
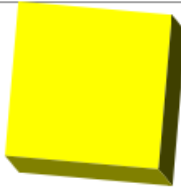
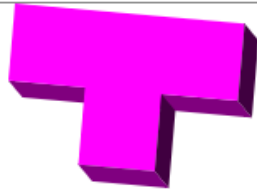
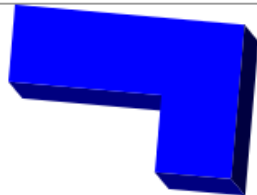


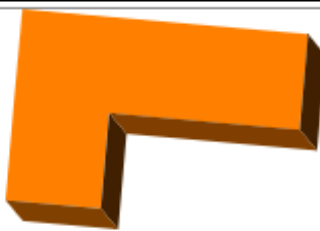
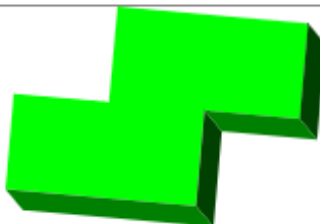
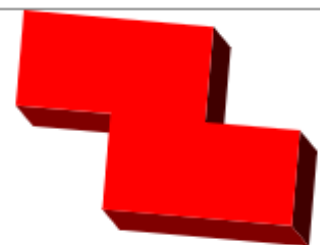
OOP期末專題 俄羅斯的方塊

組員：賴林鴻、林文約

遊戲規則

- 俄羅斯方塊是由數種四格骨牌構成，如圖。

	L	最多可消 4格
	O	最多可消 2格
	T	最多可消 3格
	J	最多可消 3格

	L	最多可消 3格
	S	最多可消 2格
	Z	最多可消 2格

- 目標:
- 五格骨牌

遊戲規則

- 玩家可以做的操作有：以90度為單位旋轉方塊，以格子為單位左右移動方塊，以及讓方塊加速落下。
- 方塊下落到區域最下方或是著落到其他方塊上無法再向下移動時，就會固定在該處，然後新的方塊出現在區域上方開始落下。
- 當區域中某一橫行的格子全部由方塊填滿時，則該列會被消除並成為玩家的得分。同時消除的行數越多，得分指數級上升。
- 當固定的方塊堆到區域最頂端而無法消除層數時，則遊戲結束。

介面

- 使用Terminal作為開發環境(Portability)
- 利用畫面刷新製作動畫
- 思考如何在非windows作業系統下開發程式
- 使用Cloud 9作為開發介面

執行畫面節錄

The image shows a title screen for a game called "TETRIS BATTLE". The title is displayed in a pixelated, blocky font. "TETRIS" is in a light blue color and "BATTLE" is in a dark grey color. Both words are set against a solid red rectangular background. This red rectangle is centered on a black background.

