矩陣 y 軸向下平移距離->因為
#define LIFE_X                  100;
#define LIFE_Y                  0;
#define CANDY_GOLD              1
#define CANDY_SMILE             2
#define COIN1                   3
#define HEART                   4
#define FIRE                    5
#define WALL_1                  6                       //腳底下的地板
#define TRAP_1                  7                       //離地板距離 100     一段跳就過啦
#defineSKY_STOP_1               8                       //離頂端距離 300 得蹲下才過得了
#define GROUND_STOP_1           9                       //離地板距離 150     得二段跳才過的了
#define BONUSTIME               10
#define MAGNET                  11
#define BIG                     12
#define RUN_OUT    1
namespace game_framework
{
        ////////////////////////////////////////////////////////////////////////////
        // 這個 class 為糖果地圖:CandyMap
        ////////////////////////////////////////////////////////////////////////////
        CandyMap::CandyMap() :MAX_LIFE(4000)
        {
                CandyMap::Init();
        }
        void CandyMap::Init()
        {
                LIFE = MAX_LIFE;                        //設置起始生命
                MOVE_WIDTH = 10;
                TURBO_MOVE_WIDTH = 20;
                life = LIFE;
                lifeX = LIFE_X;
                lifeY = LIFE_Y;
                MW = 50;
```

```
MH = 50;
x = y = 0;
index = 0;
allDistance = 0;
turbleIndex = 0;
bigIndex = 0;
magnetTimes = 0;
magnet = 0;
manY1 = 250 + PAN_Y;
turbleDistance = 0;
is_alive = true;
onFloor = false;
magnetMode = false;
bigState = false;
turble = false;
mapChoose = 0;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//預設 : 0 --> 用來選擇 MAP 矩陣從第幾張開始，想跳到第幾個 MAP 可以從這裡設定
    //0 有 groundstop/fire/trap
    //5 有 heart
    //8 有 BIG
    //17 有 skystop
    //21 有 magnet
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
mapNow = 0;
mapXTimes = MAP_SIZE_X / MW;
int map_init[MAP_TIMES][9][20] =
{
    {
        //0
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 6, 6, 6 ,6, 6, 0, 0, 0, 6, 6 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
```

```
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 0, 0, 0, 0, 0, 0, 0, 0 },
    },
    {
        //1 has groundStop
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 6, 6, 6, 6, 6, 6, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0 },
        { 6, 6, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 5, 1, 9, 1, 1, 1, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 7 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //2
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 2, 1, 1, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1 },
        { 1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 2, 2, 1, 1, 2, 2 },
        { 1, 2, 1, 2, 2, 1, 2, 2, 1, 2, 2, 1, 2, 2, 1, 2, 2, 1, 2, 2 },
        { 0, 0, 7, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //3
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
```

```
{
    //4                    //等等要再突起地方放夾子
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 10, 1, 1, 1 },
    { 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //5
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 2 },
    { 0, 0, 2, 0, 0, 2, 3, 2, 0, 4, 0, 0, 2, 3, 2, 0, 0, 0, 0, 2 },
    { 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 2 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //6
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 2, 2, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0 },
    { 1, 2, 0, 0, 0, 0, 1, 2, 1, 0, 0, 0, 0, 3, 2, 3, 0, 0, 0, 0 },
    { 2, 2, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //7                //要放 groundstop
```

```
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0 },
        { 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 },
        { 1, 5, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 9, 0 },
        { 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 9, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //8
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2 },
        { 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0 },
        { 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0 },
        { 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 12, 1, 1, 1, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //9 has groundStop
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 },
        { 9, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 },
        { 9, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0 },
        { 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //10
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
```

```
        { 2, 2, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 2, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 9, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 9, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0 },
        { 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //11
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 2, 2, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 2, 2, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 2, 0, 0, 9, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2 },
        { 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //12    //空的地方放 groundstop
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 9, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 9, 0, 2, 2, 2, 2, 2, 0, 9, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0 },
        { 0, 9, 0, 2, 2, 2, 2, 2, 0, 9, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //13
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
```

```
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1 },
		{ 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
	},
	{
		//14		放 groundstop
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 9, 0, 0, 2, 2, 2, 2, 2, 2, 0, 0, 0, 9, 0, 0, 0, 0, 0 },
		{ 1, 0, 9, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 9, 0, 0, 2, 2, 2 },
		{ 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 2, 2, 2 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
	},
	{
		//15
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 2, 2, 2, 2, 2, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0 },
		{ 2, 2, 2, 2, 2, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
		{ 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
	},
	{
		//16
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0 },
		{ 0, 0, 0, 0, 0, 0, 0, 2, 3, 3, 2, 0, 0, 0, 0, 2, 3, 3, 2, 0 },
		{ 0, 0, 0, 0, 0, 0, 2, 2, 3, 3, 2, 2, 0, 0, 2, 2, 3, 3, 2, 2 },
```

