國立臺北科技大學

2021 Spring 資工系物件導向程式實習

期末報告

Fireboy and Watergirl

第 8 組

108AEA001 李子健

108AEA008 王　翔

目錄

一、簡介

1. 動機

Fireboy and Watergirl 是一經典的闖關遊戲，難度適中，符合老師的要求。我們兩人在小時候都曾玩過這個遊戲，對遊戲內容與遊戲規則比較熟悉。我們認為可以借此機會練習寫 C++程式，加深對 OOP 概念的理解，因此我們選擇了這個遊戲。

2. 分工

王　翔：

規劃遊戲主體架構

設計地圖

設計程式邏輯

李子健：

處理遊戲所有素材

設計動畫與音樂

設計程式邏輯

二、遊戲介紹

1. 遊戲說明

Fireboy and Watergirl 是 2D 橫版闖關遊戲，包含兩個主角，需由兩名玩家分別控制兩個主角的移動，利用各種機關，從地圖的起點出發到達終點。期間，玩家需要避開地圖上的陷阱，並撿拾得分道具。兩位主角分別具有兩種屬性（fire 與 water），每位主角只能通過與自身屬性相符的地形，否則便會死亡。因此，遊戲需要兩位玩家一起合作，互相幫助，才能順利通關。遊戲內含 10 個關卡。

遊玩方式：

玩家 1，操控一位主角（Fireboy），使用←、↑、→按鍵使主角向左、向上、向右移動。

玩家 2，操控另一位主角（Watergirl），使用 A、W、D 按鍵使主角向左、向上、向右移動。

遊戲規則：

Fireboy：能通過地圖上的 Fire 屬性地形，而 water 與 toxic 地形會導致主角死亡。

Watergirl：能通過地圖上的 water 屬性地形，而 fire 與 toxic 地形會導致主角死亡。

終點：兩位主角分別到達與其屬性相同的門后，該關卡通關。

得分：兩位主角分別觸碰與其屬性相同的鑽石（diamond）后，加一分。

機關：遊戲中存在一些機關（詳見下方），玩家需利用這些機關穿過地圖。

特殊功能：

fireboy：主角之一，火屬性。在靜止、向上跳、向下落、向左、向右的狀態中各有 5 幀連續的動畫，跳起時有音效。

watergirl：主角之一，水屬性。 在靜止、向上跳、向下落、向左、向右的狀態中各有 5 幀連續的動畫，跳起時有音效。

fire pool：火屬性的地形，僅允許 fireboy 通過，會導致 watergirl 死亡。每格地形各有 15 幀連續的動畫，當主角通過地形時會發出音效。

water pool：水屬性的地形，僅允許 watergirl 通過，會導致 fireboy 死亡。每格地形各有 15 幀連續的動畫，當主角通過地形時會發出音效。

toxic pool：毒屬性地形，會導致兩位主角死亡。每格地形均有 15 幀連續的動畫。

鑽石（diamond）：fire diamond 與 water diamond，是遊戲中的得分道具。主角只能撿拾與其屬性相同的鑽石才能得分。主角撿拾鑽石會發出音效。