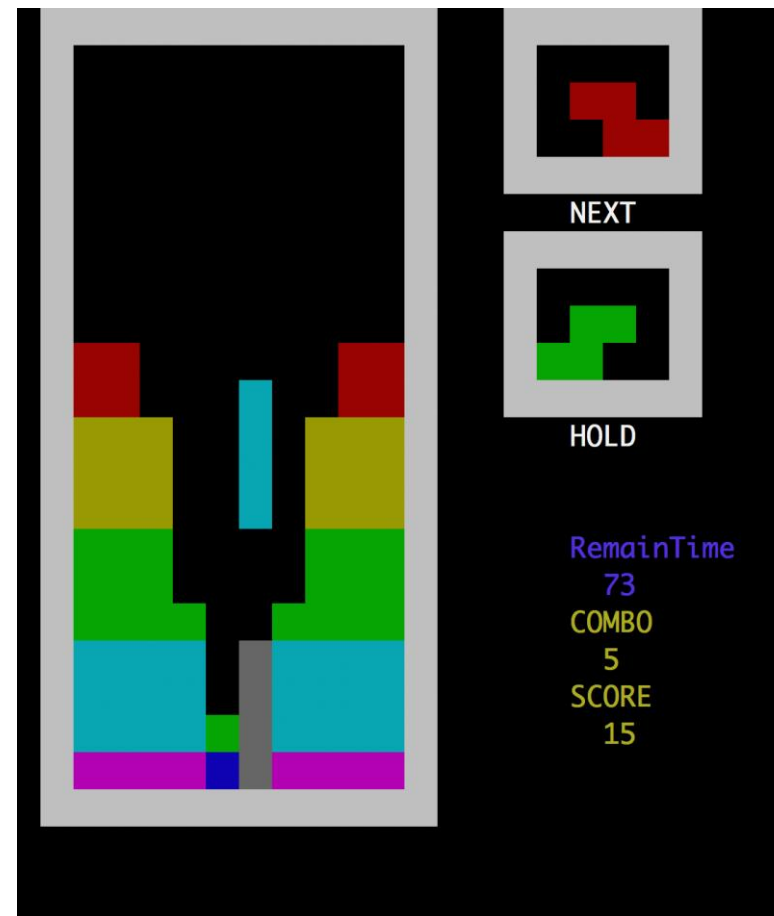
TETRIS
BATTLE

Welcome to play Tetris Battle!
This game was developed by Henry Lai & Ken Lin!
Enjoy and have fun!
Date: 2018/05/23

Type your name to start: ■

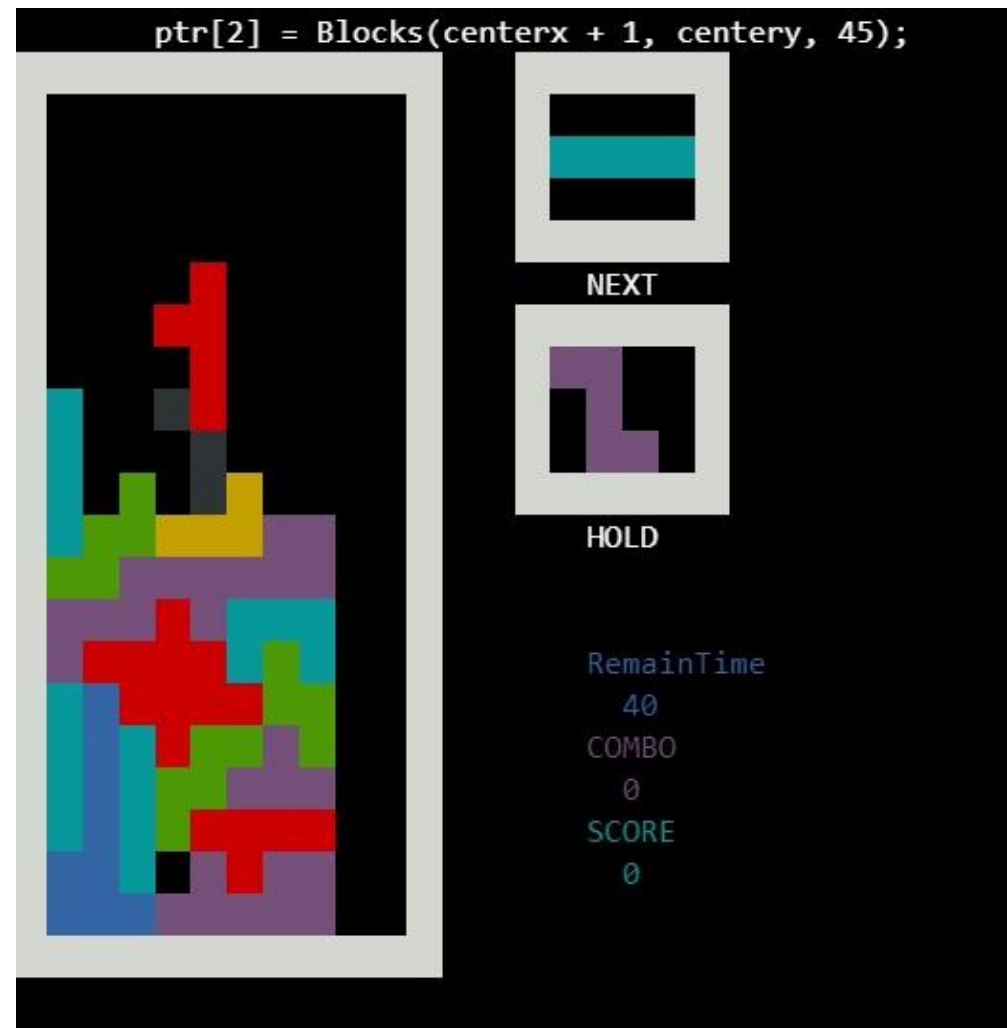
遊戲介面

- 方向鍵：
 - 控制方塊左右移動
 - $\uparrow \leftarrow \downarrow \rightarrow$
- 空白鍵：
 - 讓方塊直接落下
- 字元c：
 - 保留手上方塊，並把保留區的方塊拿到手上
- 字元p：
 - 暫停遊戲