```
        { 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 0, 7, 0, 0, 2, 2, 2, 2, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //17
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 2, 3, 3, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 2, 2, 3, 3, 2, 2, 0, 0, 0, 1, 1, 1, 1, 1, 5, 1, 1, 1 },
        { 0, 0, 0, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //18            //放 skystop
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 0, 0, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 0, 0, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 0, 0, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 0, 0, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 0, 0, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 0, 0, 8, 0, 0, 8, 0, 0, 8, 0, 1, 1, 0, 8, 0, 0, 8, 0, 0, 8 },
        { 1, 1, 1, 1, 3, 3, 3, 3, 3, 1, 0, 0, 1, 3, 3, 3, 3, 1, 1, 1 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    },
    {
        //19            // set groundStop
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0 },
        { 0, 1, 1, 1, 1, 1, 9, 1, 1, 1, 1, 0, 1, 1, 1, 9, 1, 1, 1, 1 },
        { 3, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
```

```
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //20
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 },
        { 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //21
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 3, 11, 3, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1 },
        { 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //22            has trap_1
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 2, 0, 0, 2, 0, 0, 0, 0, 0, 0, 2, 0, 0, 2, 0, 0, 0, 0 },
        { 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
```

```
{
    //23            has ground
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0 },
    { 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0 },
    { 0, 0, 2, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 1, 0, 9, 0, 1, 0 },
    { 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 9, 0, 0, 1 },
    { 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //24            has trap
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 2, 2, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1 },
    { 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 7, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //25
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 0, 0, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0 },
    { 1, 1, 2, 2, 2, 2, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1 },
    { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
    { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
    //26
```

```
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 2, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0 },
        { 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0 },
        { 0, 1, 0, 9, 0, 1, 0, 0, 0, 0, 1, 0, 9, 0, 1, 0, 0, 0, 0, 0 },
        { 1, 0, 0, 9, 0, 0, 1, 1, 1, 1, 0, 0, 9, 0, 0, 1, 4, 1, 1, 1 },
        { 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //27
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 1, 1, 2, 2, 1, 1, 2, 1, 1, 2, 2, 2, 1, 1, 2, 2 },
        { 0, 0, 0, 0, 1, 1, 2, 2, 1, 1, 2, 1, 1, 2, 2, 2, 1, 1, 2, 2 },
        { 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 0, 7, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //28
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0 },
        { 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0 },
        { 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 9, 0, 1, 0, 0, 0 },
        { 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 9, 0, 0, 1, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //29
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0 },
```

```
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 2, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0, 0, 0 },
        { 0, 0, 1, 1, 0, 0, 0, 0, 0, 2, 0, 9, 0, 2, 0, 0, 0, 0, 0, 0 },
        { 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 9, 0, 0, 1, 1, 1, 1, 0, 0 },
        { 0, 0, 7, 0, 0, 0, 0, 0, 0, 0, 0, 9, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //30
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 9, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9, 9, 0, 0, 0, 0, 0, 0 },
        { 9, 9, 0, 1, 1, 1, 1, 1, 2, 2, 2, 0, 0, 9, 9, 0, 0, 2, 2, 2 },
        { 9, 9, 0, 1, 1, 1, 1, 1, 2, 2, 2, 0, 0, 9, 9, 0, 0, 2, 2, 2 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //31
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 2, 0, 0, 0, 0, 0 },
        { 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 2 },
        { 2, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //32
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 3, 3, 3, 3, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 1, 1, 1 },
        { 0, 3, 3, 3, 3, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 1, 4, 1 },
        { 0, 2, 2, 0, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0 },
```

```
        { 2, 2, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0 },
        { 2, 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //33
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //34
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
        { 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 2, 2, 2, 0, 0, 0, 0, 0, 0, 1 },
        { 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
        { 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6 },
},
{
        //BONUSTIME 35
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
```

```
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
    },
    {   //36
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
    },
    {   //37
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 ,1, 1, 1, 1, 1 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
        { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 ,2, 2, 2, 2, 2 },
    },
    {   //38
        { 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 2, 2, 2 ,2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
        { 0, 0, 0, 0, 2, 2, 2 ,2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 },
        { 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0 },
    },
    {   //39
```

```
			{ 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 2, 2, 2 ,2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
			{ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1 },
			{ 0, 0, 0, 0, 2, 2, 2 ,2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0 },
		},
		{	//40
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		},
		{	//41
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
			{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },
		}
	};
	for (int k = 0; k < MAP_TIMES; k++)
	{
		for (int i = 0; i < 9; i++)
		{
			for (int j = 0; j < 20; j++)
```