平臺（platform）：水平平臺與垂直平臺，主角可被平臺支撐，或被平臺阻擋道路。平臺通過按鈕或拉桿切換位置。

按鈕（button）：可以控制平台的位置，主角必須踩在按鈕上才能使其維持激活狀態。按鈕被踩下或放開時有動畫。

拉桿（stick）：可以控制平台的位置，主角通過觸碰拉杆切換激活與非激活的狀態。拉杆被切換時有動畫。

風扇（wind）：當主角站在風扇上時，可以跳的更高。風扇與吹風的特效有動畫。

門（door）：地圖的終點，兩位主角必須都抵達各自相應的終點才能過關。當

主角進入門或離開門時，門會展示開啓與關閉的動畫與音效。有兩種門分別對應兩個主角的屬性，每種門各有 22 幀連續的動畫。

箱子（box）：主角可以推動箱子，讓箱子壓住按鈕維持其激活的狀態，或是踩在箱子上跳往更高處。

密技：

小鍵盤上的“+”（加號）鍵可快速通關當前關卡，立刻跳至下一關卡。

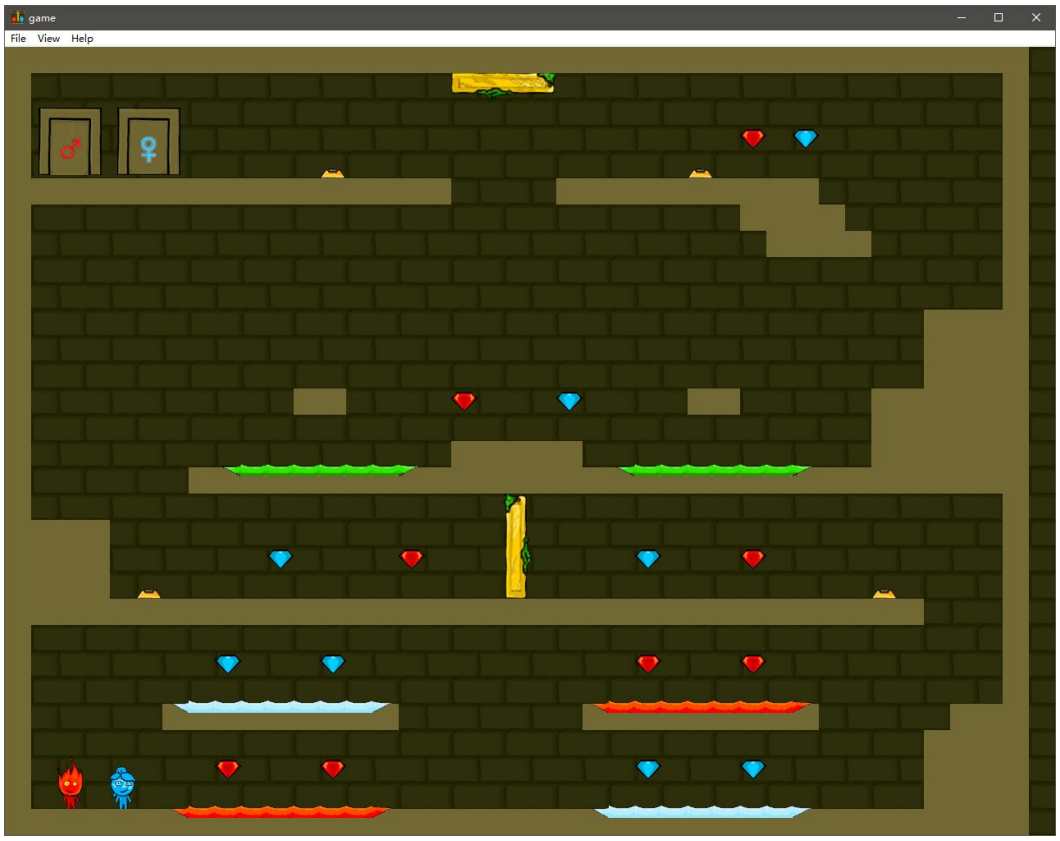
2. 遊戲圖形

標題畫面：可選擇開始遊戲，或者打開關於畫面

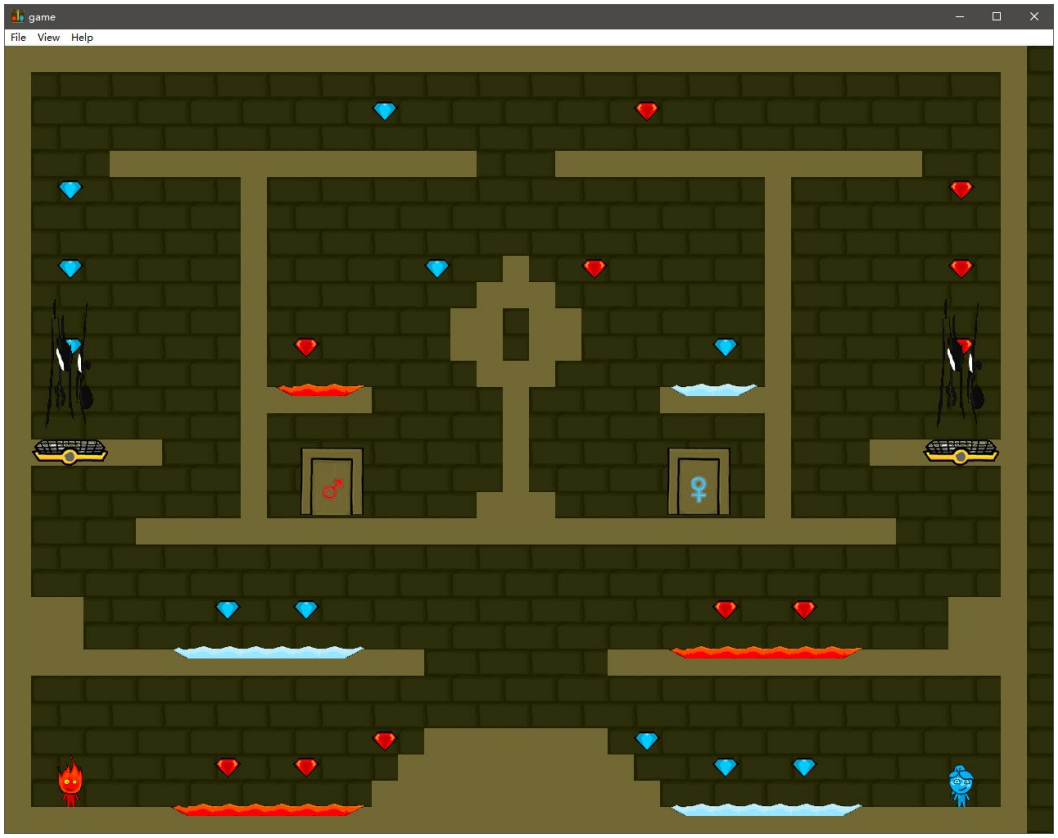
關於畫面：展示老師與同學的名字，可打開原始遊戲的頁面，或返回標題畫面

第一關：包含遊戲操作提示、三種地形、水平平臺、鑽石、按鈕、拉杆、門

第二關：包含三種地形、垂直平臺、水平平臺、鑽石、按鈕、拉杆、門



第三關：包含兩種地形、風扇、鑽石、門



過關選單：顯示兩位主角是否到達終點、是否得分

遊戲結束選單：主角死亡后顯示，讓玩家選擇是否重玩關卡，或開始新遊戲

箱子： 

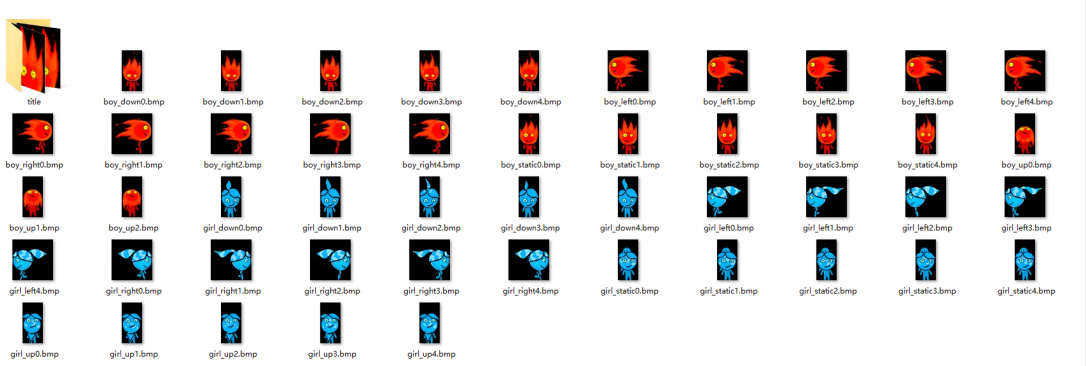
鑽石： 

平臺： 

開關： 

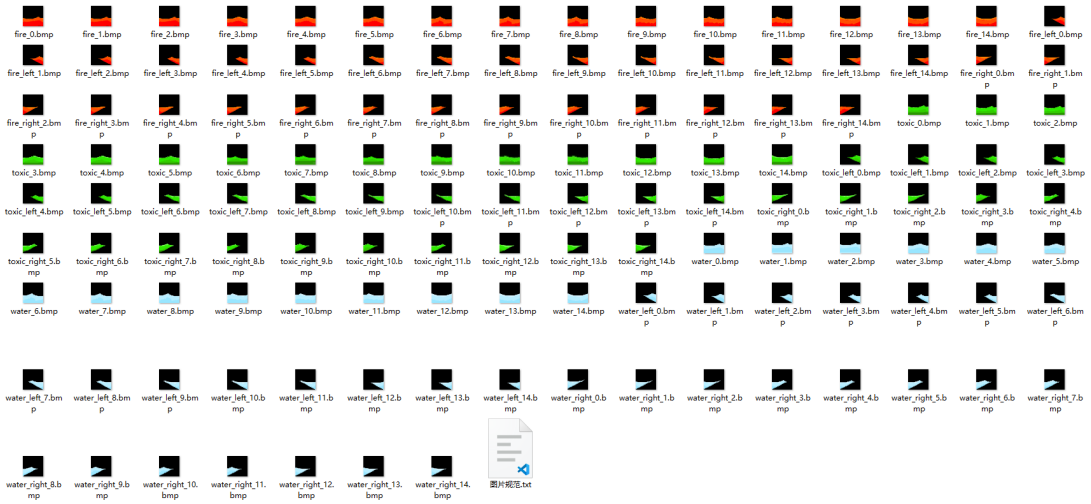
拉桿： 

人物動畫：靜止、下落、上跳、向左、向右



風扇動畫：

地形動畫：fire、water、toxic



開門與關門動畫：

自製關卡編輯器：

3. 遊戲音效

1.背景音樂

2.主角跳起音效

3.主角死亡音效

4.撿拾鑽石音效

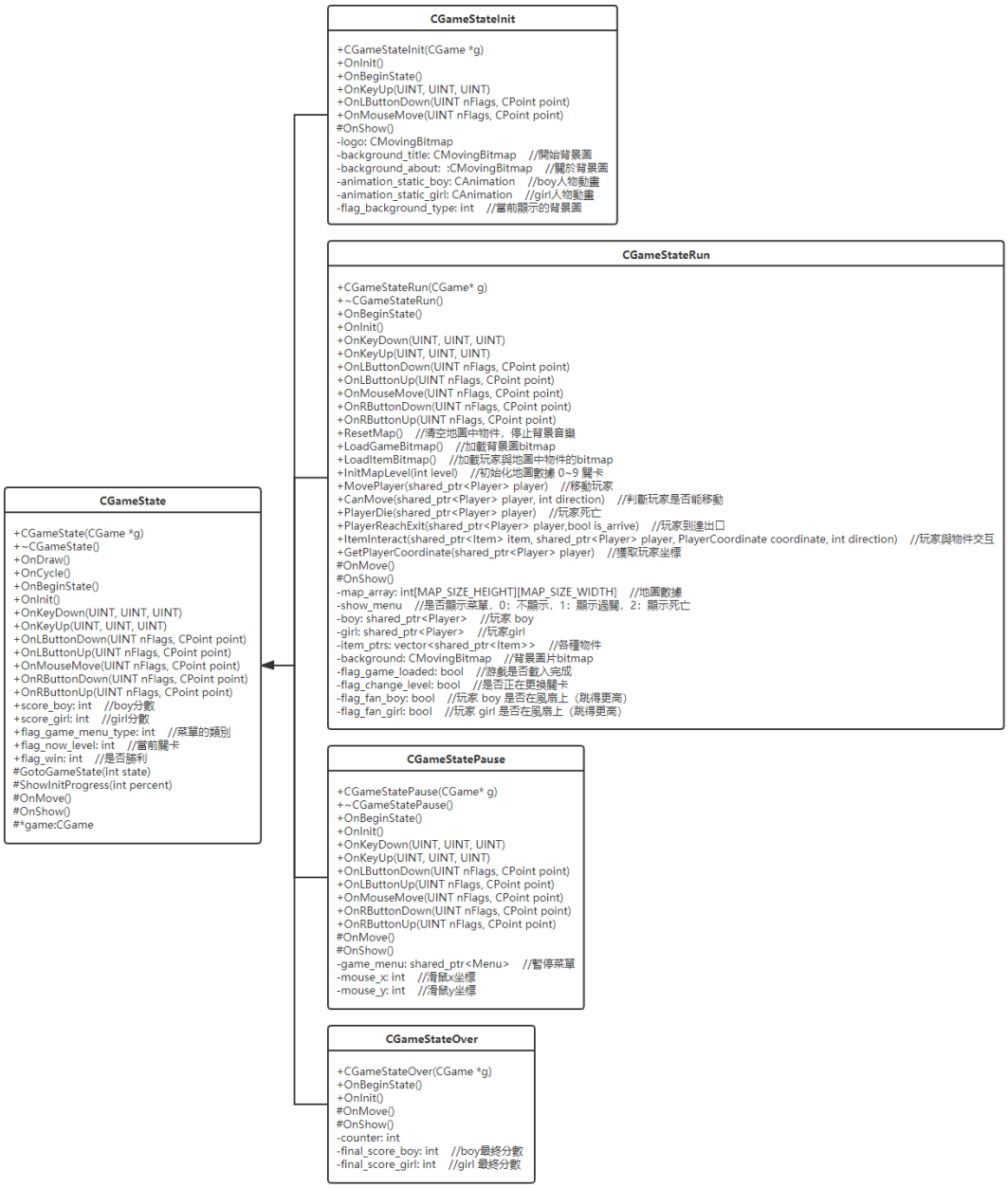
5.門開啓音效

6.門關閉音效

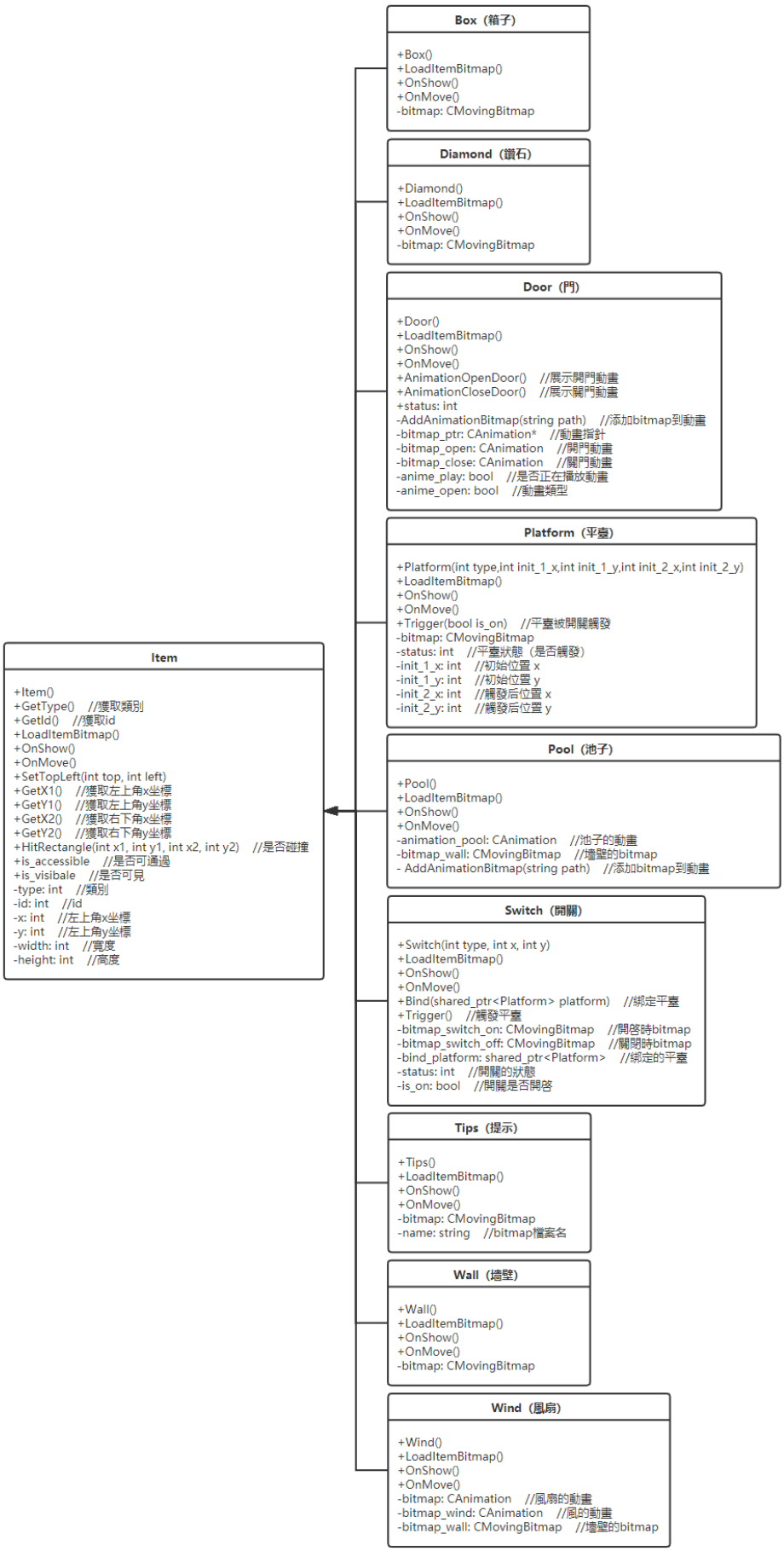
三、程式設計

1. 程式架構

CGameState:



Item：


Box（箱子）
+Box()
+LoadItemBitmap()
+OnShow()
+OnMove()
-bitmap: CMovingBitmap
Diamond（鑽石）
+Diamond()
+LoadItemBitmap()
+OnShow()
+OnMove()
-bitmap: CMovingBitmap
Door（門）
+Door()
+LoadItemBitmap()
+OnShow()
+OnMove()
+AnimationOpenDoor() //展示開門動畫
+AnimationCloseDoor() //展示關門動畫
+status: int
-AddAnimationBitmap(string path) //添加bitmap到動畫
-bitmap_ptr: CAnimation* //動畫指針
-bitmap_open: CAnimation //開門動畫
-bitmap_close: CAnimation //關門動畫
-anime_play: bool //是否正在播放動畫
-anime_open: bool //動畫類型
Platform（平臺）
+Platform(int type,int init_1_x,int init_1_y,int init_2_x,int init_2_y)
+LoadItemBitmap()
+OnShow()
+OnMove()
+Trigger(bool is_on) //平臺被開關觸發
-bitmap: CMovingBitmap
-status: int //平臺狀態（是否觸發）
-init_1_x: int //初始位置 x
-init_1_y: int //初始位置 y
-init_2_x: int //觸發后位置 x
-init_2_y: int //觸發后位置 y
Item
+Item()
+GetType() //獲取類別
+GetId() //獲取id
+LoadItemBitmap()
+OnShow()
+OnMove()
+SetTopLeft(int top, int left)
+GetX1() //獲取左上角x坐標
+GetY1() //獲取左上角y坐標
+GetX2() //獲取右下角x坐標
+GetY2() //獲取右下角y坐標
+HitRectangle(int x1, int y1, int x2, int y2) //是否碰撞
+is_accessible //是否可通過
+is_visibale //是否可見
-type: int //類別
-id: int //id
-x: int //左上角x坐標
-y: int //左上角y坐標
-width: int //寬度
-height: int //高度
Pool（池子）
+Pool()
+LoadItemBitmap()
+OnShow()
+OnMove()
-animation_pool: CAnimation //池子的動畫
-bitmap_wall: CMovingBitmap //墻壁的bitmap
- AddAnimationBitmap(string path) //添加bitmap到動畫
Switch（開關）
+Switch(int type, int x, int y)
+LoadItemBitmap()
+OnShow()
+OnMove()
+Bind(shared_ptr<Platform> platform) //綁定平臺
+Trigger() //觸發平臺
-bitmap_switch_on: CMovingBitmap //開啓時bitmap
-bitmap_switch_off: CMovingBitmap //關閉時bitmap
-bind_platform: shared_ptr<Platform> //綁定的平臺
-status: int //開關的狀態
-is_on: bool //開關是否開啓
Tips（提示）
+Tips()
+LoadItemBitmap()
+OnShow()
+OnMove()
-bitmap: CMovingBitmap
-name: string //bitmap檔案名
Wall（墻壁）
+Wall()
+LoadItemBitmap()
+OnShow()
+OnMove()
-bitmap: CMovingBitmap
Wind（風扇）
+Wind()
+LoadItemBitmap()
+OnShow()
+OnMove()
-bitmap: CAnimation //風扇的動畫
-bitmap_wind: CAnimation //風的動畫
-bitmap_wall: CMovingBitmap //墻壁的bitmap

Other Class：

Player

+Player(bool boy)　//是否是boy
+~Player()
+LoadBitmapPlayer()
+OnMove()
+OnMove(CGameStateRun* game)
+OnShow()
+GetX1()　//獲取左上角 x 坐標
+GetY1()　//獲取左上角 y 坐標
+GetX2()　//獲取右下角 x 坐標
+GetY2()　//獲取右下角 y 坐標
+GetVelocity()　//獲取坐標
+GetStep()　//獲取步長
+SetTopLeft(int top, int left)　//設定左上角坐標
+SetVerticalState(int state)　//設置垂直狀態（上、下、靜止）
+SetHorizontalState(int state)　//設置水平狀態（左，右、靜止）
+GetVerticalState()　//獲取垂直狀態（上、下、靜止）
+GetHorizontalState()　//獲取水平狀態（左，右、靜止）
+HitRectangle(int tx1, int ty1, int tx2, int ty2)　//碰撞檢測
+is_boy: bool　//是否是boy
+is_visible: bool　// 是否可見（死亡後不可見）
+is_die: bool　//是否死亡
+reach_exit: bool　//是否到達出口
+x: int　//左上角x坐標
+y: int　//左上角y坐標
+moving_vertical: int　//垂直移動狀態（上、下、靜止）
+moving_horizontal: int　//水平移動狀態（上、下、靜止）
+velocity: int　//目前的速度（pixel/frame）
+score: int　//分數
+GetStatic(): CAnimation &　//獲取靜止動畫
+GetLeft() CAnimation &　//獲取向左動畫
+GetRight() CAnimation &　//獲取向右動畫
+GetUp() CAnimation &　//獲取向上動畫
+GetDown() CAnimation &　//獲取向下動畫
+SetAni(CAnimation& address)　//設置動畫
-animation_static: CAnimation　//禁止動畫
-animation_left: CAnimation　//向左動畫
-animation_right: CAnimation　//向右動畫
-animation_up: CAnimation　//向上動畫
-animation_down: CAnimation　//向下動畫
-charactor_ani : CAnimation *　//當前動畫指針

Menu

+Menu()
+LoadItemBitmap()
+OnShow(int menuType)　//顯示指定類別的選單
+OnMove(int menuType)　//顯示指定類別的選單
-bitmap_pause0: CMovingBitmap　//玩家未得分之選單背景
-bitmap_pause1: CMovingBitmap　//玩家有得分之選單背景
-bitmap_over: CMovingBitmap　//游戲結束之選單背景
-top: int
-left: int

2. 程式類別

類別名稱	.h 檔行數	.cpp 檔行數	說明
CGameStateInit	19	146	遊戲標題畫面與關於畫面
CGameStateOver	15	67	遊戲結束畫面
CGameStateRun	107	1621	遊戲主要邏輯
CGameStatePause	32	154	遊戲的選單畫面
Menu	20	49	選單
Player	56	188	主角
Item	32	57	物件（基類）
Box	17	35	箱子
Diamond	17	51	鑽石
Door	42	126	門
Platform	30	87	平臺
Pool	21	133	地形（火、水、毒）
Switch	37	74	開關
Tips	19	44	提示
Wall	17	36	牆壁
Wind	19	60	風扇
總行數	500	2928	

3. 程式技術

地圖：

每一個關卡都分別使用一個 int[30][40]陣列存儲其地圖數據，每個數值代表一格區塊，每格區塊為長寬 32px 的正方形。陣列內的編號表示地圖中的物件，每個物件可使用一個或多個區塊。

地圖數據定義如下：

編號	類別	說明	大小（寬 x 高）	是否可通過
0	None	空白	1x1	是
1	Wall	墻壁	1x1	否
2	Fire Pool \	火屬性地形（下坡）	1x1	否
3	Fire Pool /	火屬性地形（上坡）	1x1	否
4	Water Pool \	水屬性地形（下坡）	1x1	否
5	Water Pool /	水屬性地形（上坡）	1x1	否
6	Toxic Pool \	毒屬性地形（下坡）	1x1	否
7	Toxic Pool /	毒屬性地形（上坡）	1x1	否
100	Fireboy	主角（火）	1x2	是
101	Watergirl	主角（水）	1x2	是
200	Fire Diamond	鑽石（火）	1x1	是
201	Water Diamond	鑽石（水）	1x1	是
300	Fire Pool	火屬性地形	1x1	否
301	Water Pool	水屬性地形	1x1	否
302	Toxic Pool	毒屬性地形	1x1	否

400	Fire Door	終點（火）	3x3	是
401	Water Door	終點（水）	3x3	是
500	Switch（Button）	開關（按鈕）	1x1	是
501	Switch（Stick）	開關（拉桿）	1x1	是
502	Platform（Horizon）	平臺（水平）	4x1	否
503	Platform（Vertical）	平臺（垂直）	1x4	否
600	Wind	風扇	1x4	否
601	Box	箱子	1x1	否
1000	Tips	遊戲提示	N/A	是

物件：

在地圖中，除了主角以外的所有物件都繼承自父類 Item，采用統一的介面以便處理。在地圖載入時，程式會讀取地圖陣列，通過工廠方法模式建立物件對象，並加入至 CGameStateRun 的 vector 中。所有的物件均采用智慧型指標（shared_ptr）以防止產生 memory leak，在主角通關進入下一關前，vector 中的所有物件會被釋放。

由於 game framework 自身限制（在下一章節中詳細說明），物件本身僅存儲自己的屬性（包括 id、類別、大小、位置、可通行性，等）、圖片、動畫與音效，主角與物件的交互邏輯由 CGameStateRun 中的程式完成。

機關：

機關是一種比較特殊的物件，雖然同為繼承自 Item，但其中包含的屬性更多，無法從地圖陣列中直接載入，需要額外的方法初始化。例如：

平臺：會被開關切換位置，因此建構函式中包含兩個座標。

開關：需綁定相應的平臺，因此在建構時需傳入一個指向平臺的 pointer。

主角：

遊戲中包含兩個不同的主角，使用同一類別，在初始化時通過不同的 flag 區分。為了使主角靈活移動，我們使用兩個變數分別表示人物在水平與垂直方向上的狀態：

（1）水平方向上的狀態：向左、向右、靜止。

（2）垂直方向上的狀態：向上、向下、靜止。

這兩個變數相互獨立，因此主角在水平方向上的運動狀態與垂直方向上的運動狀態無關。我們為主角在各個方向上的移動均繪製了動畫，並在跳躍時播放音效。

主角移動邏輯：

（1）當主角在水平方向上移動時，程式會先判斷當前方向上是否可通行，若可通行則主角每次前進 4 px。

（2）當主角跳起時，程式會首先判斷上方是否可通行，若可通行則給予主角一個向上的初始速度（16 px/cycle），並給予主角一個向下的加速度（1 px/cycle）作為重力。為防止下落過快，當主角下落速度達到閾值（5 px/cycle）時不再加速。

（3）當主角在垂直方向上處於靜止狀態時，程式會判斷主角下方的地面是否可同行，若可通行則給予玩家向下的加速度（同上）。

素材：

在原版遊戲中，各部分的素材被拼接成一整張大圖片，通過其使用的遊戲框架讀取素材對應的 json 檔案自動裁切。我們用 python 設計了一個工具，通過讀取 json 檔案，自動獲取在大圖中素材的大小與位置，自動裁切，自動將 png 格式的原圖轉換為 bmp 檔案。同時，根據地圖中區塊的大小（每格寬高 32px），我們將遊戲中的所有素材均縮放至 32 px 的整數倍大小，並採用適當的方式修補縮放導致的變型。通過工具的幫助，我們總共裁切了約 300 張圖片素材。