遊戲介面

- 五格方塊
- 超刺激、訓練腦力與空間感

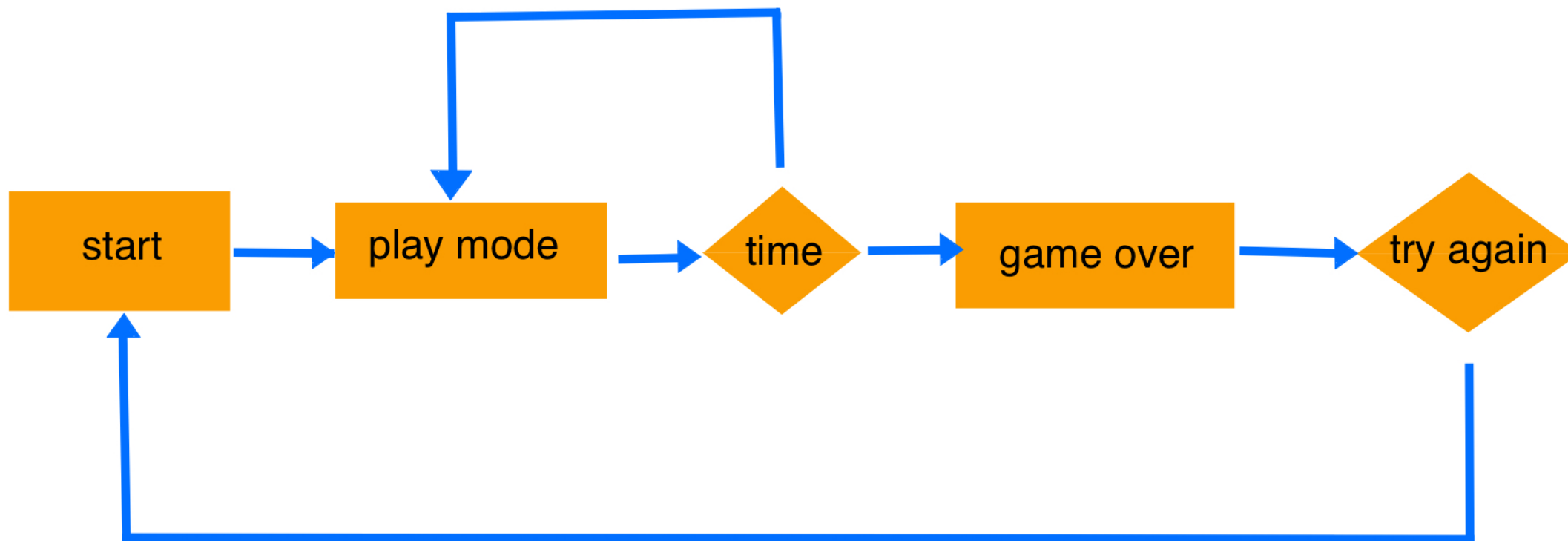


遊戲介面

- 排行榜

RANK				
Rank	Name	Score	H_combo	
1	henryLai	93	8	Your Score is : 0
2	123	66	11	Your Highest Combo is : 0
3	123	55	5	
4	Sara	50	2	Do You Want To Try Again?
5	23	28	3	1)Try Again!
6	Fan	20	3	2)Back to Menu
7	123	6	7	3)End of Game
8	Fan	6	3	
9	1	0	0	
Type your Choice :				

遊戲流程圖



物件

- Class Game_Opreatoring_System
- Class Score
- Class Timex

遊戲運行系統

分數

計時

- Class Pool
- Class boundary
- Class Move, Blocks and Shape
- Class Shape5
- Class Rank
-

待消方塊池

邊界

方塊與形狀

特殊形狀(五格)

排名

Main function

實做概念:

- ✓ 1. 使用列舉enum
 - 簡單、淺顯易懂
- ✓ 2. While 迴圈
 - 重複執行遊戲
- ✓ 3. Game operating system
 - 透過Game operating system 控制，執行對應的動作
 - 最後依據玩家的結果，回傳對應的 GameStatus

```
enum GameStatus{
    Menu,      // 開頭選單
    Loading,   // 載入畫面
    Playing1,  // 遊戲進行中
    Playing2,  // 遊戲進行中
    Pause,     // 遊戲暫停
    GameOver   // 遊戲結束
};

int main()
{
    GameStatus status = Menu;
    Game_Opreatoring_System OS;

    while(1){
        if ( status == Menu )
            status = OS.menu();

        if ( status == Loading )
            status = OS.loading();

        if ( status == Playing1 )
            status = OS.playing_mode1();

        if ( status == Playing2 )
            status = OS.playing_mode2();

        if ( status == GameOver )
            status = OS.gameover();
    }

    return 0;
}
```

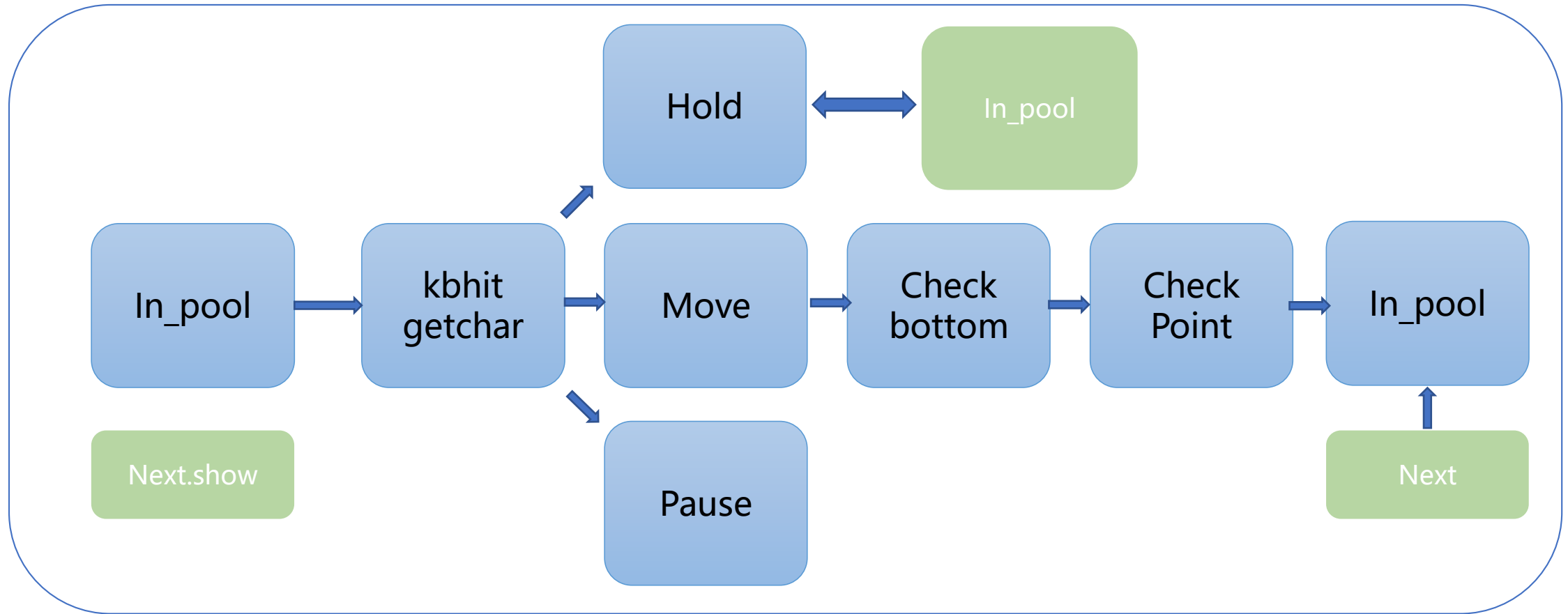
Game Operating System

- 程式的核心class(類似main函式)
- 將各個階段分開實踐
- Menu階段輸入玩家資料，包括名字與遊戲難度與種類
- Loading為等待動畫
- Playing為遊戲主程式，最精華且最重要的一段(寫最久)
- Gameover遊戲結束，顯示分數排名並儲存記錄

```
class Game_Opreatoring_System
{
private:
    int map_type;
    int mode;
    int hardness;
    int person;
    score_system total;
    string name;
public:
    Game_Opreatoring_System(int map = 1,int mod = 1,int har
    :map_type(map), mode(mod), hardness(hard), person(per),

    void show_Tetris_battle();
    GameStatus menu();
    GameStatus loading();
    GameStatus playing_mode1();
    GameStatus playing_mode2();
    GameStatus playing_mode3();
    GameStatus playing_mode4();
    GameStatus gameover();
    void pause();
```