```
                {
                    map[k][i][j] = map_init[k][i][j];          //初始化地圖物件陣列
                    effectMap[k][i][j] = 0;                    //初始化物件特殊功能陣列
                    mapX[k][i][j] = 0;                         //初始化物件 X 軸位址
                    mapY[k][i][j] = 0;                         //初始化物件 Y 軸位址
                    magnetYDistance[k][i][j] = 0;              //初始化磁鐵每段 Y 吸引的距離
                    magnetXDistance[k][i][j] = 0;              //初始化磁鐵每段 X 吸引的距離
                }
            }
        }
}
void CandyMap::LoadBitmap()
{
    if (bmpCandyGold.isBitmapLoad() == false)                  //load 過就不重複 load
    {
        bmpCandyGold.LoadBitmap(IDB_candyGold, RGB(255, 255, 255));
        bmpCoin1.LoadBitmap(IDB_coin1, RGB(255, 255, 255));
        bmpCandySmile.LoadBitmap(IDB_candySmile, RGB(255, 255, 255));
        bmpSkyStop1.LoadBitmap(IDB_skyStop1, RGB(255, 255, 255));
        bmpHeart.LoadBitmap(IDB_heart, RGB(255, 255, 255));
        bmpFire.LoadBitmap(IDB_fire, RGB(255, 255, 255));
        bmpWall1.LoadBitmap(IDB_wall_1, RGB(255, 255, 255));
        lifeScroll.LoadBitmap(IDB_life, RGB(255, 255, 255));
        bmpBonusTime.LoadBitmap(IDB_BONUSTIME, RGB(255, 255, 255));
        bmpMagnet.LoadBitmap(IDB_magnet, RGB(255, 255, 255));
        bmpBig.LoadBitmap(IDB_big, RGB(255, 255, 255));
        bmpTrap1.LoadBitmap(IDB_trap_1, RGB(255, 255, 255));          //靜止模式
        bmpBarBorder.LoadBitmap(IDB_barBorder, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_1, RGB(255, 255, 255));      //動畫模式
        animationTrap.AddBitmap(IDB_trap_2, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_3, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_4, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_5, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_6, RGB(255, 255, 255));
        animationTrap.AddBitmap(IDB_trap_7, RGB(255, 255, 255));
        bmpGroundStop1.LoadBitmap(IDB_groundStop1, RGB(255, 255, 255));  //靜止模式
        animationGroundStop.AddBitmap(IDB_groundStop1, RGB(255, 255, 255));
        animationGroundStop.AddBitmap(IDB_groundStop2, RGB(255, 255, 255));
```

```
		animationGroundStop.AddBitmap(IDB_groundStop3, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop4, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop5, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop6, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop7, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop8, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop9, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop10, RGB(255, 255, 255));
		animationGroundStop.AddBitmap(IDB_groundStop11, RGB(255, 255, 255));
		animationShine.AddBitmap(IDB_shine1);
		animationShine.AddBitmap(IDB_shine2);
		animationShine.AddBitmap(IDB_shine3);
		CAudio::Instance()->Load(COIN_1, "sounds\\coin01.mp3");
	}
}
void CandyMap::OnShow()
{
	if ((allDistance) >= MAP_SIZE_X)			//超過視窗 SIZE_X 更換 Map 的 index
	{
		index = 0;
		allDistance = allDistance % 1000;		//規 0
		mapChoose += 1;
	}
	//最後一張地圖
	//遊戲結束
	if (this->IsEnd() == 1) {
		MOVE_WIDTH = 0;
	}
	if (turble == true && turbleIndex<110)			//進入 Turble 模式，計算 TURBLE 模式時間
	{
		if ((MOVE_WIDTH < TURBO_MOVE_WIDTH))
		{
			//////////////////////////////////////////////////////////////////////////
			//設定 MOVE_WIDTH 為 TURBO_MOVE_WIDTH(用漸加的方式
			//////////////////////////////////////////////////////////////////////////
			MOVE_WIDTH += 1;
		}
		else
```

```
        {
            MOVE_WIDTH = 20;
        }
        turbleIndex++;                                          //turble 時間增加
        ////////////////////////////////////////////////////////////////////////
        //Turble 時間到，進行結束動畫
        ////////////////////////////////////////////////////////////////////////
        if ((turbleIndex >= 105) && (turbleIndex <= 110)) {
            MOVE_WIDTH -= 1;                                    //逐漸變弱
            if (turbleIndex >= 100 || MOVE_WIDTH <= 10) {
                turble = false;
                turbleIndex = 0;
                MOVE_WIDTH = 10;
            }
        }
}
else MOVE_WIDTH = 10;
////////////////////////////////////////////////////////////////////////
//計算變大狀態時間
////////////////////////////////////////////////////////////////////////
if (bigState == true && bigIndex<200) {
        bigIndex++;
        if (bigIndex >= 200) {
            bigState = false;
            bigIndex = 0;
        }
}
////////////////////////////////////////////////////////////////////////
//計算磁鐵狀態時間
////////////////////////////////////////////////////////////////////////
if (magnetMode == true) {
        magnetTimes++;
        if (magnetTimes >= 200) {
            magnetMode = false;
            magnetTimes = 0;
        }
}
int effect = 0;                                                 //物件的特殊效果
```