自製關卡編輯器：

我們用.net 配合 winform 技術設計了一個關卡編輯器，通過將原版遊戲畫面截圖載入編輯器，我們可以很方便地使用可視化的方式設計關卡布局，並將設計出的關卡快速導出至遊戲可讀取的格式（int 類別 2D 陣列）。我們決定在課程結束後將此工具開源，並上傳至 github。

四、結語

1. 問題及解決方法

在遊戲的製作過程中，我們遇到了大量的問題，每週都花費了相當多的時間與精力，相互討論，並研究如何解決。Game Framework 對我們來說是一個完全陌生的程式，我們只能一點點地修改內置的 demo，探索如何使用這個框架。

首先，我們完全不知道應該如何構建地圖。在閱讀了陳偉凱老師寫的“遊戲地圖與座標系統概論”後，我們才知道應該把地圖分割為 2D 陣列，用編碼來表示是否有障礙物。我們注意到，原版遊戲中地圖內的元素非常多，而且地形十分複雜，如果一個個填進陣列裏將是十分浩大的工程。於是，我們用.net 做了一個可視化的關卡編輯器：首先載入遊戲畫面截圖，再將網格繪製在圖片上，根據滑鼠在圖片上的坐標除以格子的寬度，算出滑鼠的坐標；當滑鼠點擊某格時，修改 2D 陣列中相應位置上的編碼；最後，將 2D 陣列轉換為 c++的陣列格式複製到剪貼板。我們自製的關卡編輯器大大加快了遊戲開發進程，使得我們不必將大量時間花費在單調且重複的工作上。

當我們開始在地圖中引入遊戲角色，新的問題出現了。原本我們的角色移動方式是，判斷地圖的 2D 陣列中，角色的前進方向上是否有阻礙。然而，如果角色按照格子來移動，但在遊戲畫面上以畫素點的坐標顯示，這兩者之間需要轉換。如果玩家只向前進入某個格子的邊界，是否應該判定該玩家完全進入這一格呢？我們應該用哪一個畫素點當作玩家在畫面上的坐標呢？這個問題困擾了我們兩週時間，經過反覆實驗和討論，我們最終決定採用 AABB 碰撞檢測（Axis-aligned Bounding Box）。簡單來說，就是用矩形把物體包起來，檢查矩形之間是否發生碰撞。從這時起，我們也發現在 demo 中的物件寫法並不適合我們的遊戲，我們

必須將地圖中的所有元素視作統一的物件，包括牆壁、機關、道具，等。所有的物件都繼承於一個基類，並採用統一的介面封裝起來。

當角色在熒幕上移動起來時，新的問題出現了：有時候角色會卡進牆裡。這個問題又困擾了我們許久，直到我們注意到 game gramework 是採用 game cycle 的方式運作，即每隔一段時間（1/30 秒，即 33 毫秒）運作一次，並不是即時的。因此，程式需要預先判斷角色前方一段距離是否可通行，才能向前移動一次。並且，角色在上升或下落時也需考慮到。

當我們做完第一關，開始做第二關時，又出現了問題。之前我們向老師請教過關於 CGameState 與遊戲關卡的關係，老師說遊戲運行的邏輯必須全部在 CGameStateRun 內，而不是一個關卡一個 CGameState，所以我們也是這麼做的。但我們發現，如果想要換關卡，必須以某個事件觸發，這會引發一些問題。我們的設定是，當兩個主角都抵達地圖終點時，視作該關卡完成。因此，遊戲角色與地圖終點完全重合，便可觸發遊戲換關。換關卡的邏輯由地圖中的物件觸發 CGameStateRun 中的函式，並在下一關卡加載前清空當前的遊戲地圖並加載新的地圖。當 CGameStateRun 中的函式執行完畢並，會返回到地圖物件的邏輯處理中，但此時物件已經被清除，這會導致 pointer 指向程式中未知的區域導致程式宕掉。由於 Game Framework 只提供了顯示 bitmap 與播放音樂的功能，並未提供事件綁定的功能，所有的邏輯都需要我們自己實現。最後，我們只能讓主程式觸發關卡切換，並建立 flag 使遊戲在關卡切換完成前跳過碰撞處理。由此一來，物件中所有的邏輯與事件都需放進 CGameStateRun 中，物件本身只存放自身的屬性（包括 id、類別、大小、位置、可通行性，等）、圖片、動畫與音效。這導致我們的 CGameStateRun 的檔案增長到 1600 行，但我們已儘可能保證程式碼可讀

性，加入大量注釋，並將單個 cpp 檔案依照其中的函式功能分割為數個檔案。

完成了遊戲的換關邏輯後，不經意間發現的一個小 bug 讓我們警覺起來：當關卡切換後，遊戲角色下方的地圖區塊會憑空消失，且每次消失的數量與位置完全隨機；更奇怪的是，在不同的同學電腦上，地圖區塊消失的大致區域不同；在我的電腦上，地圖左下方的區塊會消失，而在李子健同學的電腦上，地圖右下方的區塊會消失。在經過反覆的調試依然無法解決後，我們猜到這個 bug 可能是換關時，存儲物件 pointer 的 vector 沒有徹底清空，導致 pointer 指向了錯誤的記憶體。然而，清空 pointer 類別的 vector 在 c++中似乎並不容易，甚至會讓程式宕掉。於是，我們將所有用到 pointer 的地方全部改為智慧型指標（shared_ptr）。不僅解決了這個 bug，也讓我們的程式再也沒有出現過 memory leak。

在遊戲素材的處理上，我們也經歷了不少波折。在原版遊戲中，圖片素材以 png 格式呈現，並有透明背景。由於素材的邊緣有輕微的過渡色，雖然肉眼難以區分，但將其轉換為 bmp 格式並用 game framework 載入，並設定白色為透明通道，會發現素材的周圍有一圈極薄的淺色輪廓。如果要一個個圖片，pixel by pixel 地修圖，那將是極大的工作量。因此我們改為設定黑色為透明通道，效果意外地不錯。

在原版遊戲中，各張素材被拼接為一整張大圖，且素材大小與位置及不規律，程式無法直接使用，找尋畫素邊界與切圖也很麻煩。但我們發現，原版遊戲在 json 檔案中存儲了每張素材的原始大小與其在大圖中的絕對坐標。因此，我們用 python 開發了一個小工具，自動讀取素材圖片與其對應的 json 檔案，自動分割圖片並轉化為黑色背景的 bnp 檔案。該工具極大地提升了我們的工作效率，使我們能實現原版遊戲中的幾乎所有動畫。我們也為素材檔案命名方式、素材拉伸裁

切方式等統一了團隊標準，方便協同合作。

除此以外，我們還發現了 game framework 中的一些限制。原本，我們將遊戲設定為 60fps 運行，即在 gamelib 中設定為每 17ms 運行一次 game cycle。但這會導致 CMovingBitmap 出現 bug：動畫無法使用 SetDelayCount 方法調速，無論設定為多少，動畫均按照預設的固定倍速播放，只有將遊戲改回 30fps 才恢復正常。可能是 game framework 從設計之初就沒有考慮到遊戲會以其他幀率運行。另外，遊戲的解析度必須設定為標準解析度才能全屏，這可能是 DirectX API 中帶來的限制。

2. 時間表

週次	組員 A	組員 B	說明
1	5	4	尋找原型遊戲
2	5	5	製作素材
3	5	5	完成 git 練習，設計關卡，製作素材
4	20	14	設計關卡，設計角色移動，製作素材
5	19	16	添加機關，設計角色移動，製作素材
6	16	9	添加機關，設計角色與物件交互，製作素材
7	15	12	完成第一關並在期中報告中展示
8	15	15	根據報告建議優化角色移動，添加角色移動動畫
9	8	10	添加地圖與機關動畫，製作素材，添加音效
10	10	10	設計關卡，製作素材，添加音效
11	8	8	設計關卡，製作素材，添加音效
12	8	6	完成十個關卡並在期中報告中展示
13	8	8	添加遊戲選單，關於畫面，添加密技
14	5	5	添加通關畫面，game over 畫面
15	6	6	測試遊戲
16	8	8	封裝安裝檔
17	8	5	撰寫報告
總計	169	146	

3. 貢獻比例

王翔：50%

李子健：50%

4. 自我檢核表

序號	項目	完成否	無法完成原因
1	解決 Memory leak	☑已完成 ☐未完成	
2	自定遊戲 Icon	☑已完成 ☐未完成	
3	有 About 畫面	☑已完成 ☐未完成	
4	初始化面說明按鍵及滑鼠	☑已完成 ☐未完成	
5	之用法與密技	☑已完成 ☐未完成	
6	上傳 setup/apk/source 檔	☑已完成 ☐未完成	
7	setup 檔可正確執行	☑已完成 ☐未完成	
8	報告字型、點數、對齊、行距	☑已完成 ☐未完成	
9	頁碼等格式正確	☑已完成 ☐未完成	
10	全熒幕啟動-改列加分項	☑已完成 ☐未完成	

5. 收獲

（1）王翔

經過本次程式設計實習課程，我對 OOP 的概念有了更深入的理解。在遊戲畫面上的所有物件都應被視作統一的類別，即它們都繼承與同一個基類。雖然不同的的物件可以實現不同的功能，但他們都應該有相同的基本功能：LoadBitmap、OnMove、OnShow、物大小、是否可見、是否可通過、主角與物件接觸時產生的事件，等。這樣才能將所有物件放進同一個容器內，程式在每個 game cycle 中只需遍歷一次就能訪問所有物件。

由於遊戲地圖上的物件種類繁多，不可能用手動的方式一個個 new 出來，我使用了 design pattern 中的 Factory method pattern 從 2D 陣列中自動讀取並建立物件，2D 陣列則由我編寫的關卡編輯器設計並產生。

在本課程的學習中，我也認識到團隊協作的重要性。團隊成員間必須經過充分溝通，在程式碼風格、函式命名、檔案命名、程式業務邏輯等方面達成共識，並用文檔記錄下程式開發規範，才能使團隊協作更有效率。正確地使用分散式版本控制軟體（Git）也很重要，每當添加新功能時都要寫明注釋，如果出現問題可以快速回滾，並根據 commit 記錄縮小需除錯的程式碼範圍。

（2）李子健

通過這學期的這一門課程，讓我在物件導向程式設計的技術更熟練，以往的課程只會寫一些很小的作業，沒有聯繫。而在這門課裡一學期就做一個專案，需要很多時間和精力去進行分析和設計，並進行排錯。

在這門課當中我負責的是一些動作邏輯的部分和音樂、素材的設計，比如碰撞檢測。一開始實作的方式只是檢測玩家物件加一段距離是否會碰撞，後面發現這樣實作出來的效果很差，有時候會卡在牆裡，後面我們尋求解決辦法，最好採用了 AABB 碰撞盒算法，來判斷是否會碰撞，這個問題就完美解決了。

一開始做素材的時候，我們選擇了白底作為背景，但是可能是因為 bmp 的特性，很多圖片保存為 bmp 格式後，那個白色都不是純白，所以顯示在遊戲裡面會有毛邊，後面我突發奇想，想到用黑底試試，結果效果好非常多！於是後面的素材我們就採用黑底作為背景。

在添加動畫的時候遇到了 memory leak 的問題，我們嘗試了很多方法都沒找到 leak 的位置在哪，後面通過智慧型指標解決了，這鍛煉了我的 debug 能力。

我還熟悉了 Git 的用法，之前只是偶爾用過 GitHub，並沒有很長時間地用 Git，並且我了解到 commit 的信息很重要，因為這可以讓小組成員清楚地知道我更新了什麼地方。

總的來說，這門課讓我受益良多，讓我增進了程式設計的能力和與他人協作的能力，還鍛煉了我的耐心（有時候遇到 bug 會很煩）。

6. 心得、感想

（1）王翔

這門課是我在本學期花費時間與精力最多的一門課。自己從頭開始設計出一款遊戲是我一直以來想做的，但由於課業原因，我一直沒有時間去熟悉一款遊戲框架並掌握其用法。感謝這門課程讓我有機會去實作一個完整的遊戲。game framework 雖然只是一款以教育為目的的遊戲框架，但對我來說是一個好的開始。它讓我有了編寫遊戲的基本概念，包括 OOP、game cycle、事件驅動、按鍵處理、素材處理、邏輯處理等。從一開始空白的遊戲畫面，漸漸地加上地圖，遊戲角色；再到讓角色動起來，與地圖上的各種元素交互；最後形成一個完整的遊戲。這個過程讓我相當有成就感，讓我第一次覺得程式開發可以是這麼有趣。每週我都投入了大量時間與精力到遊戲開發中，但我覺得這是值得的。在將來，假如還有機會，我會開發更多的遊戲，並移植到各種平臺上，甚至是遊戲機中。

最後，非常感謝李子健同學願意與我一起合作編寫遊戲，也非常感謝老師與助教爲我們提供的指導與幫助。沒有你們的付出，我無法一個人完成如此龐大的工程。這門課是我大學生涯中的最後一門課，也是讓我印象最深刻的一門課。

（2）李子健

這門課讓我了解到如何通過一個 game framework 去製作一個遊戲。實不相瞞，我的程式設計的啟蒙就是遊戲，在十年前我在進行精靈寶可夢的 GBA 客製化，那時候通過歐洲的一些大神寫出來的工具去寫腳本，讓遊戲裡面觸發不同劇情。但那時候實在能力不足，本來有一個偉大計劃就是製作出有自己世界的寶可夢遊戲，但是實在無心力去製作。這學期修了這門課，讓我了解到製作一個遊戲絕對不是一件容易的事情，因為製作一個遊戲，比如這個遊戲是什麼、這個遊戲各種物件的 data model 怎麼設計、怎麼去處理動作邏輯、怎麼去判斷人物碰撞、怎麼讓人物觸發不同劇情……很多很多的問題，絕對不是容易的事！這學期從一開始定題目、收集素材、製作地圖、加入人物、實作動作邏輯……一步一步地走到最後形成一個完整的遊戲，我的感覺是非常自豪的。在解決出現的 bug 之後，會感覺非常爽，就好像運動完有多巴胺分泌一樣，這大概就是寫代碼的魅力吧！

很感謝王翔同學和我一組，我們兩個人協作一起完成這個遊戲。感謝老師和助教，在我們遇到問題的時候給我們提供解答。沒有你們，這個遊戲不會成型。

這是一門令人印象深刻的課！

7. 對於本課程的建議

1.希望能開放同學選擇的遊戲類別

我們原本打算製作 RPG 遊戲，但老師將遊戲的類別限定為闖關類，即必須有關卡與過關。我們能理解老師可能是為了方便遊戲演示，或是方便評分，但我們依然非常想做 RPG 遊戲，這可以說是我們的一大遺憾。

RPG，即角色扮演遊戲（Role-Playing Game）。在遊戲中，玩家扮演虛擬世界中的一個或者幾個角色進行遊戲，玩家通過操控遊戲角色與敵人戰鬥，提升等級、收集裝備和完成遊戲設置的任務，並體驗劇情。通常這類遊戲都是由玩家扮演角色在遊戲世界中漫遊，而一路上的各種遭遇（如戰鬥、交談、會見重要人物等）則是玩家人物成長及遊戲進行的重要關鍵所在。