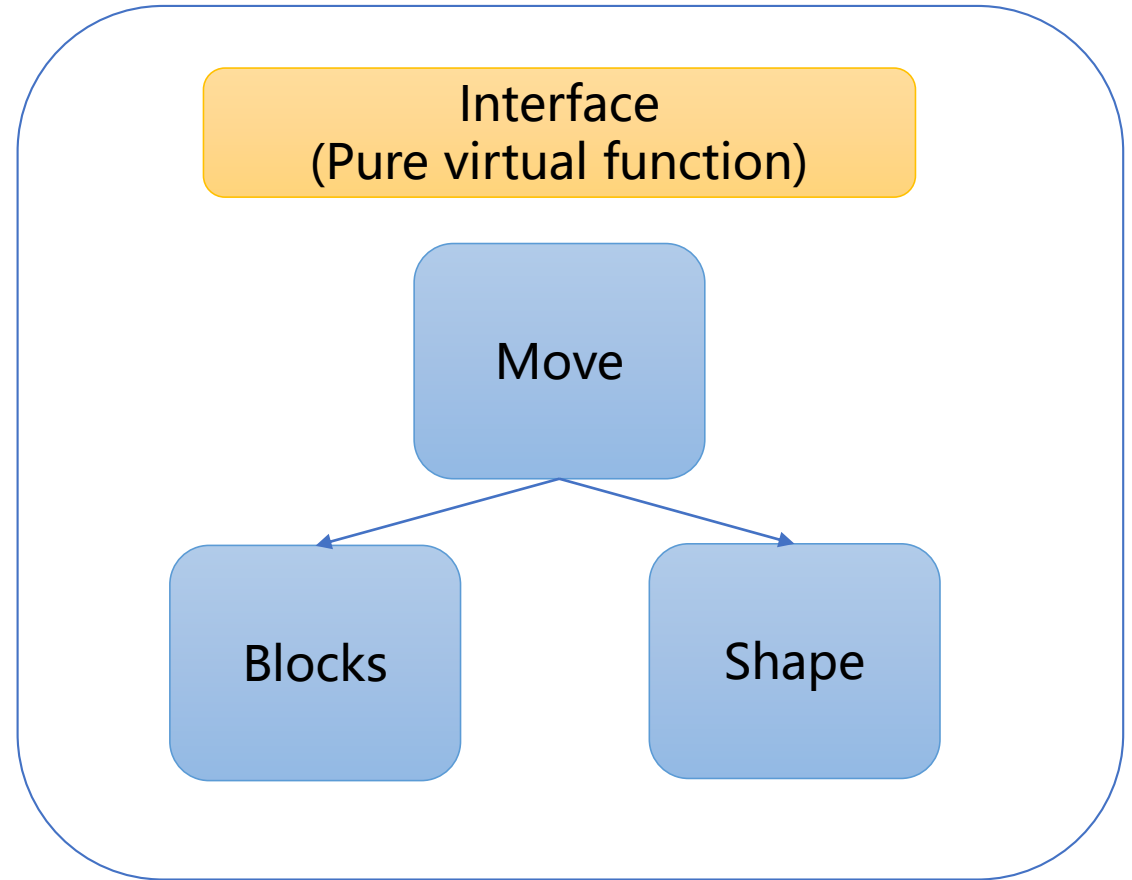
Playing



Move, Blocks and Shape

實做概念:

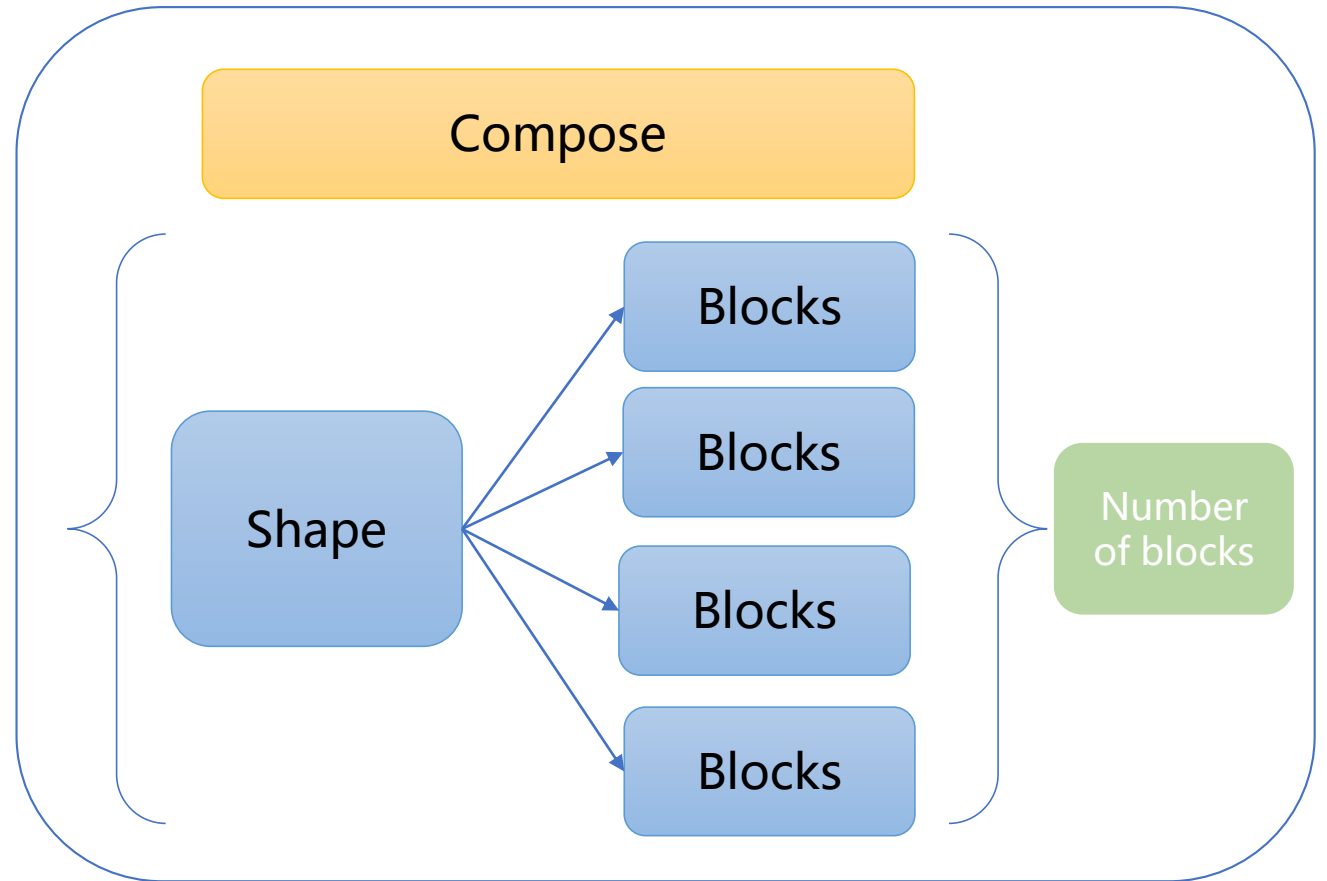
- ✓ 1. Polymorphism
 - 方塊與不同形狀同樣具有移動功能，故建立interface。
- ✓ 2. Pure Virtual Function
 - 讓Blocks and Shape各自override



Shape

實做概念:

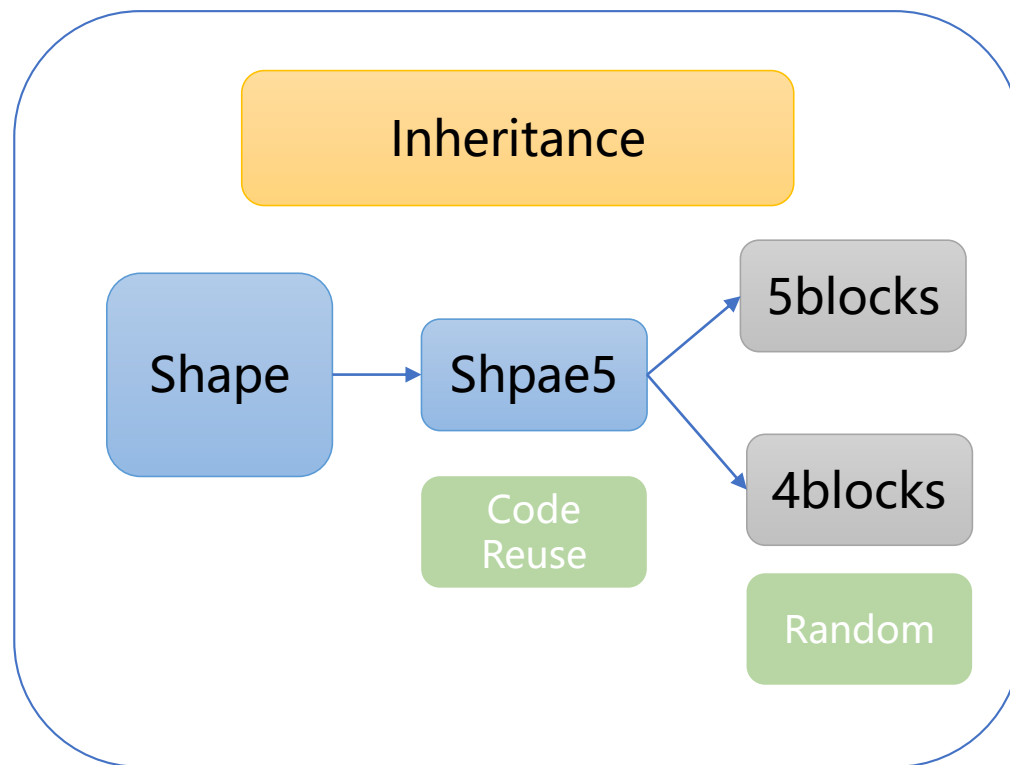
- ✓ 1. Compose
 - 由方塊組成不同形狀
 - Has-a relationship
- ✓ 2. Dynamic Arrays
 - `Blocks *Ptr = new Blocks[NumberOfBlocks]`
- ✓ 3. Copy Constructor
 - 因為成員中有用到指標



Shape5

實做概念:

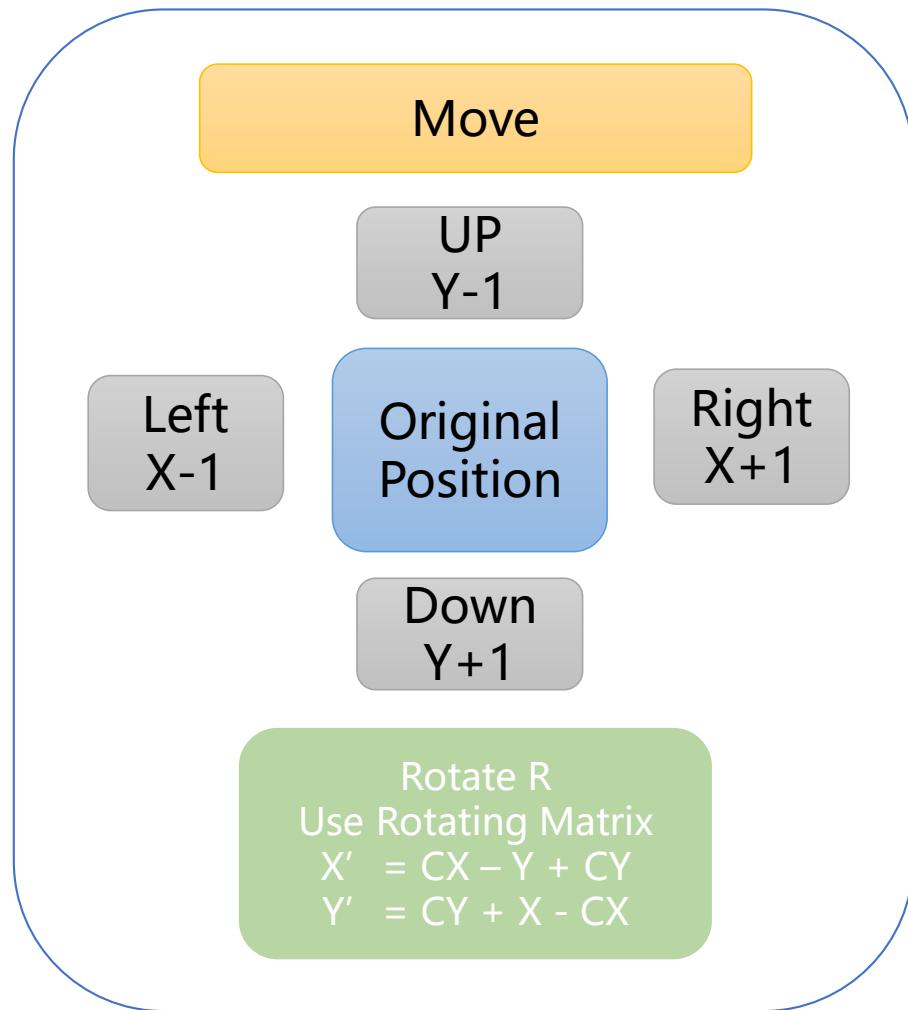
- ✓ 1. Inheritance
 - 五個方塊的形狀
 - 繼承由四個方塊組成的shape
- ✓ 2. Random
 - 隨機產生五個或四個方塊組成的形狀
 - 五個出現的機率決定難度
- ✓ 3. Static member
 - 為了防止五個的形狀連續出現多次
 - 建立static member 紀錄次數