```
int yHeight = 0;
int yHeight1 = 0;
int oldX = 0;                                                    //物件上一個狀態的 X
int oldY = 0;                                                    //物件上一個狀態的 Y
int magnetY = 0;                                                 //磁鐵狀態時 Y 需移動的距離
int magnetX = 0;                                                 //磁鐵狀態時 Y 需移動的距離
allDistance += MOVE_WIDTH;
//////////////////////////////////////////////////////////////////////
//每一次都多跑一張預備，才會有連貫得效果
//////////////////////////////////////////////////////////////////////
for (int i = 0; i < 9; i++)
{
    for (int j = 0; j < 20; j++)
    {
        effect = effectMap[mapChoose % MAP_TIMES][i][j];
        oldX = mapX[mapChoose % MAP_TIMES][i][j];
        oldY = mapY[mapChoose % MAP_TIMES][i][j];
        magnetY = magnetYDistance[mapChoose % MAP_TIMES][i][j];
        magnetX = magnetXDistance[mapChoose % MAP_TIMES][i][j];
        yHeight = y + MH * i + PAN_Y;
        yHeight1 = y + MH * i + PAN_Y;
        ////////////////////////////////////////////////////////////////////////////////////////////////////
        //畫面上的圖
        //顯示每個地圖資源，得分的 Candy，各種障礙物
        ////////////////////////////////////////////////////////////////////////////////////////////////////
        switch (map[mapChoose % MAP_TIMES][i][j])
        {
        case 0:
            break;
        case CANDY_GOLD:
            //////////////////////////////////////////////////////////////////////
            //設定磁鐵狀態被吸引的動畫
            //////////////////////////////////////////////////////////////////////
            if ((magnetMode == true) && (GetOnShowX(j) >= (manX1)) && (GetOnShowX(j) <=
(manX1 + 150)))                                                  //磁鐵模式
            {
            //////////////////////////////////////////////////////////////////////
            //平均 Move_width 移動距離裡 靠近的距離
```

```
                int newX = oldX - (oldX - manX1) / (MOVE_WIDTH / 2);
                int newY = oldY - (oldY - (manY2 + manY1) / 2) / (MOVE_WIDTH / 2);
                mapX[mapChoose % MAP_TIMES][i][j] = newX;
                mapY[mapChoose % MAP_TIMES][i][j] = newY;
                bmpCandyGold.SetTopLeft(newX, newY);
                bmpCandyGold.ShowBitmap();
            }
            ////////////////////////////////////////////////////////////////////////
            //正常模式的物件動畫
            ////////////////////////////////////////////////////////////////////////
            else {
                bmpCandyGold.SetTopLeft(((MW * j) - allDistance), yHeight);
                ////////////////////////////////////////////////////////////////////////
                //設定 candy1 的相對位置(j 為 X 軸,i 為 Y 軸)
                ////////////////////////////////////////////////////////////////////////
                bmpCandyGold.ShowBitmap();
                mapX[mapChoose % MAP_TIMES][i][j] = ((MW * j) - allDistance);
                mapY[mapChoose % MAP_TIMES][i][j] = yHeight;
            }
            break;
        case COIN1:
            ////////////////////////////////////////////////////////////////////////
            //設定磁鐵狀態被吸引的動畫
            ////////////////////////////////////////////////////////////////////////
            if ((magnetMode == true) && (GetOnShowX(j) > (manX1)) && (GetOnShowX(j) <=
(manX1 + 150)))
            {
                int newX = oldX - (oldX - manX1) / (MOVE_WIDTH / 2);
                int newY = oldY - (oldY - (manY2 + manY1) / 2) / (MOVE_WIDTH / 2);
                mapX[mapChoose % MAP_TIMES][i][j] = newX;
                mapY[mapChoose % MAP_TIMES][i][j] = newY;
                bmpCoin1.SetTopLeft(newX, newY);
                bmpCoin1.ShowBitmap();
            }
            ////////////////////////////////////////////////////////////////////////
            //正常模式的物件動畫
            ////////////////////////////////////////////////////////////////////////
```

```
                else
                {
                    bmpCoin1.SetTopLeft(((MW * j) - allDistance), yHeight);
                    bmpCoin1.ShowBitmap();
                    mapX[mapChoose % MAP_TIMES][i][j] = ((MW * j) - allDistance);
                    mapY[mapChoose % MAP_TIMES][i][j] = yHeight;
                }
                break;
            case CANDY_SMILE:
                ////////////////////////////////////////////////////////////////////////
                //設定磁鐵狀態被吸引的動畫
                ////////////////////////////////////////////////////////////////////////
                if ((magnetMode == true) && (GetOnShowX(j) > (manX1)) && (GetOnShowX(j) <=
(manX1 + 150)))
                {
                    int newX = oldX - (oldX - manX1) / (MOVE_WIDTH / 2);
                    int newY = oldY - (oldY - (manY2 + manY1) / 2) / (MOVE_WIDTH / 2);
                    mapX[mapChoose % MAP_TIMES][i][j] = newX;
                    mapY[mapChoose % MAP_TIMES][i][j] = newY;
                    bmpCandySmile.SetTopLeft(newX, newY);
                    bmpCandySmile.ShowBitmap();
                }
                ////////////////////////////////////////////////////////////////////////
                //正常模式的物件動畫
                ////////////////////////////////////////////////////////////////////////
                else
                {
                    bmpCandySmile.SetTopLeft(((MW * j) - allDistance), yHeight);
                    bmpCandySmile.ShowBitmap();
                    mapX[mapChoose % MAP_TIMES][i][j] = ((MW * j) - allDistance);
                    mapY[mapChoose % MAP_TIMES][i][j] = yHeight;
                }
                break;
            case SKY_STOP_1:
                ////////////////////////////////////////////////////////////////////////
                //設定被撞飛的動畫
                ////////////////////////////////////////////////////////////////////////
                if (effect == RUN_OUT)
```