我們認為，做 RPG 遊戲同樣可以讓同學們學到很多內容，並且其製作難度並不亞於闖關遊戲。

2.希望對 game framework 中的限制與 bug 作出說明

（1）游戏帧率必須固定為 30fps，否則動畫播放速度會被拖慢，且無法調整

（2）GotoGameState 方法只能從 CGameState 內呼叫，因此遊戲中所有的邏輯處理都必須在 CGameState 內完成。

五、附錄

1. mygame.h

```
/*
 * mygame.h: 本檔案儲遊戲本身的 class 的 interface
 * Copyright (C) 2002-2008 Woei-Kae Chen <wkc@csie.ntut.edu.tw>
 *
 * This file is part of game, a free game development framework for windows.
 *
 * game is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * game is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 *   2004-03-02 V4.0
 *      1. Add CGameStateInit, CGameStateRun, and CGameStateOver to
 *         demonstrate the use of states.
 *   2005-09-13
 *      Rewrite the codes for CBall and CEraser.
 *   2005-09-20 V4.2Beta1.
 *   2005-09-29 V4.2Beta2.
 *   2006-02-08 V4.2
 *      1. Rename OnInitialUpdate() -> OnInit().
 *      2. Replace AUDIO_CANYON as AUDIO_NTUT.
 *      3. Add help bitmap to CGameStateRun.
 *   2006-09-09 V4.3
 *      1. Rename Move() and Show() as OnMove and OnShow() to emphasize that they are
 *         event driven.
 *   2008-02-15 V4.4
 *      1. Add namespace game_framework.
 *      2. Replace the demonstration of animation as a new bouncing ball.
 *      3. Use ShowInitProgress(percent) to display loading progress.
*/
#include "CEraser.h"
#include "CBall.h"
#include "CBouncingBall.h"
//#include "Map.h"
namespace game_framework {
    /////////////////////////////////////////////////////////////////////////////
    // Constants
    /////////////////////////////////////////////////////////////////////////////
    /////////////////////////////////////////////////////////////////////////////
    // 這個 class 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;                             // csie 的 logo
        CMovingBitmap background_title;                 // 冰火人開始背景
        CMovingBitmap background_about;                 // 冰火人關於背景
```

```
        CAnimation animation_static_boy;
        CAnimation animation_static_girl;
        int flag_background_type = 0;                           //0: title, 1:about
    };
    /////////////////////////////////////////////////////////////////////////////
    //// 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
    //protected:
    //    void OnMove();                                          // 移動遊戲元素
    //    void OnShow();                                          // 顯示這個狀態的遊戲畫面
    //private:
    //    const int        NUMBALLS; // 球的總數
    //    CMovingBitmap    background;    // 背景圖
    //    CMovingBitmap    help;        // 說明圖
    //    CBall            *ball;            // 球的陣列
    //    CMovingBitmap    corner;          // 角落圖
    //    CEraser          eraser;          // 拍子
    //    CInteger         hits_left; // 剩下的撞擊數
    //    CBouncingBall    bball;        // 反覆彈跳的球
    //    ////////////////////////////////////////////////
    //    //int                map_now_level;//游戏目前关卡，0~9
    //    //vector<Map*>       maps;           //游戏地图
    //    //Player*            player_boy;     //boy
    //    //Player*            player_girl;//girl
    //    Map                *map;//地图逻辑
    //};
    /////////////////////////////////////////////////////////////////////////////
    // 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);
        void OnBeginState();                                  // 設定每次重玩所需的變數
        void OnInit();
    protected:
        void OnMove();                                          // 移動遊戲元素
        void OnShow();                                          // 顯示這個狀態的遊戲畫面
    private:
        int counter;     // 倒數之計數器
        //最终结算分数
        int final_score_boy = 0;
        int filal_score_girl = 0;
    };
}
```

2. mygame.cpp

```
/*
 * mygame.cpp: 本檔案儲遊戲本身的 class 的 implementation
 * Copyright (C) 2002-2008 Woei-Kae Chen <wkc@csie.ntut.edu.tw>
 *
 * This file is part of game, a free game development framework for windows.
 *
 * game is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
```

```
 * the Free Software Foundation; either version 2 of the License, or
 * (at your option) any later version.
 *
 * game is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 *
 * History:
 *   2002-03-04 V3.1
 *           Add codes to demostrate the use of CMovingBitmap::ShowBitmap(CMovingBitmap &).
 *   2004-03-02 V4.0
 *      1. Add CGameStateInit, CGameStateRun, and CGameStateOver to
 *         demonstrate the use of states.
 *      2. Demo the use of CInteger in CGameStateRun.
 *   2005-09-13
 *      Rewrite the codes for CBall and CEraser.
 *   2005-09-20 V4.2Beta1.
 *   2005-09-29 V4.2Beta2.
 *      1. Add codes to display IDC_GAMECURSOR in GameStateRun.
 *   2006-02-08 V4.2
 *      1. Revise sample screens to display in English only.
 *      2. Add code in CGameStateInit to demo the use of PostQuitMessage().
 *      3. Rename OnInitialUpdate() -> OnInit().
 *      4. Fix the bug that OnBeginState() of GameStateInit is not called.
 *      5. Replace AUDIO_CANYON as AUDIO_NTUT.
 *      6. Add help bitmap to CGameStateRun.
 *   2006-09-09 V4.3
 *      1. Rename Move() and Show() as OnMove and OnShow() to emphasize that they are
 *         event driven.
 *   2006-12-30
 *      1. Bug fix: fix a memory leak problem by replacing PostQuitMessage(0) as
 *         PostMessage(AfxGetMainWnd()->m_hWnd, WM_CLOSE,0,0).
 *   2008-02-15 V4.4
 *      1. Add namespace game_framework.
 *      2. Replace the demonstration of animation as a new bouncing ball.
 *      3. Use ShowInitProgress(percent) to display loading progress.
 *   2010-03-23 V4.6
 *      1. Demo MP3 support: use lake.mp3 to replace lake.wav.
*/
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "mygame.h"
#include <windows.h>
namespace game_framework {
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的遊戲開頭畫面物件
/////////////////////////////////////////////////////////////////////////////
CGameStateInit::CGameStateInit(CGame *g)
: CGameState(g)
{
}
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_BACKGROUND);
	background_title.LoadBitmap("RES\\title\\title.bmp");
	background_about.LoadBitmap("RES\\title\\about.bmp");
	//start_text.LoadBitmapA("RES\\STARTTEXT.bmp", RGB(255, 255, 255));
	animation_static_boy.AddBitmap("RES\\player\\title\\boy_static0.bmp", RGB(0, 0, 0));
	animation_static_boy.AddBitmap("RES\\player\\title\\boy_static1.bmp", RGB(0, 0, 0));
	animation_static_boy.AddBitmap("RES\\player\\title\\boy_static2.bmp", RGB(0, 0, 0));
	animation_static_boy.AddBitmap("RES\\player\\title\\boy_static3.bmp", RGB(0, 0, 0));
	animation_static_boy.AddBitmap("RES\\player\\title\\boy_static4.bmp", RGB(0, 0, 0));
	animation_static_girl.AddBitmap("RES\\player\\title\\girl_static0.bmp", RGB(0, 0, 0));
	animation_static_girl.AddBitmap("RES\\player\\title\\girl_static1.bmp", RGB(0, 0, 0));
	animation_static_girl.AddBitmap("RES\\player\\title\\girl_static2.bmp", RGB(0, 0, 0));
	animation_static_girl.AddBitmap("RES\\player\\title\\girl_static3.bmp", RGB(0, 0, 0));
	animation_static_girl.AddBitmap("RES\\player\\title\\girl_static4.bmp", RGB(0, 0, 0));
	animation_static_boy.SetDelayCount(2);
	animation_static_girl.SetDelayCount(2);
	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
}
void CGameStateInit::OnLButtonDown(UINT nFlags, CPoint point)
{
	//GotoGameState(GAME_STATE_RUN);		// 切換至 GAME_STATE_RUN
	/*
	about: x=530~750, y=830~930
	open origin game: x=350~730, y=490~580
	back to title: x=430~840, y=670~750
	*/
	int x = point.x;
	int y = point.y;
	switch (flag_background_type)
	{
	case 0:
	{
		if (x >= 530 && x <= 750 && y >= 830 && y <= 930)
		{
			//about
			flag_background_type = 1;
		}
		else
		{
			GotoGameState(GAME_STATE_RUN);		// 切換至 GAME_STATE_RUN
		}
	}
		break;
	case 1:
	{
		if (x >= 350 && x <= 730 && y >= 490 && y <= 580)
		{
			//open original game
			char*					linkChar
=
"https://html5.gamedistribution.com/a55c9cc9c21e4fc683c8c6857f3d0c75/";
			ShellExecute(NULL, NULL, linkChar, NULL, NULL, SW_SHOWNORMAL);
		}
```

```
            else if (x >= 430 && x <= 840 && y >= 670 && y <= 750)
            {
                //back to game title
                flag_background_type = 0;
            }
        }
            break;
        default:
            break;
        }
}
void CGameStateInit::OnMouseMove(UINT nFlags, CPoint point)
{
    //TRACE("x=%d,y=%d\n", point.x, point.y);
}
void CGameStateInit::OnShow()
{
    //
    // 貼上 logo
    //
    switch (flag_background_type)
    {
    case 0:
        background_title.SetTopLeft(0, 0);
        background_title.ShowBitmap();
        break;
    case 1:
        background_about.SetTopLeft(0, 0);
        background_about.ShowBitmap();
        break;
    default:
        break;
    }
    animation_static_boy.SetTopLeft(400, 740);
    animation_static_boy.OnShow();
    animation_static_boy.OnMove();
    animation_static_girl.SetTopLeft(848, 740);
    animation_static_girl.OnShow();
    animation_static_girl.OnMove();
    //
    // Demo 螢幕字型的使用，不過開發時請盡量避免直接使用字型，改用 CMovingBitmap 比較
好
    //
    //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(455, 640, "Please click mouse or press SPACE to begin.");
    //pDC->TextOut(5, 810, "Press Ctrl-F to switch in between window mode and full screen mode.");
    //if (ENABLE_GAME_PAUSE)
    //    pDC->TextOut(5, 840, "Press Ctrl-Q to pause the Game.");
    //pDC->TextOut(5, 870, "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
}
/////////////////////////////////////////////////////////////////////////////
// 這個 class 為遊戲的結束狀態(Game Over)
/////////////////////////////////////////////////////////////////////////////
CGameStateOver::CGameStateOver(CGame *g)
: CGameState(g)
{
}
void CGameStateOver::OnMove()
{
    counter--;
    if (counter < 0)
        GotoGameState(GAME_STATE_INIT);
```

```
}
void CGameStateOver::OnBeginState()
{
	counter = 30 * 5; // 5 seconds
	final_score_boy = score_boy;
	filal_score_girl = score_girl;
}
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。
	//
	//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
	////
	//// 最終進度為 100%
	////
	//ShowInitProgress(100);
}
void CGameStateOver::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));
	char str[80];                                 // Demo 數字對字串的轉換
	if (flag_win)
	{
		sprintf(str, "You Win ! (%d)", counter / 30);
	}
	else
	{
		sprintf(str, "You Lose ! (%d)", counter / 30);
	}
	pDC->TextOut(570, 420, str);
	//todo: fixbug 分数第二次显示被清零
	sprintf(str, "Boy Score: %d", final_score_boy);
	pDC->TextOut(580, 450, str);
	sprintf(str, "Girl Score: %d", filal_score_girl);
	pDC->TextOut(580, 480, str);
	pDC->SelectObject(fp);                        // 放掉 font f (千萬不要漏了放掉)
	CDDraw::ReleaseBackCDC();                     // 放掉 Back Plain 的 CDC
}
}
```

3. CGameStateRun.h

```
#pragma once
#define MAP_SIZE_WIDTH 40
#define MAP_SIZE_HEIGHT 30
#define MAP_GIRD_PIXEL 32
#include "Item.h"
#include "Wall.h"
#include "Diamond.h"
#include "Door.h"
#include "Switch.h"
#include "Platform.h"
#include "Pool.h"
#include "Switch.h"
#include "Wind.h"
#include "Box.h"
#include "Player.h"
#include "Menu.h"
#include "Tips.h"
```

```
#include <memory>
namespace game_framework {
	struct PlayerCoordinate
	{
		int x1;
		int y1;
		int x2;
		int y2;
	};
	enum AUDIO_ID {
		AUDIO_DIE,			// 0
		AUDIO_ADV,			// 1
		AUDIO_DIAMOND,				// 2
		AUDIO_GIRLJUMP,				// 2
		AUDIO_BOYJUMP,		// 2
		AUDIO_LAKE,
		AUDIO_DOOR
	};
	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
		//////////////////////////////////////////////////////////////
		void ResetMap();
		void LoadGameBitmap();
		void LoadItemBitmap();
		void InitMapLevel(int level);//初始化地图数据 0~9
		void MovePlayer(shared_ptr<Player> player);//移动玩家
		//void MovePlatform(shared_ptr<Player> platform, shared_ptr<Player> player1,
shared_ptr<Player> player2);//移动平台
		bool CanMove(shared_ptr<Player> player, int direction);//判定玩家是否能移动
		void PlayerDie(shared_ptr<Player> player);//玩家死亡
		void PlayerReachExit(shared_ptr<Player> player,bool is_arrive);//玩家到达出口
		void ItemInteract(shared_ptr<Item> item, shared_ptr<Player> player, PlayerCoordinate
coordinate, int direction);
		//void DeleteItem(shared_ptr<Item> item);//删除元素
		PlayerCoordinate GetPlayerCoordinate(shared_ptr<Player> player);//获取玩家座标
		//void ShowPauseMenu();//显示暂停菜单
		//void ShowOverMenu();//显示结束菜单
	protected:
		void OnMove();											// 移動遊戲元素
		void OnShow();											// 顯示這個狀態的遊戲畫面
	private:
		//int now_level = 0;//目前的关卡
		int	map_array[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH];//地图数据
		int show_menu;//是否显示菜单，0：不显示，1：显示过关，2：显示死亡
		//int score_boy = 0;
		//int score_girl = 0;
		shared_ptr<Player> boy;
		shared_ptr<Player> girl;
		//vector<Item*>		items;//各种物体
		vector<shared_ptr<Item>>		item_ptrs;//各种物体
		CMovingBitmap	background;//背景
		bool flag_game_loaded = false;
		bool flag_change_level = false;
		bool flag_fan_boy = false;
		bool flag_fan_girl = false;
		//-----------------------------------------------
		//shared_ptr<Menu> game_menu;
		//int flag_game_menu = 0;
```

```
        //int mouse_click = 0;
        //int mouse_x = 0;
        //int mouse_y = 0;
        //-----------------------------------------------
    };
}
```

4. CGameStateRun_Main.cpp

```
#pragma once
#define MAP_SIZE_WIDTH 40
#define MAP_SIZE_HEIGHT 30
#define MAP_GIRD_PIXEL 32
#include "Item.h"
#include "Wall.h"
#include "Diamond.h"
#include "Door.h"
#include "Switch.h"
#include "Platform.h"
#include "Pool.h"
#include "Switch.h"
#include "Wind.h"
#include "Box.h"
#include "Player.h"
#include "Menu.h"
#include "Tips.h"
#include <memory>
namespace game_framework {
    struct PlayerCoordinate
    {
        int x1;
        int y1;
        int x2;
        int y2;
    };
    enum AUDIO_ID {
        AUDIO_DIE,                  // 0
        AUDIO_ADV,                  // 1
        AUDIO_DIAMOND,                  // 2
        AUDIO_GIRLJUMP,                 // 2
        AUDIO_BOYJUMP,      // 2
        AUDIO_LAKE,
        AUDIO_DOOR
    };
    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
        ////////////////////////////////////////////////////////////////
        void ResetMap();
        void LoadGameBitmap();
        void LoadItemBitmap();
        void InitMapLevel(int level);//初始化地图数据 0~9
        void MovePlayer(shared_ptr<Player> player);//移动玩家
        //void MovePlatform(shared_ptr<Player> platform, shared_ptr<Player> player1,
shared_ptr<Player> player2);//移动平台
        bool CanMove(shared_ptr<Player> player, int direction);//判定玩家是否能移动
        void PlayerDie(shared_ptr<Player> player);//玩家死亡
        void PlayerReachExit(shared_ptr<Player> player,bool is_arrive);//玩家到达出口
        void ItemInteract(shared_ptr<Item> item, shared_ptr<Player> player, PlayerCoordinate
coordinate, int direction);
```

```
		//void DeleteItem(shared_ptr<Item> item);//删除元素
		PlayerCoordinate GetPlayerCoordinate(shared_ptr<Player> player);//获取玩家座标
		//void ShowPauseMenu();//显示暂停菜单
		//void ShowOverMenu();//显示结束菜单
	protected:
		void OnMove();							// 移動遊戲元素
		void OnShow();							// 顯示這個狀態的遊戲畫面
	private:
		//int now_level = 0;//目前的关卡
		int	map_array[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH];//地图数据
		int show_menu;//是否显示菜单，0：不显示，1：显示过关，2：显示死亡
		//int score_boy = 0;
		//int score_girl = 0;
		shared_ptr<Player> boy;
		shared_ptr<Player> girl;
		//vector<Item*>		items;//各种物体
		vector<shared_ptr<Item>>		item_ptrs;//各种物体
		CMovingBitmap	background;//背景
		bool flag_game_loaded = false;
		bool flag_change_level = false;
		bool flag_fan_boy = false;
		bool flag_fan_girl = false;
		//---------------------------------------------
		//shared_ptr<Menu> game_menu;
		//int flag_game_menu = 0;
		//int mouse_click = 0;
		//int mouse_x = 0;
		//int mouse_y = 0;
		//---------------------------------------------
	};
}
```

5. CGameStateRun_InitMap.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include <memory>
#include "CGameStateRun.h"
namespace game_framework
{
	static int global_id;//全局 id
	void CGameStateRun::InitMapLevel(int level)//初始化地图数据 0~9
	{
		CAudio::Instance()->Play(AUDIO_ADV, true);
		//停止游戏逻辑
		flag_game_loaded = false;
		//清空地图
		ResetMap();
		//设置目前关卡
		flag_now_level = level;
		//设置新的 id;
		global_id = 0;
		//定义地图数据
		int map_level_0[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
			{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},
			{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,0,0,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,100,101,0,400,0,0,401,0,0,1,0},
			{1,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,0,201,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
			{1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,1,1,1,1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,1,0},
			{1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0},
```

```
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0},
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,6,302,302,302,7,1,1,1,0,0,0,0,1,0},
                {1,0,101,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0},

        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,/*601*/0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,201,0,0,0,0,0,1,1,1,1,0},
                {1,0,100,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0},

        {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,300,300,300,3,1,1,1,4,301,301,301,5,1,1,1,1,1,1,1,0}
            };
            int map_level_1[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {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},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,400,0,0,401,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,201,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,200,0,0,0,201,0,0,0,0,1,1,0,0,0,0,0,1,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},

        {1,0,0,0,0,0,0,1,6,302,302,302,302,302,302,7,1,1,1,1,1,1,1,6,302,302,302,302,302,302,7,1,1,1,1,1
,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,0,0,0,0,0,0,0,201,0,0,0,0,200,0,0,0,0,0,0,0,201,0,0,0,200,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,201,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,200,0,0,0,200,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},

        {1,0,0,0,0,0,4,301,301,301,301,301,301,301,5,0,0,0,0,0,0,0,2,300,300,300,300,300,300,300,3,0,0,
0,0,0,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,200,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,201,0,0,0,0,0,0,1,1,1,1,0},
                {1,0,100,0,101,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},

        {1,1,1,1,1,1,2,300,300,300,300,300,300,300,3,1,1,1,1,1,1,1,4,301,301,301,301,301,301,301,5,1,1,
1,1,1,1,1,1,0}
            };
            int map_level_2[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {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},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,1,0},
                {1,0,201,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,200,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,201,0,0,0,0,0,0,1,0,0,0,0,0,0,201,0,0,1,0,0,200,0,0,0,0,0,0,1,0,0,0,0,0,0,200,0,1,0},
```

```
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,1,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,201,0,0,0,0,0,0,1,0,200,0,0,0,0,0,0,1,1,0,1,1,0,0,0,0,0,0,201,0,1,0,0,0,0,0,0,200,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,2,300,300,3,0,0,0,0,0,1,0,0,0,0,0,4,301,301,5,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,600,0,0,1,1,0,0,0,1,0,400,0,0,0,0,0,0,0,1,0,0,0,0,0,401,0,0,0,1,0,0,0,1,1,600,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,0,0,0,0,0,201,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,0,200,0,0,0,0,0,1,1,1,0},
                {1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0},

        {1,1,1,1,1,1,4,301,301,301,301,301,301,5,1,1,0,0,0,0,0,0,1,1,2,300,300,300,300,300,300,3,1,1,1
,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,1,1,1,1,1,1,1,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,200,0,0,200,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,201,0,0,201,0,0,0,0,0,0,0,1,0},
                {1,0,100,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,101,0,1,0},

        {1,1,1,1,1,1,2,300,300,300,300,300,300,3,1,1,1,1,1,1,1,1,1,1,1,4,301,301,301,301,301,301,5,1,1,1
,1,1,1,0}
        };
        int map_level_3[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,101,0,0,1,0},
                {1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,1,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,1,0,201,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,201,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,200,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,0,0,0,1,0,0,0,1,1,1,1,1,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,200,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,201,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,200,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,400,0,0,401,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,1,600,0,0,1,1,1,1,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,201,0,0,0,0,0,0,0,0,0,1,1,0},
                {1,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0},
                {1,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,201,0,0,0,1,1,1,0},
                {1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0},

        {1,0,0,0,0,100,0,1,1,1,1,6,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,7,
1,1,1,1,1,1,1,1,1,1,0},
                {1,600,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0}
        };
        int map_level_4[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,0,0,1,0},

        {1,0,0,0,0,0,1,4,301,301,301,301,301,5,1,0,0,0,0,0,0,0,0,1,2,300,300,300,300,300,3,1,0,0,0,0,0,
1,0},
                {1,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,200,0,0,0,201,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0},
```

```
                {1,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,1,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,201,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,200,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,401,0,0,0,1,0,200,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,400,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,201,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,200,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,1,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,1,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,201,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,200,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,600,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,600,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,100,0,0,0,0,0,1,0,0,0,0,0,101,0,1,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,0},

        {1,2,300,300,300,300,300,300,300,3,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,4,301,301,301,301,301,
301,301,301,5,1,0},
        };
        int map_level_5[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {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},
                {1,1,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,1,1,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,100,0,1,1,0},
                {1,1,0,0,0,0,0,0,0,1,1,1,1,0,200,0,0,0,0,0,200,0,0,0,0,0,200,0,0,0,0,0,0,1,1,1,1,1,1,0},
                {1,0,0,0,0,200,0,0,0,1,1,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,0,0,0,0,1,0},

        {1,0,0,0,0,0,0,0,1,1,0,0,0,0,1,1,4,301,301,5,1,1,4,301,301,5,1,1,4,301,301,5,1,1,0,0,0,0,0,1,0},
                {1,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0},
                {1,0,0,0,200,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,200,0,0,1,1,0,0,0,1,0,1,0,0,1,0},
                {1,1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,1,1,1,0,0,1,0,0,1,0},
                {1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,4,301,5,1,1,1,1,1,1,1,1,1,1,1,1,400,0,0,0,1,0,0,1,0},
                {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,1,1,0},
                {1,1,0,101,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0},
                {1,1,1,1,1,1,1,0,0,0,0,0,0,0,201,0,0,0,0,0,201,0,0,0,0,0,201,0,1,1,1,1,0,0,0,0,0,0,0,1,1,0},
                {1,1,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,1,0,0,1,1,0,0,201,0,0,0,0,1,0},

        {1,0,0,0,0,0,1,1,2,300,300,3,1,1,2,300,300,3,1,1,2,300,300,3,1,1,0,0,0,0,1,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,1,0},
                {1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,0,1,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,1,0,1,0,0,0,1,1,0,0,201,0,0,1,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,201,0,0,0,1,0},
                {1,0,0,1,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,1,0},
                {1,0,0,1,0,0,0,0,1,0,0,0,1,1,1,1,1,1,0,0,1,1,2,300,3,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,0},
                {1,0,0,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,401,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0},
                {1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0},
                {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}
        };
        int map_level_6[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
                {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},
                {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0},
                {1,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0},
                {1,0,100,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,1,0,101,0,0,0,0,0,1,0},
                {1,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,1,0},
                {1,1,0,0,0,0,0,1,1,1,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,1,0},
                {1,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
```

```
        {1,0,0,200,0,0,0,0,1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,1,1,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1,0},
        {1,0,0,1,1,2,300,300,300,3,1,1,1,1,1,0,0,0,0,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,200,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,200,0,1,0,0,0,0,0,0,1,4,301,301,5,1,1,1,0},
        {1,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,201,0,0,1,1,1,1,1,1,1,1,1,1,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,201,0,1,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,1,0},
        {1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,1,0},
        {1,1,1,1,1,1,1,1,4,301,5,1,1,4,5,1,1,1,0,0,0,1,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,1,1,1,1,1,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,0,0,1,2,3,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,201,1,1,1,2,3,1,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,200,0,0,0,0,201,0,0,0,0,0,1,0,0,0,1,1,1,1,1,0,201,0,1,1,1,1,0,0,1,0},
        {1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,1,1,1,2,300,300,3,1,1,4,301,301,5,1,0,0,0,0,0,0,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
        {1,1,0,0,0,0,0,0,0,0,200,0,201,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,400,0,0,401,0,0,0,0,0,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {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}
    };
    int map_level_7[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
        {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},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,1,0,0,1,0},
        {1,1,100,0,0,0,0,0,200,0,0,0,0,0,201,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0},
        {1,1,1,1,101,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0,1,1,1,0},
        {1,0,0,1,1,1,1,2,300,3,1,1,4,301,5,1,1,2,300,3,1,1,0,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,1,1,0},
        {1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,1,0},
        {1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,1,0,0,1,0},
        {1,1,1,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,200,0,1,4,5,1,0,0,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,1,0,0,1,1,0,0,1,0},

{1,0,0,1,4,5,1,1,4,301,5,1,1,4,301,301,301,301,301,5,1,1,1,1,1,1,0,0,0,0,1,1,0,0,0,1,0,0,1,0},
        {1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,201,0,0,0,0,0,1,0},
        {1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,1,0},
        {1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,1,0,0,1,0,1,1,4,301,301,301,5,1,0},
        {1,0,0,0,1,1,1,1,1,1,0,0,0,0,200,0,1,1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,1,0,1,1,1,1,1,1,1,0},
        {1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,1,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,200,0,0,0,0,0,201,0,0,1,0},
        {1,0,0,0,0,0,0,0,1,4,301,5,1,1,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,1,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,1,1,1,0},
        {1,400,0,0,401,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,0,0,1,0},

{1,1,1,1,1,1,1,1,1,0,0,0,1,6,302,302,302,302,302,302,302,302,7,1,0,1,1,1,1,1,1,0,1,1,0,0,0,0,1,0},
        {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}
    };
    int map_level_8[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
        {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},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,1,1,1,0,0,0,0,0,0,1,0},
        {1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,200,0,1,0},

{1,0,0,0,1,1,2,300,300,300,3,1,1,0,0,0,0,100,101,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
        {1,0,0,1,1,1,0,0,0,0,0,0,0,0,201,0,1,1,1,1,0,0,0,0,200,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0},
        {1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},

{1,0,0,1,0,0,0,1,4,301,301,301,301,301,5,1,0,0,0,1,1,2,300,3,1,2,3,1,0,0,201,0,0,0,0,1,0,0,1,0},


```

```
{1,0,0,1,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,1,0,0,1,0},
{1,0,0,1,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,1,0},
{1,0,0,1,1,0,0,201,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,1,1,1,1,1,0},
{1,0,0,1,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,200,0,0,1,1,0,0,0,1,1,0,0,0,1,1,1,1,0},
{1,0,0,0,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,201,0,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,4,5,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0},
{1,1,0,0,0,0,0,0,0,0,0,0,200,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,1,4,301,301,5,1,1,1,0},
{1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,0},
{1,1,0,1,1,1,4,5,1,4,301,5,1,1,4,301,5,1,0,0,0,1,1,0,0,0,0,0,1,1,1,0,0,0,0,400,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,201,0,1,1,0,0,0,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
{1,401,0,0,0,0,0,0,1,1,0,0,0,200,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,201,0,0,0,0,0,0,0,0,1,0},
{1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},
{1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,1,0},
{1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0},

{1,0,0,1,6,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,302,
302,302,302,302,302,302,302,302,302,302,302,302,302,7,1,0},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
};
int map_level_9[MAP_SIZE_HEIGHT][MAP_SIZE_WIDTH] = {
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0},
{1,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0},
{1,0,0,0,1,1,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0},
{1,1,1,1,1,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0},
{1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,0,0,0,0,1,1,1,0,0,0,0,1,0},
{1,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,0},
{1,1,0,0,0,0,0,0,1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0},
{1,0,0,0,0,0,1,1,0,0,0,1,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,1,1,0},
{1,0,0,0,0,1,1,1,0,0,0,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,1,0},
{1,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,0,0,0,1,0,0,1,1,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,0,1,1,0,0,0,1,1,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,1,1,0,0,0,0,0,0,1,1,0,0,1,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,1,0},
{1,1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,1,1,1,0,0,0,0,1,1,1,1,1,1,0,0,0,1,1,1,1,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,1,1,0,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,4,301,301,301,5,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,1,1,0,1,0},
{1,0,0,0,0,0,0,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,1,1,1,1,0,0,0,1,1,1,0},
{1,1,0,200,0,201,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1,1,0},
{1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,1,1,1,0},
{1,1,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,2,300,3,1,0,0,0,0,0,1,1,1,1,0},
{1,1,1,1,1,1,1,1,1,1,4,301,301,301,5,1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,0},
{1,1,0,0,0,0,0,0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},
{1,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,1,1,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,400,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,1,1,401,0,0,0,1,0},
{1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,1,0},

{1,0,0,0,1,1,2,300,300,300,3,1,0,0,0,0,0,0,101,1,1,1,1,1,1,1,1,1,1,1,100,0,0,0,0,0,0,0,1,1,0},
{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0}
};
//地图赋值
switch (level)
{
case 0:
{
for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
{
for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
{
map_array[i][j] = map_level_0[i][j];
}
```

```
        }
        //添加 tips
        //move-water
        shared_ptr<Tips> tips_move_water = make_shared<Tips>("move-water", 3, 22);
        item_ptrs.push_back(tips_move_water);
        //move-fire
        shared_ptr<Tips> tips_move_fire = make_shared<Tips>("move-fire", 2, 26);
        item_ptrs.push_back(tips_move_fire);
        //fire-and-water
        shared_ptr<Tips> tips_fire_and_water = make_shared<Tips>("fire-and-water", 19, 24);
        item_ptrs.push_back(tips_fire_and_water);
        //green-goo
        shared_ptr<Tips> tips_green_goo = make_shared<Tips>("green-goo", 19, 20);
        item_ptrs.push_back(tips_green_goo);
        //lever
        shared_ptr<Tips> tips_lever = make_shared<Tips>("lever", 2, 17);
        item_ptrs.push_back(tips_lever);
        //button
        shared_ptr<Tips> tips_button = make_shared<Tips>("button", 10, 13);
        item_ptrs.push_back(tips_button);
        //box
        //shared_ptr<Tips> tips_box = make_shared<Tips>("box", 23, 7);
        //item_ptrs.push_back(tips_box);
        //cheat
        shared_ptr<Tips> tips_cheat = make_shared<Tips>("cheat", 20, 7);
        item_ptrs.push_back(tips_cheat);
        //diamond
        shared_ptr<Tips> tips_diamond = make_shared<Tips>("diamond", 1, 1);
        item_ptrs.push_back(tips_diamond);
        //door
        shared_ptr<Tips> tips_door = make_shared<Tips>("door", 15, 1);
        item_ptrs.push_back(tips_door);
        //添加机关
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(502, 1, 16, 1, 19);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick = make_shared<Switch>(501, 9, 20);
        platform1_stick->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(502, 34, 13, 34, 16);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(500, 10, 15);
        shared_ptr<Switch> platform2_stick2 = make_shared<Switch>(500, 30, 11);
        platform2_stick1->Bind(platform2);//绑定平台
        platform2_stick2->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        item_ptrs.push_back(platform2_stick2);
        break;
    }
    case 1:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_1[i][j];
            }
        }
        //添加机关
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(503, 19, 17, 19, 11);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(500, 5, 20);
        shared_ptr<Switch> platform1_stick2 = make_shared<Switch>(500, 33, 20);
        platform1_stick1->Bind(platform1);//绑定平台
        platform1_stick2->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        item_ptrs.push_back(platform1_stick2);
```