```
		{
			bmpSkyStop1.SetTopLeft(oldX, oldY);
			bmpSkyStop1.ShowBitmap();
			mapX[mapChoose % MAP_TIMES][i][j] = oldX - 19;
			mapY[mapChoose % MAP_TIMES][i][j] = oldY - 10;
		}
		//////////////////////////////////////////////////////////////////////////
		//普通正常狀態
		//////////////////////////////////////////////////////////////////////////
		else
		{
			bmpSkyStop1.SetTopLeft(((MW * j) - allDistance), 0);
			bmpSkyStop1.ShowBitmap();
			mapX[mapChoose % MAP_TIMES][i][j] = ((MW * j) - allDistance);
			mapY[mapChoose % MAP_TIMES][i][j] = 0;
		}
		break;
	case GROUND_STOP_1:
		//////////////////////////////////////////////////////////////////////////
		//設定被撞飛的動畫
		//////////////////////////////////////////////////////////////////////////
		if (effect == RUN_OUT)
		{
			animationGroundStop.SetTopLeft(oldX, oldY);
			animationGroundStop.OnShow();
			mapX[mapChoose % MAP_TIMES][i][j] = oldX - 19;
			mapY[mapChoose % MAP_TIMES][i][j] = oldY - 15;
		}
		//////////////////////////////////////////////////////////////////////////
		//正常模式的物件動畫
		//////////////////////////////////////////////////////////////////////////
		else
		{
bmpGroundStop1.SetTopLeft(((MW*j)-allDistance),SIZE_Y - bmpGroundStop1.Height() - 2 * MH);
bmpGroundStop1.ShowBitmap();
mapX[mapChoose % MAP_TIMES][i][j] = ((MW * j) - allDistance);
mapY[mapChoose % MAP_TIMES][i][j] = SIZE_Y - bmpGroundStop1.Height() - 2 * MH;
		}
```

```
		break;
	case HEART:
		bmpHeart.SetTopLeft(((MW * j) - allDistance), yHeight);
		bmpHeart.ShowBitmap();
		break;
	case FIRE:
		bmpFire.SetTopLeft(((MW * j) - allDistance), yHeight);
		bmpFire.ShowBitmap();
		break;
	case TRAP_1:
		//////////////////////////////////////////////////////////////////////////
		//設定被撞飛的動畫
		//////////////////////////////////////////////////////////////////////////
		if (effect == RUN_OUT)
		{
			animationTrap.SetTopLeft(oldX, oldY);
			animationTrap.OnShow();
			mapX[mapChoose % MAP_TIMES][i][j] = oldX - 19;
			mapY[mapChoose % MAP_TIMES][i][j] = oldY - 15;
		}
		//////////////////////////////////////////////////////////////////////////
		//正常模式的物件動畫
		//////////////////////////////////////////////////////////////////////////
		else
		{
		bmpTrap1.SetTopLeft(((MW * j) - allDistance), (SIZE_Y - bmpTrap1.Height()) - 2 * MH);
		bmpTrap1.ShowBitmap();
		mapX[mapChoose % MAP_TIMES][i][j] = (MW * j) - allDistance;
		mapY[mapChoose % MAP_TIMES][i][j] = (SIZE_Y - bmpTrap1.Height()) - 2 * MH;
		}
		break;
	case WALL_1:
		bmpWall1.SetTopLeft(((MW * j) - allDistance), yHeight);
		bmpWall1.ShowBitmap();
		break;
	case BONUSTIME:
		bmpBonusTime.SetTopLeft(((MW * j) - allDistance), yHeight);
		bmpBonusTime.ShowBitmap();
```

```
            break;
        case MAGNET:
            bmpMagnet.SetTopLeft(((MW * j) - allDistance), yHeight);
            bmpMagnet.ShowBitmap();
            break;
        case BIG:
            bmpBig.SetTopLeft(((MW * j) - allDistance), yHeight);
            bmpBig.ShowBitmap();
            break;
        default:
            ASSERT(0);
        }
        ////////////////////////////////////////////////////////////////////////////////////////////////////////////
        //第二章圖(後臺預備)
        ////////////////////////////////////////////////////////////////////////////////////////////////////////////
        switch (map[(mapChoose + 1) % MAP_TIMES][i][j])
        {
        case 0:
            break;
        case CANDY_GOLD:
            bmpCandyGold.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
            bmpCandyGold.ShowBitmap();
            break;
        case COIN1:
            bmpCoin1.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
            bmpCoin1.ShowBitmap();
            break;
        case CANDY_SMILE:
            bmpCandySmile.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
            bmpCandySmile.ShowBitmap();
            break;
        case SKY_STOP_1:
            bmpSkyStop1.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), 0);
            bmpSkyStop1.ShowBitmap();
            break;
        case GROUND_STOP_1:
            bmpGroundStop1.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), (SIZE_Y -
bmpGroundStop1.Height()) - 2 * MH);
```