```
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(502, 17, 1, 17, 5);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(500, 12, 4);
        shared_ptr<Switch> platform2_stick2 = make_shared<Switch>(500, 26, 4);
        platform2_stick1->Bind(platform2);//绑定平台
        platform2_stick2->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        item_ptrs.push_back(platform2_stick2);
        break;
    }
    case 2:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_2[i][j];
            }
        }
        break;
    }
    case 3:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_3[i][j];
            }
        }
        //添加机关
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(503, 7, 1, 7, 4);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(500, 32, 13);
        platform1_stick1->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(503, 11, 21, 11, 24);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(501, 4, 3);
        platform2_stick1->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        break;
    }
    case 4:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_4[i][j];
            }
        }
        //添加机关
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(502, 16, 25, 19, 25);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(501, 13, 21);
        platform1_stick1->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(502, 19, 19, 16, 19);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(501, 25, 15);
        platform2_stick1->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        //platform3
```

```
        shared_ptr<Platform> platform3 = make_shared<Platform>(502, 16, 13, 19, 13);
        item_ptrs.push_back(platform3);
        shared_ptr<Switch> platform3_stick1 = make_shared<Switch>(501, 13, 9);
        platform3_stick1->Bind(platform3);//绑定平台
        item_ptrs.push_back(platform3_stick1);
        break;
    }
    case 5:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_5[i][j];
            }
        }
        break;
    }
    case 6:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_6[i][j];
            }
        }
        //添加机关
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(502, 17, 17, 13, 17);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(501, 3, 15);
        platform1_stick1->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(502, 20, 22, 24, 22);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(501, 31, 23);
        platform2_stick1->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        //platform3
        shared_ptr<Platform> platform3 = make_shared<Platform>(503, 26, 25, 26, 21);
        item_ptrs.push_back(platform3);
        shared_ptr<Switch> platform3_button1 = make_shared<Switch>(500, 19, 28);
        shared_ptr<Switch> platform3_button2 = make_shared<Switch>(500, 36, 28);
        platform3_button1->Bind(platform3);//绑定平台
        platform3_button2->Bind(platform3);//绑定平台
        item_ptrs.push_back(platform3_button1);
        item_ptrs.push_back(platform3_button2);
        break;
    }
    case 7:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_7[i][j];
            }
        }
        break;
    }
    case 8:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_8[i][j];
```

```
            }
        }
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(503, 12, 0, 12, 4);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(501, 35, 5);
        platform1_stick1->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(502, 32, 9, 35, 9);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(501, 10, 11);
        platform2_stick1->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        break;
    }
    case 9:
    {
        for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
        {
            for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
            {
                map_array[i][j] = map_level_9[i][j];
            }
        }
        //platform1
        shared_ptr<Platform> platform1 = make_shared<Platform>(503, 27, 23, 27, 27);
        item_ptrs.push_back(platform1);
        shared_ptr<Switch> platform1_stick1 = make_shared<Switch>(501, 25, 21);
        platform1_stick1->Bind(platform1);//绑定平台
        item_ptrs.push_back(platform1_stick1);
        //platform2
        shared_ptr<Platform> platform2 = make_shared<Platform>(503, 4, 15, 4, 12);
        item_ptrs.push_back(platform2);
        shared_ptr<Switch> platform2_stick1 = make_shared<Switch>(501, 2, 28);
        platform2_stick1->Bind(platform2);//绑定平台
        item_ptrs.push_back(platform2_stick1);
        //platform3
        shared_ptr<Platform> platform3 = make_shared<Platform>(502, 28, 16, 25, 16);
        item_ptrs.push_back(platform3);
        shared_ptr<Switch> platform3_button1 = make_shared<Switch>(500, 17, 17);
        platform3_button1->Bind(platform3);//绑定平台
        item_ptrs.push_back(platform3_button1);
        break;
    }
    default:
        break;
    }
    //加载物件
    for (size_t i = 0; i < MAP_SIZE_HEIGHT; i++)
    {
        for (size_t j = 0; j < MAP_SIZE_WIDTH; j++)
        {
            switch (map_array[i][j])
            {
            case 1:
            {
                shared_ptr<Item> wall = make_shared<Wall>();
                wall->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                item_ptrs.push_back(wall);
                break;
            }
            case 2:
            {
                shared_ptr<Item> fire_pool = make_shared<Pool>(2);
                fire_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                item_ptrs.push_back(fire_pool);
                break;
            }
```

```
                case 3:
                {
                    shared_ptr<Item> fire_pool = make_shared<Pool>(3);
                    fire_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(fire_pool);
                    break;
                }
                case 4:
                {
                    shared_ptr<Item>    water_pool = make_shared<Pool>(4);
                    water_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(water_pool);
                    break;
                }
                case 5:
                {
                    shared_ptr<Item> water_pool = make_shared<Pool>(5);
                    water_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(water_pool);
                    break;
                }
                case 6:
                {
                    shared_ptr<Item> toxic_pool = make_shared<Pool>(6);
                    toxic_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(toxic_pool);
                    break;
                }
                case 7:
                {
                    shared_ptr<Item> toxic_pool = make_shared<Pool>(7);
                    toxic_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(toxic_pool);
                    break;
                }
                case 100:
                {
                    boy = make_unique<Player>(true);
                    boy->SetTopLeft(i * MAP_GIRD_PIXEL, j * MAP_GIRD_PIXEL);
                    break;
                }
                case 101:
                {
                    girl = make_unique<Player>(false);
                    girl->SetTopLeft(i * MAP_GIRD_PIXEL, j * MAP_GIRD_PIXEL);
                    break;
                }
                case 200:
                {
                    shared_ptr<Item> fire_diamond = make_shared<Diamond>(200);
                   fire_diamond->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(fire_diamond);
                    break;
                }
                case 201:
                {
                    shared_ptr<Item>    water_diamond = make_shared<Diamond>(201);
                    water_diamond->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL *
j);
                    item_ptrs.push_back(water_diamond);
                    break;
                }
                case 300:
                {
                    shared_ptr<Item> fire_pool = make_shared<Pool>(300);
                    fire_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                    item_ptrs.push_back(fire_pool);
                    break;
                }
```

```
                    case 301:
                    {
                        shared_ptr<Item> water_pool = make_shared<Pool>(301);
                        water_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                        item_ptrs.push_back(water_pool);
                        break;
                    }
                    case 302:
                    {
                        shared_ptr<Item> toxic_pool = make_shared<Pool>(302);
                        toxic_pool->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                        item_ptrs.push_back(toxic_pool);
                        break;
                    }
                    case 400:
                    {
                        shared_ptr<Item> fire_door = make_shared<Door>(400);
                        fire_door->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                        item_ptrs.push_back(fire_door);
                        break;
                    }
                    case 401:
                    {
                        shared_ptr<Item> water_door = make_shared<Door>(401);
                        water_door->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                        item_ptrs.push_back(water_door);
                        break;
                    }
                    case 600:
                    {
                        shared_ptr<Item> wind = make_shared<Wind>();
                        wind->SetTopLeft(MAP_GIRD_PIXEL* i, MAP_GIRD_PIXEL* j);
                        item_ptrs.push_back(wind);
                        break;
                    }
                    case 601:
                    {
                        shared_ptr<Item> box = make_shared<Box>();
                        box->SetTopLeft(MAP_GIRD_PIXEL * i, MAP_GIRD_PIXEL * j);
                        item_ptrs.push_back(box);
                        break;
                    }
                    default:
                        break;
                    }
                }
            }
            LoadItemBitmap();
            //继续游戏逻辑
            flag_game_loaded = true;
        }
}
```

6. CGameStateRun_Interact.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include <memory>
#include "CGameStateRun.h"
#include <algorithm>
namespace game_framework
{
    void CGameStateRun::ItemInteract(shared_ptr<Item> item, shared_ptr<Player>
player,PlayerCoordinate coordinate, int direction)
    {
        //获取玩家坐标
```

```
const int x1 = coordinate.x1;
const int y1 = coordinate.y1;
const int x2 = coordinate.x2;
const int y2 = coordinate.y2;
//获取物品类型
int type = item->GetType();
if (type == 200 && player->is_boy)//fire diamond
{
	CAudio::Instance()->Play(AUDIO_DIAMOND, true);
	player->score++;
	item->is_visibale = false;
	//DeleteItem(item);
	CAudio::Instance()->Play(AUDIO_DIAMOND, false);
	//TRACE("boy score=%d, total=%d\n", player->score, score_boy);
}
else if (type == 201 && !player->is_boy)//water diamond
{
	CAudio::Instance()->Play(AUDIO_DIAMOND, true);
	player->score++;
	item->is_visibale = false;
	//DeleteItem(item);
	CAudio::Instance()->Play(AUDIO_DIAMOND, false);
	//TRACE("girl score=%d, total=%d\n", player->score, score_boy);
}
else if (type == 300 || type == 301 || type == 302)//只有水池中部有效
{
	if (direction == DIRECTION_DOWN)//如果玩家下方接触水池上方
	{
		CAudio::Instance()->Play(AUDIO_LAKE, true);
		CAudio::Instance()->Play(AUDIO_LAKE, false);
		if (type == 300 && !player->is_boy)//fire & girl
		{
			//CAudio::Instance()->Play(AUDIO_DIE, true);
			PlayerDie(player);
			//CAudio::Instance()->Stop(AUDIO_DIE);
		}
		else if (type == 301 && player->is_boy)//water & boy
		{
			//CAudio::Instance()->Play(AUDIO_DIE, true);
			PlayerDie(player);
			//CAudio::Instance()->Stop(AUDIO_DIE);
		}
		else if (type == 302)//toxic && both
		{
			//CAudio::Instance()->Play(AUDIO_DIE, true);
			PlayerDie(player);
			//CAudio::Instance()->Stop(AUDIO_DIE);
		}
	}
}
else if (type == 500)//按钮
{
	//指针转到 switch 类
	shared_ptr<Switch> switch_ptr = std::static_pointer_cast<Switch>(item);
	if ((x1 > switch_ptr->GetX1() - 16) && (x2 < switch_ptr->GetX2() + 16))//扩大判定范
围？
	{
		if (switch_ptr->status == SWITCH_RELEASE)//按钮只能被按下一次
		{
			switch_ptr->is_on = true;
			switch_ptr->status = SWITCH_PRESS;
		}
	}
	else
	{
		if (switch_ptr->status == SWITCH_PRESS)//玩家离开后按钮被松开
		{
			switch_ptr->is_on = false;
			switch_ptr->status = SWITCH_RELEASE;
```

```
                }
            }
            //触发开关
            switch_ptr->Trigger();
        }
        else if (type == 501)//拉杆
        {
            //指针转到 switch 类
            shared_ptr<Switch> switch_ptr = std::static_pointer_cast<Switch>(item);
            if ((x1 > switch_ptr->GetX1() - 16) && (x2 < switch_ptr->GetX2() + 16))//扩大判定范
围？
            {
                if (switch_ptr->status == SWITCH_RELEASE)//拉杆被触发，转换状态
                {
                    switch_ptr->is_on = !switch_ptr->is_on;
                    switch_ptr->status = SWITCH_PRESS;
                }
            }
            else
            {
                if (switch_ptr->status == SWITCH_PRESS)//玩家离开后，不松开
                {
                    switch_ptr->status = SWITCH_RELEASE;
                }
            }
            //触发开关
            switch_ptr->Trigger();
        }
        else if (type == 400 && player->is_boy)//门
        {
            //指针转到 door 类
            shared_ptr<Door> door_ptr = std::static_pointer_cast<Door>(item);
            if ((x1 > door_ptr->GetX1()) && (x2 < door_ptr->GetX2()))//玩家完全进入门内
            {
                if (door_ptr->status == DOOR_LEAVE)
                {
                    door_ptr->status = DOOR_ENTER;
                    PlayerReachExit(player, true);
                    door_ptr->AnimationOpenDoor();//触发开门动画
                    CAudio::Instance()->Play(AUDIO_DOOR, true);
                    CAudio::Instance()->Play(AUDIO_DOOR, false);
                }
            }
            else//玩家离开门
            {
                if (door_ptr->status == DOOR_ENTER)
                {
                    door_ptr->status = DOOR_LEAVE;
                    PlayerReachExit(player, false);
                    door_ptr->AnimationCloseDoor();//触发关门动画
                    CAudio::Instance()->Play(AUDIO_DOOR, true);
                    CAudio::Instance()->Play(AUDIO_DOOR, false);
                }
            }
        }
        else if (type == 401 && !player->is_boy)//门
        {
            //指针转到 door 类
            shared_ptr<Door> door_ptr = std::static_pointer_cast<Door>(item);
            if ((x1 > door_ptr->GetX1()) && (x2 < door_ptr->GetX2()))//玩家完全进入门内
            {
                if (door_ptr->status == DOOR_LEAVE)
                {
                    door_ptr->status = DOOR_ENTER;
                    PlayerReachExit(player, true);
                    door_ptr->AnimationOpenDoor();//触发开门动画
                    CAudio::Instance()->Play(AUDIO_DOOR, true);
                    CAudio::Instance()->Play(AUDIO_DOOR, false);
                }
```

```
                }
                else//玩家离开门
                {
                    if (door_ptr->status == DOOR_ENTER)
                    {
                        door_ptr->status = DOOR_LEAVE;
                        PlayerReachExit(player, false);
                        door_ptr->AnimationCloseDoor();//触发关门动画
                        CAudio::Instance()->Play(AUDIO_DOOR, true);
                        CAudio::Instance()->Play(AUDIO_DOOR, false);
                    }
                }
            }
            else if (type == 600)//风扇
            {
                shared_ptr<Wind> wind_ptr = std::static_pointer_cast<Wind>(item);
                if ((x1 > wind_ptr->GetX1() - 16) && (x2 < wind_ptr->GetX2() + 16))//玩家进入
                {
                    if (direction == DIRECTION_DOWN)//如果玩家下方接触风扇上方
                    {
                        if (player->GetVerticalState() == DIRECTION_NONE)
                        {
                            //TRACE("in\n");
                            if (player->is_boy)
                            {
                                flag_fan_boy = true;
                            }
                            else
                            {
                                flag_fan_girl = true;
                            }
                        }
                    }
                }
                else//玩家离开
                {
                    //TRACE("out\n");
                    if (player->is_boy)
                    {
                        flag_fan_boy = false;
                    }
                    else
                    {
                        flag_fan_girl = false;
                    }
                }
            }
            else if (type == 601)//箱子
            {
            }
        }
}
```

7. CGameStateRun_KeyInput.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include <memory>
#include "CGameStateRun.h"
namespace game_framework
{
    const char KEY_LEFT = 37;
    const char KEY_UP = 38;
    const char KEY_RIGHT = 39;
    const char KEY_DOWN = 40;
    const char KEY_W = 87;
```

```
const char KEY_A = 65;
const char KEY_S = 83;
const char KEY_D = 68;
const char KEY_PLUS = 107;
bool flag_pause;
bool flag_gameover;
void CGameStateRun::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)
{
	if (flag_game_loaded)
	{
		//TRACE("KEY VALUE=%d\n", nChar);
		switch (nChar)
		{
		case KEY_UP:
			if (boy->GetVerticalState() == DIRECTION_NONE)
			{
				boy->SetVerticalState(DIRECTION_UP);
			}
			break;
			//case KEY_DOWN:
			//	break;
		case KEY_LEFT:
			if (boy->GetHorizontalState() == DIRECTION_NONE)
			{
				boy->SetHorizontalState(DIRECTION_LEFT);
			}
			break;
		case KEY_RIGHT:
			if (boy->GetHorizontalState() == DIRECTION_NONE)
			{
				boy->SetHorizontalState(DIRECTION_RIGHT);
			}
			break;
			////////////////////////////////////////////////
		case KEY_W:
			if (girl->GetVerticalState() == DIRECTION_NONE)
			{
				girl->SetVerticalState(DIRECTION_UP);
			}
			break;
			//case KEY_S:
			//	break;
		case KEY_A:
			if (girl->GetHorizontalState() == DIRECTION_NONE)
			{
				girl->SetHorizontalState(DIRECTION_LEFT);
			}
			break;
		case KEY_D:
			if (girl->GetHorizontalState() == DIRECTION_NONE)
			{
				girl->SetHorizontalState(DIRECTION_RIGHT);
			}
			break;
		case KEY_PLUS:
			//cheat
			//reset flag
			boy->reach_exit = false;
			girl->reach_exit = false;
			//暂停游戏逻辑
			flag_game_loaded = false;
			//准备切换关卡
			flag_change_level = true;
			//玩家分数累加到总分
			score_boy += boy->score;
			score_girl += girl->score;
			//显示暂停菜单
			//切换关卡
			flag_now_level++;
```

```
                InitMapLevel(flag_now_level);
                //显示暂停菜单
                if (score_boy == 0 && score_girl == 0)
                {
                    flag_game_menu_type = 0;//pause menu, score=0
                }
                else
                {
                    flag_game_menu_type = 1;//pause menu, score!=0
                }
                GotoGameState(GAME_STATE_PAUSE);
            default:
                break;
            }
        }
}
void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)
{
        if (flag_game_loaded)
        {
            switch (nChar)
            {
                //case KEY_UP:
                //    break;
                //case KEY_DOWN:
                //    break;
            case KEY_LEFT:
                if (boy->GetHorizontalState() == DIRECTION_LEFT)
                {
                    boy->SetHorizontalState(DIRECTION_NONE);
                }
                break;
            case KEY_RIGHT:
                if (boy->GetHorizontalState() == DIRECTION_RIGHT)
                {
                    boy->SetHorizontalState(DIRECTION_NONE);
                }
                break;
                //case KEY_W:
                //    break;
                //case KEY_S:
                //    break;
            case   KEY_A:
                if (girl->GetHorizontalState() == DIRECTION_LEFT)
                {
                    girl->SetHorizontalState(DIRECTION_NONE);
                }
                break;
            case KEY_D:
                if (girl->GetHorizontalState() == DIRECTION_RIGHT)
                {
                    girl->SetHorizontalState(DIRECTION_NONE);
                }
                break;
            default:
                break;
            }
        }
}
void CGameStateRun::OnLButtonDown(UINT nFlags, CPoint point)   // 處理滑鼠的動作
{
        //eraser.SetMovingLeft(true);
        //mouse_click = 1;
        //TRACE("mouse click\n");
}
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
		//mouse_x = point.x;
		//mouse_y = point.y;
		//TRACE("x=%d,y=%d\n", mouse_x, mouse_y);
	}
	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);
	}
}
```

8. CGameStatePause.h

```
#pragma once
#include "Menu.h"
#include<memory>
namespace game_framework {
	class CGameStatePause :public CGameState
	{
	public:
		CGameStatePause(CGame* g);
		~CGameStatePause();
		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
	protected:
		void OnMove();                                          // 移動遊戲元素
		void OnShow();
	private:
		shared_ptr<Menu> game_menu;
		int mouse_x;
		int mouse_y;
	};
}
```

9. CGameStatePause.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include<memory>
#include "CGameStatePause.h"
#include "Menu.h"
#define MAXLEVEL 10
namespace game_framework
{
	CGameStatePause::CGameStatePause(CGame* g)
		: CGameState(g)
	{
	}
	CGameStatePause::~CGameStatePause()
	{
	}
	void CGameStatePause::OnBeginState()
	{
```

```
	//每次载入都会加载
}
void CGameStatePause::OnInit()
{
	//只加载一次
	game_menu = make_shared<Menu>();
	game_menu->LoadItemBitmap();
}
void CGameStatePause::OnLButtonDown(UINT nFlags, CPoint point)  // 處理滑鼠的動作
{
	//eraser.SetMovingLeft(true);
	//TRACE("mouse click\n");
	switch (flag_game_menu_type)
	{
	case 0:
		//pause
		//score=0
		if (flag_now_level == MAXLEVEL)
		{
			//完成最后一关
			//you win
			flag_win = true;
			//展示分数
			GotoGameState(GAME_STATE_OVER);
			//关卡清零
			flag_now_level = 0;
			//分数清零
			score_boy = 0;
			score_girl = 0;
		}
		else
		{
			GotoGameState(GAME_STATE_RUN);
		}
		break;
	case 1:
		//pause
		//score!=0
		if (flag_now_level== MAXLEVEL)
		{
			//完成最后一关
			//you win
			flag_win = true;
			//展示分数
			GotoGameState(GAME_STATE_OVER);
			//关卡清零
			flag_now_level = 0;
			//分数清零
			score_boy = 0;
			score_girl = 0;
		}
		else
		{
			GotoGameState(GAME_STATE_RUN);
		}
		break;
	case 2:
		//over
		if (mouse_x<640)
		{
			//you lose
			flag_win = false;
			//展示分数
			GotoGameState(GAME_STATE_OVER);
			//关卡清零
			flag_now_level = 0;
			//分数清零
			score_boy = 0;
			score_girl = 0;
```

```
                }
                else
                {
                        //retry
                        GotoGameState(GAME_STATE_RUN);
                }
                break;
        default:
                break;
        }
}
void CGameStatePause::OnLButtonUp(UINT nFlags, CPoint point)// 處理滑鼠的動作
{
        //eraser.SetMovingLeft(false);
}
void CGameStatePause::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
        mouse_x = point.x;
        mouse_y = point.y;
        //TRACE("x=%d,y=%d\n", mouse_x, mouse_y);
}
void CGameStatePause::OnRButtonDown(UINT nFlags, CPoint point)   // 處理滑鼠的動作
{
        //eraser.SetMovingRight(true);
}
void CGameStatePause::OnRButtonUp(UINT nFlags, CPoint point)       // 處理滑鼠的動作
{
        //eraser.SetMovingRight(false);
}
void CGameStatePause::OnKeyDown(UINT, UINT, UINT)
{
}
void CGameStatePause::OnKeyUp(UINT, UINT, UINT)
{
}
void CGameStatePause::OnMove()
{
}
void CGameStatePause::OnShow()
{
        int type = flag_game_menu_type;
        game_menu->OnShow(type);
}
}
```

10. Player.h

```
#pragma once
#define PLAYER_GIRD_PIXEL 32
#define PLAYER_STEP_PIXEL 4
#define INITIAL_VELOCITY 16
namespace game_framework {
        class Map;
        class Player
        {
        public:
                Player(bool boy);
                ~Player();
                void LoadBitmapPlayer();//載入图形
                void OnMove();
                //void OnMove(CGameStateRun* game);//移动
                void OnShow();//显示
                int    GetX1();
                int    GetY1();
                int    GetX2();
                int    GetY2();
                int GetVelocity();
                int GetStep();
                void SetTopLeft(int top, int left);// 设定左上角坐标
```

```
        void SetVerticalState(int state);//设定垂直状态
        void SetHorizontalState(int state);//设定水平状态
        int GetVerticalState();//获取垂直状态
        int GetHorizontalState();//获取水平状态
        //bool HitRectangle(int tx1, int ty1, int tx2, int ty2);//碰撞检测
        bool is_boy;// 是否是男孩（火）
        bool is_visible;// 是否可见
        bool is_die;//是否死亡
        bool reach_exit;//到达出口
        int x, y;//坐标
        int moving_vertical, moving_horizontal;//移动的状态
        int   velocity;// 目前的速度(pixel/frame)
        int score;//分数
        CAnimation & GetStatic();
        CAnimation & GetLeft();
        CAnimation & GetRight();
        CAnimation & GetUp();
        CAnimation & GetDown();
        void SetAni(CAnimation& address);
    private:
        CAnimation animation_static;
        CAnimation animation_left;
        CAnimation animation_right;
        CAnimation animation_up;
        CAnimation animation_down;
        CAnimation* charactor_ani;
    };
}
```

11. Player.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "CGameStateRun.h"
#include "Player.h"
namespace game_framework {
    class Map;
    Player::Player(bool boy)
    {
        score = 0;
        is_boy = boy;
        x = 0;
        y = 0;
        moving_vertical = DIRECTION_NONE;
        moving_horizontal = DIRECTION_NONE;
        velocity = INITIAL_VELOCITY;
        is_visible = true;
        is_die = false;
        reach_exit = false;
    };
    Player::~Player()
    {
    };
    void Player::LoadBitmapPlayer()//載入图形
    {
        if (is_boy)
        {
            //bitmap.LoadBitmapA("RES\\player\\boy_static.bmp", RGB(255, 255, 255));
            animation_static.AddBitmap("RES\\player\\boy_static0.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\boy_static1.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\boy_static2.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\boy_static3.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\boy_static4.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\boy_left0.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\boy_left1.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\boy_left2.bmp", RGB(0, 0, 0));
```

```
            animation_left.AddBitmap("RES\\player\\boy_left3.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\boy_left4.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\boy_right0.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\boy_right1.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\boy_right2.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\boy_right3.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\boy_right4.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\boy_up0.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\boy_up1.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\boy_up2.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\boy_down0.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\boy_down1.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\boy_down2.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\boy_down3.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\boy_down4.bmp", RGB(0, 0, 0));
            charactor_ani = &animation_static;
            //charactor_ani->SetDelayCount(2);
        }
        else
        {
            //bitmap.LoadBitmapA("RES\\player\\girl_static.bmp", RGB(255, 255, 255));
            animation_static.AddBitmap("RES\\player\\girl_static0.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\girl_static1.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\girl_static2.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\girl_static3.bmp", RGB(0, 0, 0));
            animation_static.AddBitmap("RES\\player\\girl_static4.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\girl_left0.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\girl_left1.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\girl_left2.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\girl_left3.bmp", RGB(0, 0, 0));
            animation_left.AddBitmap("RES\\player\\girl_left4.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\girl_right0.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\girl_right1.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\girl_right2.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\girl_right3.bmp", RGB(0, 0, 0));
            animation_right.AddBitmap("RES\\player\\girl_right4.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\girl_up0.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\girl_up1.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\girl_up2.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\girl_up3.bmp", RGB(0, 0, 0));
            animation_up.AddBitmap("RES\\player\\girl_up4.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\girl_down0.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\girl_down1.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\girl_down2.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\girl_down3.bmp", RGB(0, 0, 0));
            animation_down.AddBitmap("RES\\player\\girl_down4.bmp", RGB(0, 0, 0));
            charactor_ani = &animation_static;
            //charactor_ani->SetDelayCount(2);
        }
};
void Player::OnMove()
{
    animation_static.OnMove();
    animation_left.OnMove();
    animation_right.OnMove();
    animation_down.OnMove();
    animation_up.OnMove();
}
void Player::OnShow()//显示
{
    animation_static.SetTopLeft(x, y - PLAYER_GIRD_PIXEL);
    animation_down.SetTopLeft(x, y - PLAYER_GIRD_PIXEL);
    animation_up.SetTopLeft(x, y - PLAYER_GIRD_PIXEL);
    animation_left.SetTopLeft(x , y - PLAYER_GIRD_PIXEL);
    animation_right.SetTopLeft(x - PLAYER_GIRD_PIXEL, y - PLAYER_GIRD_PIXEL);
    if (is_visible)
    {
        charactor_ani->SetDelayCount(2);
        charactor_ani->OnShow();
```

```
	}
};
int	Player::GetX1()
{
	return x;
}
int Player::GetY1()
{
	return y;
}
int	Player::GetX2()
{
	return x + PLAYER_GIRD_PIXEL;
}
int Player::GetY2()
{
	return y + PLAYER_GIRD_PIXEL;
}
int Player::GetVelocity()
{
	return velocity;
}
int Player::GetStep()
{
	return PLAYER_STEP_PIXEL;
}
void Player::SetTopLeft(int top, int left)//  设定左上角坐标
{
	x = left;
	y = top;
}
void Player::SetVerticalState(int state)//设定垂直状态
{
	moving_vertical = state;
}
void Player::SetHorizontalState(int state)//设定水平状态
{
	moving_horizontal = state;
}
int Player::GetVerticalState()//获取垂直状态
{
	return moving_vertical;
}
int Player::GetHorizontalState()//获取水平状态
{
	return moving_horizontal;
}
CAnimation& Player::GetStatic()
{
	return animation_static;
}
CAnimation& Player::GetLeft()
{
	return animation_left;
}
CAnimation& Player::GetRight()
{
	return animation_right;
}
CAnimation& Player::GetUp()
{
	return animation_up;
}
CAnimation& Player::GetDown()
{
	return animation_down;
}
void Player::SetAni(CAnimation& ani)
{
```

```
            charactor_ani = &ani;
        }
};
```

12. Menu.h

```
#pragma once
namespace game_framework {
    class CMovingBitmap;
    class Menu
    {
    public:
        Menu();
        void LoadItemBitmap();
        void OnShow(int menuType);
        void OnMove(int menuType);
    private:
        CMovingBitmap bitmap_pause0;
        CMovingBitmap bitmap_pause1;
        CMovingBitmap bitmap_over;
        int top = 0;
        int left = 0;
    };
}
```

13. Menu.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Menu.h"
namespace game_framework
{
    Menu::Menu()
    {
    }
    void Menu::LoadItemBitmap()
    {
        bitmap_pause0.LoadBitmapA("RES\\menu\\pause0.bmp");
        bitmap_pause1.LoadBitmapA("RES\\menu\\pause1.bmp");
        bitmap_over.LoadBitmapA("RES\\menu\\over.bmp");
    }
    void Menu::OnShow(int menuType)
    {
        switch (menuType)
        {
        case 0:
            bitmap_pause0.SetTopLeft(left, top);
            bitmap_pause0.ShowBitmap();
            break;
        case 1:
            bitmap_pause1.SetTopLeft(left,top);
            bitmap_pause1.ShowBitmap();
            break;
        case 2:
            bitmap_over.SetTopLeft(left, top);
            bitmap_over.ShowBitmap();
            break;
        default:
            break;
        }
    }
    void Menu::OnMove(int menuType)
    {
    }
}
```

14. Item.h

```
#pragma once
namespace game_framework {
	class CGameStateRun;
	class Item
	{
	public:
		Item();
		int GetType();				//获取类型
		int GetId();				//获取 id
		virtual void LoadItemBitmap() = 0;  //加载位图
		virtual void OnShow() = 0;					//显示
		virtual void OnMove() = 0;					//移动
		void SetTopLeft(int top, int left);//  设定左上角坐标
		int	GetX1();				//左上角 x 坐标
		int	GetY1();				//左上角 y 坐标
		int	GetX2();				//右下角 x 坐标
		int	GetY2();				//右下角 y 坐标
		bool HitRectangle(int x1, int y1, int x2, int y2);//是否相撞
		bool is_accessible;			//是否可通过
		bool is_visibale;			//是否可见
	protected:
		int type;					//类别
		int id;						//id
		int x, y;					//左上角坐标
		int width;					//宽
		int height;					//高
	};
}
```

15. Item.cpp

```
#include "stdafx.h"
#include "Item.h"
namespace game_framework
{
	int global_id;//全局 id
	Item::Item()
	{
		id = global_id++;
		//TRACE("item id=%d\n", id);
	}
	int Item::GetType()
	{
		return type;
	}
	int Item::GetId()
	{
		return id;
	}
	void Item::SetTopLeft(int top, int left)
	{
		x = left;
		y = top;
	}
	int	Item::GetX1()
	{
		return x;
	}
	int	Item::GetY1()
	{
		return y;
	}
	int	Item::GetX2()
	{
		return x + width;
	}
	int	Item::GetY2()
	{
```

```
		return y + height;
	}
	bool Item::HitRectangle(int tx1, int ty1, int tx2, int ty2)
	{
		int x1 = x;
		int y1 = y;
		int x2 = GetX2();
		int y2 = GetY2();
		return (tx2 > x1 && tx1 < x2 && ty2 > y1 && ty1 < y2);
	}
}
```

16. Box.h

```
#pragma once
#include "Item.h"
namespace game_framework {
	class CMovingBitmap;
	class Box : public Item
	{
	public:
		Box();
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
	private:
		CMovingBitmap bitmap;
	};
}
```

17. Box.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Box.h"
#include "CGameStateRun.h"
namespace game_framework
{
	Box::Box()
	{
		width = 64;
		height = 64;
		is_accessible = false;
	}
	void Box::LoadItemBitmap()
	{
		bitmap.LoadBitmapA("RES\\box\\box.bmp", RGB(255, 255, 255));
	}
	void Box::OnShow()
	{
		bitmap.SetTopLeft(x, y);
		bitmap.ShowBitmap();
	}
	void Box::OnMove()
	{
	}
}
```

18. Diamond.h

```
#pragma once
#include "Item.h"
namespace game_framework {
	class CMovingBitmap;
	class Diamond : public Item
	{
	public:
```