```
                    bmpGroundStop1.ShowBitmap();
                    break;
                case HEART:
                    bmpHeart.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpHeart.ShowBitmap();
                    break;
                case FIRE:
                    bmpFire.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpFire.ShowBitmap();
                    break;
                case TRAP_1:
                    bmpTrap1.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), (SIZE_Y -
bmpTrap1.Height()) - 2 * MH);
                    bmpTrap1.ShowBitmap();
                    break;
                case WALL_1:
                    bmpWall1.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpWall1.ShowBitmap();
                    break;
                case BONUSTIME:
                    bmpBonusTime.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpBonusTime.ShowBitmap();
                    break;
                case MAGNET:
                    bmpMagnet.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpMagnet.ShowBitmap();
                    break;
                case BIG:
                    bmpBig.SetTopLeft(MAP_SIZE_X + ((MW * j) - allDistance), yHeight);
                    bmpBig.ShowBitmap();
                    break;
                default:
                    ASSERT(0);
                }
            }
        }
        ////////////////////////////////////////////////////////////////////////
        //lifeScroll 生命條狀態動畫
```

```
	//////////////////////////////////////////////////////////////////////////
	for (int i = 0; i <life / 20; i++) {
		lifeScroll.SetTopLeft(lifeX + i * 2, y);
		lifeScroll.ShowBitmap();
	}
	animationShine.SetTopLeft(lifeX + (((life / 20) - 1) * 2), y);
	animationShine.OnShow();
	bmpBarBorder.SetTopLeft(lifeX, y);
	bmpBarBorder.ShowBitmap();
}
bool CandyMap::GetIsTurbleMode() {
	return turble;
}
bool CandyMap::GetIsBigMode() {
	return bigState;
}
int CandyMap::GetNowY() {
	return manY1;
}
int CandyMap::GetOnShowX(int realX) {
	return MW * realX - allDistance;
}
int CandyMap::GetOnShowY(int realY) {
	return MW * realY - allDistance;
}
bool CandyMap::GetIsOnFloor() {
	return onFloor;
}
bool CandyMap::IsAlive()
{
	return is_alive;
}
int CandyMap::IsEnd()
{
	//////////////////////////////////////////////////////////////////////////
	//計算當下位址是否在終點
	//////////////////////////////////////////////////////////////////////////
	int map = mapChoose + ((allDistance + 640) / 1000);
```

```
	if (map == (MAP_TIMES - 8) || mapChoose == (MAP_TIMES - 8))return 1;
	else return 0;
}
void CandyMap::OnMove()
{
	animationTrap.SetDelayCount(1);
	animationTrap.OnMove();
	animationGroundStop.SetDelayCount(1);
	animationGroundStop.OnMove();
	animationShine.SetDelayCount(2);
	animationShine.OnMove();
	index += 1;
	life--;
}
int CandyMap::GetMH()
{
	return MH;
}
int CandyMap::GetMW()
{
	return MW;
}
int CandyMap::HitCookieMan(CookieMan* cookie)
{
	manX1 = cookie->GetX1();
	manY1 = cookie->GetY1();
	manX2 = cookie->GetX2();
	manY2 = cookie->GetY2();
	return HitRectangleBox(manX1, manY1, manX2, manY2);
}
int CandyMap::HitRectangleBox(int tx1, int ty1, int tx2, int ty2)//人的 X1,Y1,X2,Y2
{
	int judgeMap = 0;
	//////////////////////////////////////////////////////////////////////////
	//判斷 CookieMap 當下的位置是否進入下一個 Map 矩陣，是的話矩陣 index++
	//////////////////////////////////////////////////////////////////////////
	if (((allDistance + tx1) / MW) + 1 >= mapXTimes)
		judgeMap = (mapChoose + 1) % MAP_TIMES;
```