```
        Diamond(int type);
        void LoadItemBitmap();
        void OnShow();
        void OnMove();
    private:
        CMovingBitmap bitmap;
    };
}
```

19. Diamond.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Diamond.h"
#include "CGameStateRun.h"
namespace game_framework
{
    Diamond::Diamond(int type)
    {
        this->type = type;
        width = 32 ;
        height = 64 ;//鑽石高度增加
        is_accessible = true;
    }
    void Diamond::LoadItemBitmap()
    {
        switch (type)
        {
        case 200:
        {
            bitmap.LoadBitmapA("RES\\diamond\\fire.bmp", RGB(255, 255, 255));
            break;
        }
        case 201:
        {
            bitmap.LoadBitmapA("RES\\diamond\\water.bmp", RGB(255, 255, 255));
            break;
        }
        default:
            break;
        }
    }
    void Diamond::OnShow()
    {
        bitmap.SetTopLeft(x, y);
        bitmap.ShowBitmap();
    }
    void Diamond::OnMove()
    {
    }
}
```

20. Door.h

```
#pragma once
#include"Item.h"
namespace game_framework {
    enum DOOR_STATUS
    {
        //DOOR_DEFAULT,
        DOOR_ENTER,
        DOOR_LEAVE
    };
    class CMovingBitmap;
    class Door : public Item
    {
```

```
	public:
		Door(int type);
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
		//void AnimationStart();
		//void AnimationReset();
		void AnimationOpenDoor();
		void AnimationCloseDoor();
		int status = DOOR_LEAVE;
	private:
		//CAnimation bitmap;
		void AddAnimationBitmap(string path);
		CAnimation* bitmap_ptr;
		CAnimation bitmap_open;
		CAnimation bitmap_close;
		bool anime_play = false;
		bool anime_open = true;
	};
}
```

21. Door.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Door.h"
#include "CGameStateRun.h"
namespace game_framework
{
	Door::Door(int type)
	{
		this->type = type;
		width = 96 - 1;
		height = 96 - 1;
		is_accessible = true;
		//设定初始动画
		bitmap_ptr = &bitmap_open;
	}
	void Door::LoadItemBitmap()
	{
		switch (type)
		{
		case 400:
		{
			string path = "RES\\door\\FinishBoy";
			AddAnimationBitmap(path);
			break;
		}
		case 401:
		{
			string path = "RES\\door\\FinishGirl";
			AddAnimationBitmap(path);
			break;
		}
		default:
			break;
		}
	}
	void Door::OnShow()
	{
		bitmap_ptr->SetTopLeft(x, y);
		bitmap_ptr->OnShow();
	}
	void Door::OnMove()
	{
		if (anime_play)//如果动画在播放中
```

```
		{
			if (!bitmap_ptr->IsFinalBitmap())//如果没有播放到最后一张
			{
				if (anime_open)//判断动画类型
				{
					bitmap_ptr = &bitmap_open;//开门动画
				}
				else
				{
					bitmap_ptr = &bitmap_close;//关门动画
				}
				bitmap_ptr->SetDelayCount(2);
				bitmap_ptr->OnMove();//继续切换动画
			}
			else
			{
				anime_play = false;//停止播放
				if (anime_open)//判断刚才结束的动画类型
				{
					bitmap_ptr = &bitmap_close;//开门动画结束，则切换到关门动画
				}
				else
				{
					bitmap_ptr = &bitmap_open;//关门动画结束，则切换到开门动画
				}
				bitmap_ptr->Reset();//重置动画
			}
		}
	}
	void Door::AddAnimationBitmap(string path)
	{
		string extension = ".bmp";
		string bitmapPath = "";
		for (int i = 0; i <= 21; i++) {
			bitmapPath = path + to_string(i) + extension;
			char* ptr1 = (char*)bitmapPath.c_str();
			bitmap_open.AddBitmap(ptr1, RGB(255, 255, 255));
			bitmapPath = path + to_string(21-i) + extension;
			char* ptr2 = (char*)bitmapPath.c_str();
			bitmap_close.AddBitmap(ptr2, RGB(255, 255, 255));
		}
	}
	void Door::AnimationOpenDoor()
	{
		//仅被触发一次
		//TRACE("door open\n");
		if (!anime_play)//如果动画未开始播放
		{
			anime_open = true;//切换到开门动画
			anime_play = true;//开始播放
		}
	}
	void Door::AnimationCloseDoor()
	{
		//仅被触发一次
		//TRACE("door close\n");
		if (!anime_play)//如果动画未开始播放
		{
			anime_open = false;//切换到关门动画
			anime_play = true;//开始播放
		}
	}
}
```

22. Platform.h

```
#pragma once
#include"Item.h"
namespace game_framework {
	class CMovingBitmap;
```

```
	class Platform : public Item
	{
		//两种平台：水平（502 platform_horizon）、垂直（503 platform_vertical）
		//初始位置：init_1_x、init_1_y
		//激活位置：init_2_x、init_2_y
	public:
		Platform(int type,int init_1_x,int init_1_y,int init_2_x,int init_2_y);
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
		//平台被开关触发
		void Trigger(bool is_on);
	private:
		CMovingBitmap bitmap;
		//平台状态
		int status;
		//初始位置
		int init_1_x;
		int init_1_y;
		int init_2_x;
		int init_2_y;
	};
}
```

23. Platform.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Platform.h"
#include "CGameStateRun.h"
namespace game_framework
{
	Platform::Platform(int type, int init_1_x, int init_1_y, int init_2_x, int init_2_y)
	{
		this->init_1_x = init_1_x * MAP_GIRD_PIXEL;
		this->init_1_y = init_1_y * MAP_GIRD_PIXEL;
		this->init_2_x = init_2_x * MAP_GIRD_PIXEL;
		this->init_2_y = init_2_y * MAP_GIRD_PIXEL;
		this->x = this->init_1_x;
		this->y = this->init_1_y;
		this->type = type;
		is_accessible = false;
		switch (type)
		{
		case 502://水平
		{
			width = 32 * 4;
			height = 32;
			break;
		}
		case 503://垂直
		{
			width = 32;
			height = 32 * 4;
			break;
		}
		default:
			break;
		}
	}
	void Platform::LoadItemBitmap()
	{
		switch (type)
		{
		case 502://水平
		{
```

```
                bitmap.LoadBitmapA("RES\\platform\\horizon.bmp", RGB(0, 0, 0));
                break;
            }
            case 503:////垂直
            {
                bitmap.LoadBitmapA("RES\\platform\\vertical.bmp", RGB(0, 0, 0));
                break;
            }
            default:
                break;
            }
        }
        void Platform::OnShow()
        {
            bitmap.SetTopLeft(x, y);
            bitmap.ShowBitmap();
        }
        void Platform::OnMove()
        {
        }
        void Platform::Trigger(bool is_on)
        {
            if (!is_on)//off 为默认状态
            {
                x = this->init_1_x;
                y = this->init_1_y;
            }
            else
            {
                x = this->init_2_x;
                y = this->init_2_y;
            }
        }
}
```

24. Pool.h

```
#pragma once
#include "Item.h"
#include <memory>
namespace game_framework {
    class Pool :public Item
    {
    public:
        Pool(int type);
        void LoadItemBitmap();
        void OnShow();
        void OnMove();
    private:
        CAnimation animation_pool;
        CMovingBitmap bitmap_wall;
        void AddAnimationBitmap(string path);
    };
}
```

25. Pool.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Pool.h"
#include "CGameStateRun.h"
namespace game_framework
{
    class Map;
    Pool::Pool(int type)
    {
```

```
	this->type = type;
	width = 32 - 1;
	height = 32 - 1;
	is_accessible = false;
}
void Pool::LoadItemBitmap()
{
	bitmap_wall.LoadBitmapA("RES\\wall.bmp");
	switch (type)
	{
	case 2:
	{
		string path = "RES\\pool\\fire_left_";
		AddAnimationBitmap(path);
		break;
	}
	case 3:
	{
		string path = "RES\\pool\\fire_right_";
		AddAnimationBitmap(path);
		break;
	}
	case 4:
	{
		string path = "RES\\pool\\water_left_";
		AddAnimationBitmap(path);
		break;
	}
	case 5:
	{
		string path = "RES\\pool\\water_right_";
		AddAnimationBitmap(path);
		break;
	}
	case 6:
	{
		string path = "RES\\pool\\toxic_left_";
		AddAnimationBitmap(path);
		break;
	}
	case 7:
	{
		string path = "RES\\pool\\toxic_right_";
		AddAnimationBitmap(path);
		break;
	}
	case 300:
	{
		string path = "RES\\pool\\fire_";
		AddAnimationBitmap(path);
		break;
	}
	case 301:
	{
		string path = "RES\\pool\\water_";
		AddAnimationBitmap(path);
		break;
	}
	case 302:
	{
		string path = "RES\\pool\\toxic_";
		AddAnimationBitmap(path);
		break;
	}
	default:
		break;
	}
}
void Pool::OnShow()
```

```
	{
		bitmap_wall.SetTopLeft(x, y);//先画墙
		bitmap_wall.ShowBitmap();
		animation_pool.SetTopLeft(x, y - 21);//提高半格画水池
		animation_pool.OnShow();
	}
	void Pool::OnMove()
	{
		animation_pool.SetDelayCount(2); //設定速度
		animation_pool.OnMove();
	}
	void Pool::AddAnimationBitmap(string path)
	{
		string extension = ".bmp";
		for (int i = 0; i <= 14; i++) {
			string bitmapPath = path + to_string(i) + extension;
			char* p = (char*)bitmapPath.c_str();
			animation_pool.AddBitmap(p, RGB(0, 0, 0));
		}
	}
}
```

26. Switch.h

```
#pragma once
#include"Item.h"
#include<memory>
namespace game_framework {
	enum SWITCH_STATUS
	{
		SWITCH_RELEASE,
		SWITCH_PRESS
	};
	class CMovingBitmap;
	class Platform;
	class Switch : public Item
	{
		//两种开关：按钮（非持续作用 500 switch_button）、拉杆（持续作用 501 switch_stick）
//
		//拉杆底座视为墙壁，在拉杆下方一格
	public:
		Switch(int type, int x, int y);
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
		//开关绑定平台
		void Bind(shared_ptr<Platform> platform);
		void Trigger();
		//开关状态
		int status = 0;
		bool is_on = false;
	private:
		CMovingBitmap bitmap_switch_on;
		CMovingBitmap bitmap_switch_off;
		//开关绑定的平台
		shared_ptr<Platform> bind_platform;
	};
}
```

27. Switch.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Switch.h"
#include "CGameStateRun.h"
namespace game_framework
```

```
{
	Switch::Switch(int type, int x, int y)
	{
		this->x = x * MAP_GIRD_PIXEL;
		this->y = y * MAP_GIRD_PIXEL;
		this->type = type;
		width = 32;
		height = 32;
		is_accessible = true;
	}
	void Switch::LoadItemBitmap()
	{
		switch (type)
		{
		case 500://按钮
		{
			bitmap_switch_on.LoadBitmapA("RES\\switch\\button_on.bmp", RGB(0, 0, 0));
			bitmap_switch_off.LoadBitmapA("RES\\switch\\button_off.bmp", RGB(0, 0, 0));
			break;
		}
		case 501://拉杆
		{
			bitmap_switch_on.LoadBitmapA("RES\\switch\\stick_on.bmp", RGB(0, 0, 0));
			bitmap_switch_off.LoadBitmapA("RES\\switch\\stick_off.bmp", RGB(0, 0, 0));
			break;
		}
		default:
			break;
		}
	}
	void Switch::OnShow()
	{
		if (is_on)
		{
			bitmap_switch_on.SetTopLeft(x, y);
			bitmap_switch_on.ShowBitmap();
		}
		else
		{
			bitmap_switch_off.SetTopLeft(x, y);
			bitmap_switch_off.ShowBitmap();
		}
	}
	void Switch::OnMove()
	{
	}
	void Switch::Bind(shared_ptr<Platform> platform)
	{
		this->bind_platform = platform;
	}
	void Switch::Trigger()
	{
		//触发平台
		bind_platform->Trigger(is_on);
	}
}
```

28. Tips.h

```
#pragma once
#include "Item.h"
namespace game_framework {
	class CMovingBitmap;
	class Tips : public Item
	{
	public:
		Tips(string name, int x, int y);
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
```

```
	private:
		CMovingBitmap bitmap;
		string name;
	};
}
```

29. Tips.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Tips.h"
#include "CGameStateRun.h"
namespace game_framework
{
	Tips::Tips(string name,int x,int y)
	{
		this->x = x * MAP_GIRD_PIXEL;
		this->y = y * MAP_GIRD_PIXEL;
		this->type = 1000;
		this->name = name;
		width = 0;
		height = 0;
		is_accessible = true;
	}
	void Tips::LoadItemBitmap()
	{
		string path = "RES\\tips\\" + name + ".bmp";
		char* p = (char*)path.c_str();
		bitmap.LoadBitmapA(p, RGB(0, 0, 0));
	}
	void Tips::OnShow()
	{
		bitmap.SetTopLeft(x, y);
		bitmap.ShowBitmap();
	}
	void Tips::OnMove()
	{
	}
}
```

30. Wall.h

```
#pragma once
#include "Item.h"
namespace game_framework {
	class CMovingBitmap;
	class Wall : public Item
	{
	public:
		Wall();
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
	private:
		CMovingBitmap bitmap;
	};
}
```

31. Wall.cpp

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

```
namespace game_framework
{
	Wall::Wall()
	{
		type = 1;
		width = 32 - 1;
		height = 32 - 1;
		is_accessible = false;
	}
	void Wall::LoadItemBitmap()
	{
		bitmap.LoadBitmapA("RES\\wall.bmp");
	}
	void Wall::OnShow()
	{
		bitmap.SetTopLeft(x, y);
		bitmap.ShowBitmap();
	}
	void Wall::OnMove()
	{
	}
}
```

32. Wind.h

```
#pragma once
#include "Item.h"
namespace game_framework {
	class CMovingBitmap;
	class Wind : public Item
	{
	public:
		Wind();
		void LoadItemBitmap();
		void OnShow();
		void OnMove();
	private:
		CAnimation bitmap;
		CAnimation bitmap_wind;
		CMovingBitmap bitmap_wall;
	};
}
```

33. Wind.cpp

```
#include "stdafx.h"
#include "Resource.h"
#include <mmsystem.h>
#include <ddraw.h>
#include "audio.h"
#include "gamelib.h"
#include "Wind.h"
namespace game_framework
{
	Wind::Wind()
	{
		type = 600;
		width = 96;
		height = 32;
		is_accessible = false;
	}
	void Wind::LoadItemBitmap()
	{
		bitmap_wall.LoadBitmapA("RES\\wall.bmp");
		bitmap.AddBitmap("RES\\wind\\0.bmp",RGB(255,255,255));
		bitmap.AddBitmap("RES\\wind\\1.bmp", RGB(255, 255, 255));
		bitmap.AddBitmap("RES\\wind\\2.bmp", RGB(255, 255, 255));
		bitmap.AddBitmap("RES\\wind\\3.bmp", RGB(255, 255, 255));
		string path = "RES\\wind_effect\\wind_effect00";
		string extension = ".bmp";
```

```
		for (int i = 0; i <= 29; i++) {
			string bitmapPath = path + to_string(i) + extension;
			char* p = (char*)bitmapPath.c_str();
			bitmap_wind.AddBitmap(p, RGB(0, 0, 0));
		}
	}
	void Wind::OnShow()
	{
		bitmap_wall.SetTopLeft(x, y);//	牆壁
		bitmap_wall.ShowBitmap();
		bitmap_wall.SetTopLeft(x + 32, y);//	牆壁
		bitmap_wall.ShowBitmap();
		bitmap_wall.SetTopLeft(x + 64, y);//	牆壁
		bitmap_wall.ShowBitmap();
		bitmap.SetTopLeft(x, y);
		bitmap.OnShow();
		bitmap.SetDelayCount(1);
		bitmap_wind.SetDelayCount(1);
		bitmap_wind.SetTopLeft(x, y-192);
		bitmap_wind.OnShow();
	}
	void Wind::OnMove()
	{
		bitmap.OnMove();
		bitmap_wind.OnMove();
	}
}
```