```
else judgeMap = mapChoose % MAP_TIMES;
////////////////////////////////////////////////////////////////////////
//範圍內的是 map 中第幾個方塊 //加 1 表示 下一次有無表示
////////////////////////////////////////////////////////////////////////
int countX = (((allDistance + tx1) / MW) + 1) % mapXTimes;
int x1 = countX * MW;                                   // 判斷碰撞方塊的左上角 x 座標
int y1 = SIZE_Y - MH;                                   // 判斷碰撞方塊的左上角 y 座標
int x2 = x1 + MW;//bmp.Width();                         // 判斷碰撞方塊的右下角 x 座標
int y2 = y1 + MH;//bmp.Height();                        // 判斷碰撞方塊的右下角 y 座標
////////////////////////////////////////////////////////////////////////
// 檢測球的矩形與參數矩形是否有交集
////////////////////////////////////////////////////////////////////////
int i = 0;
int hitTimes = 0;
////////////////////////////////////////////////////////////////////////
//偵測人物腳踩地板的狀態
////////////////////////////////////////////////////////////////////////
if (map[judgeMap][ty2 / MH][countX] == WALL_1)          //判斷有無踩到地板
{
        if (ty2 >= ((ty2 / MH)*MH + PAN_Y)) {
                onFloor = true;
        }
        else {
                onFloor = false;
        }
}
else {
        onFloor = false;
}
////////////////////////////////////////////////////////////////////////
//MagnetMode 的糖果偵測
//設定範圍內才能被吸引
////////////////////////////////////////////////////////////////////////
if (magnetMode == true)
{
        for (int i = 0; i < 9; i++) {
                if (map[judgeMap][i][countX] < 6 && map[judgeMap][i][countX] > 0)
                {
```

```
                        if (map[judgeMap][i][countX] == FIRE)
                        {
                                turble = true;
                        }
                        else    if (map[judgeMap][i][countX] == HEART)
                        {
                                life += 300;
                                if (life >= MAX_LIFE)life = MAX_LIFE;
                                //bigState = true;
                        }
                        else
                        {
                                hitTimes += 1;
                        }
                        map[judgeMap][i][countX] = 0;
                }
        }
}
//
//偵測人物範圍裡碰撞到的物品
int indexY;//要指定的 Y 軸
for (i = 0; i < (ty2 - ty1) / MH; i++)
{
        ////////////////////////////////////////////////////////////////////////
        //判斷 CookieMan 範圍內的東西 0<X<6 為吃了可以加分數的
        //此狀態吃到的東西是功能/加分
        //magnetMode 的糖果偵測寫在前面，這裡不執行。
        ////////////////////////////////////////////////////////////////////////
        indexY = ty1 / MH + i;
        if (indexY > 8)indexY = 8;
if (map[judgeMap][indexY][countX] < 6 && map[judgeMap][indexY][countX] > 0 && magnetMode == false)
        {
                CAudio::Instance()->Play(COIN_1, false);
                hitTimes += 1;
                if (map[judgeMap][indexY][countX] == FIRE)
                {
                        turble = true;
                }
```

```
		if (map[judgeMap][indexY][countX] == HEART)
		{
			life += 300;
			if (life >= MAX_LIFE)life = MAX_LIFE;
			//bigState = true;
		}
		map[judgeMap][indexY][countX] = 0;
	}
	////////////////////////////////////////////////////////////////////////
	//判斷 CookieMan 範圍內的東西 6<X<10 為吃了可以加分數的
	//此狀態偵測碰撞到的是"障礙物"
	////////////////////////////////////////////////////////////////////////
	else if ((map[judgeMap][indexY][countX] > 5) && (map[judgeMap][indexY][countX]< 10))
	////////////////////////////////////////////////////////////////////////
	//>5 是障礙物且不是 turble 加速模式
	////////////////////////////////////////////////////////////////////////
	{
		if (turble == false && bigState == false) {
			if (map[judgeMap][indexY][countX]> 6) {
				life -= 150;
				return -1;
			}
else if ((map[judgeMap][indexY][countX] == 6) && map[judgeMap][((ty1 - MH) / MH) + i][countX] == 6)//
			{
				//life -= 150;
				return -1;
			}
		}
		////////////////////////////////////////////////////////////////////////
		//設定障礙物的特殊動畫
		//設定障礙物的特殊動畫
		////////////////////////////////////////////////////////////////////////
		else {
			if (map[judgeMap][indexY][countX] == GROUND_STOP_1)
			{
				effectMap[judgeMap][4][countX] = RUN_OUT;
				effectMap[judgeMap][5][countX] = RUN_OUT;
				effectMap[judgeMap][6][countX] = RUN_OUT;
```

```
                }
                if (map[judgeMap][indexY][countX] == TRAP_1)
                {
                        effectMap[judgeMap][6][countX] = RUN_OUT;
                }
                if (map[judgeMap][indexY][countX] == SKY_STOP_1)
                {
                        effectMap[judgeMap][0][countX] = RUN_OUT;
                        effectMap[judgeMap][1][countX] = RUN_OUT;
                        effectMap[judgeMap][2][countX] = RUN_OUT;
                        effectMap[judgeMap][3][countX] = RUN_OUT;
                        effectMap[judgeMap][4][countX] = RUN_OUT;
                        effectMap[judgeMap][5][countX] = RUN_OUT;
                }
                //effectMap[judgeMap][ty1 / MH + i][countX] = RUN_OUT;
        }
    }
    ////////////////////////////////////////////////////////////////////////
    //判斷 CookieMan 範圍內的東西 X=10 為吃了進入 bonustime
    //此狀態偵測 bonusTime
    ////////////////////////////////////////////////////////////////////////
    else if ((map[judgeMap][indexY][countX] == BONUSTIME))
    {
        map[judgeMap][indexY][countX] = 0;
        return -2;
    }
    else if (map[judgeMap][indexY][countX] == MAGNET)                    //測試進入磁鐵模式
    {
        magnetMode = 1;
        map[judgeMap][indexY][countX] = 0;
    }
    else if (map[judgeMap][indexY][countX] == BIG)                       //測試進入磁鐵模式
    {
        bigState = 1;
        map[judgeMap][indexY][countX] = 0;
    }
}
////////////////////////////////////////////////////////////////////////////////////////////////
```

```
		totalScore += hitTimes;
		return hitTimes;
	}
	bool CandyMap::HitRectangle(int x1, int y1, int x2, int y2, int tx1, int ty1, int tx2, int ty2)
	{
		return (tx2 >= x1 && tx1 <= x2 && ty2 >= y1 && ty1 <= y2);
	}
	int CandyMap::GetLifeNow() {
		return life;
	}
	void CandyMap::SetBigTrans() {
		if (bigState == true)bigState = false;
		else bigState = true;
	}
	void CandyMap::SetTurbleTrans() {
		if (turble == true)turble = false;
		else turble = true;
	}
	void CandyMap::SetMagnetTrans() {
		if (magnetMode == true)magnetMode = false;
		else magnetMode = true;
	}
	void CandyMap::SetQucikMove() {
		allDistance += 100;
	}
	void CandyMap::SetAddLife() {
		life += 200;
		if (life >= MAX_LIFE)life = MAX_LIFE;
	}
}
```

BackGround.h

```
namespace game_framework {
	/////////////////////////////////////////////////////////////////////////////
	// 這個 class 為遊戲背景:background
	/////////////////////////////////////////////////////////////////////////////
	class BackGround {
	public:
		BackGround();
```

```
        void OnShow();
        void LoadBitmap();
        void OnMove();
        void SetMoveSpeed(int speed);
        void MoveState(bool state);
        int choose;
        int status;
    private:
        CMovingBitmap sky_1, sky_2, bonus_1, bonus_2;
        CAnimation sky_shine;
        int index;
        bool moveSignal;
        const int MH, MW;                                                    //地圖高寬
        int moveDistance;
    };
}
```

BackGround.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "CookieMan.h"
#include "BackGround.h"
namespace game_framework
{
    ////////////////////////////////////////////////////////////////////////////
    // 這個 class 為糖果地圖:BackGround
    ////////////////////////////////////////////////////////////////////////////
    BackGround::BackGround() : MH(1200), MW(1920)
    {
        index = 0;
        choose = 0;
        int status = 1;
        moveDistance = 5;
        moveSignal = true;
    }
```

```
void BackGround::LoadBitmap()
{
	if (sky_1.isBitmapLoad() == false)
	{
		sky_1.LoadBitmap(IDB_sky_3);
		sky_2.LoadBitmap(IDB_sky_3);
		bonus_1.LoadBitmap(IDB_BONUSMAP1);
		bonus_2.LoadBitmap(IDB_BONUSMAP1);
		sky_shine.AddBitmap(IDB_skyShine_1);
		sky_shine.AddBitmap(IDB_skyShine_2);
		sky_shine.AddBitmap(IDB_skyShine_3);
		sky_shine.AddBitmap(IDB_skyShine_1);
	}
}
void BackGround::OnMove()
{
	////////////////////////////////////////////////////////////////////////
	//偵測遊戲人物是正在走路
	////////////////////////////////////////////////////////////////////////
	if (moveSignal == true)
	{
		index++;
	}
	else
	{
		index = index;
	}
	////////////////////////////////////////////////////////////////////////
	//移動的距離超過圖片的大小
	////////////////////////////////////////////////////////////////////////
	if (moveDistance * index >= MW)
	{
		index = 0;
	}
	sky_1.SetTopLeft(0 - moveDistance * index, SIZE_Y - MH);
	sky_2.SetTopLeft(MW - moveDistance * index, SIZE_Y - MH);
	bonus_1.SetTopLeft(0 - moveDistance * index, SIZE_Y - MH);
	bonus_2.SetTopLeft(bonus_1.Width() - moveDistance * index, SIZE_Y - MH);
```

```
	//////////////////////////////////////////////////////////////////////////
	//正常模式的圖片設定
	//////////////////////////////////////////////////////////////////////////
	if (status == 1)
	{
		sky_shine.SetDelayCount(10);
		sky_shine.OnMove();
		if (sky_shine.GetCurrentBitmapNumber() == 3)
		{
			status = 0;
		}
	}
}
void BackGround::OnShow()
{
	//正常模式的 show
	if (choose == 0)
	{
		sky_1.ShowBitmap();
		sky_2.ShowBitmap();
		if (status == 1)
		{
			sky_shine.OnShow();
		}
	}
	//bonustime 的 show
	if (choose == 1)
	{
		bonus_1.ShowBitmap();
		bonus_2.ShowBitmap();
	}
}
void BackGround::SetMoveSpeed(int speed)

	moveDistance = speed;
}
void BackGround::MoveState(bool state)
{
```

```
                moveSignal = state;
        }
}
```