How To Move

實做概念:

- ✓ 1. 上下左右
 - X, Y直接加減
- ✓ 2. 旋轉
 - 使用旋轉矩陣
 - 公式如右圖
- ✓ 3. 邊界問題
 - 旋轉時判斷有無超出邊界
 - 如果有則往內踢



Score System

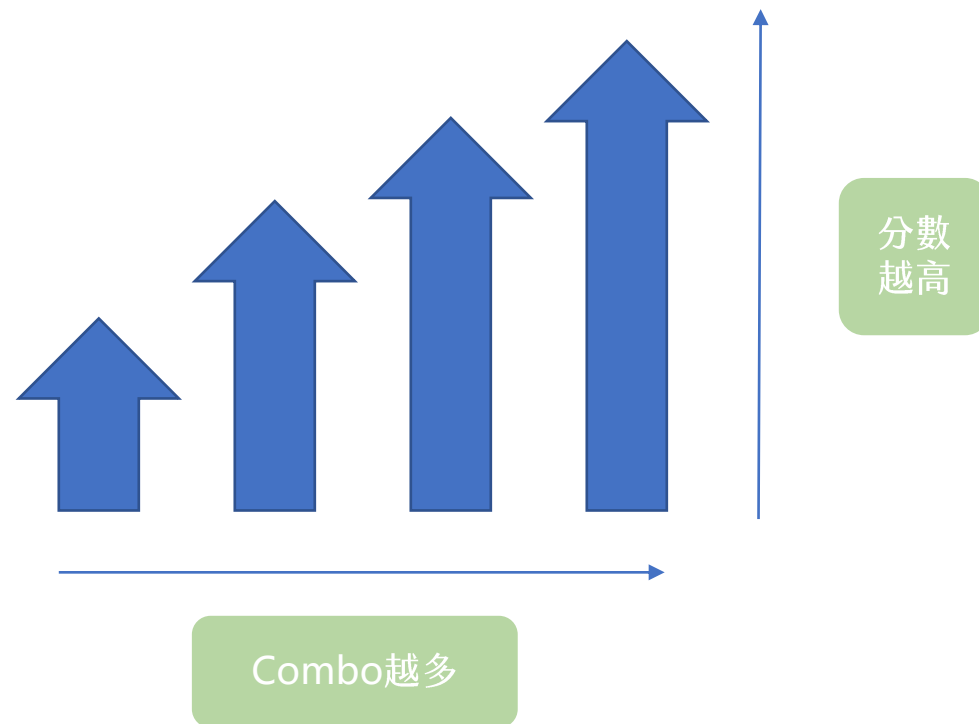
實做概念:

✓ 1. 分數計算

- $\text{Score} += \text{combo數}$
- Combo數越高，分數得越多

✓ 2. 時間計算

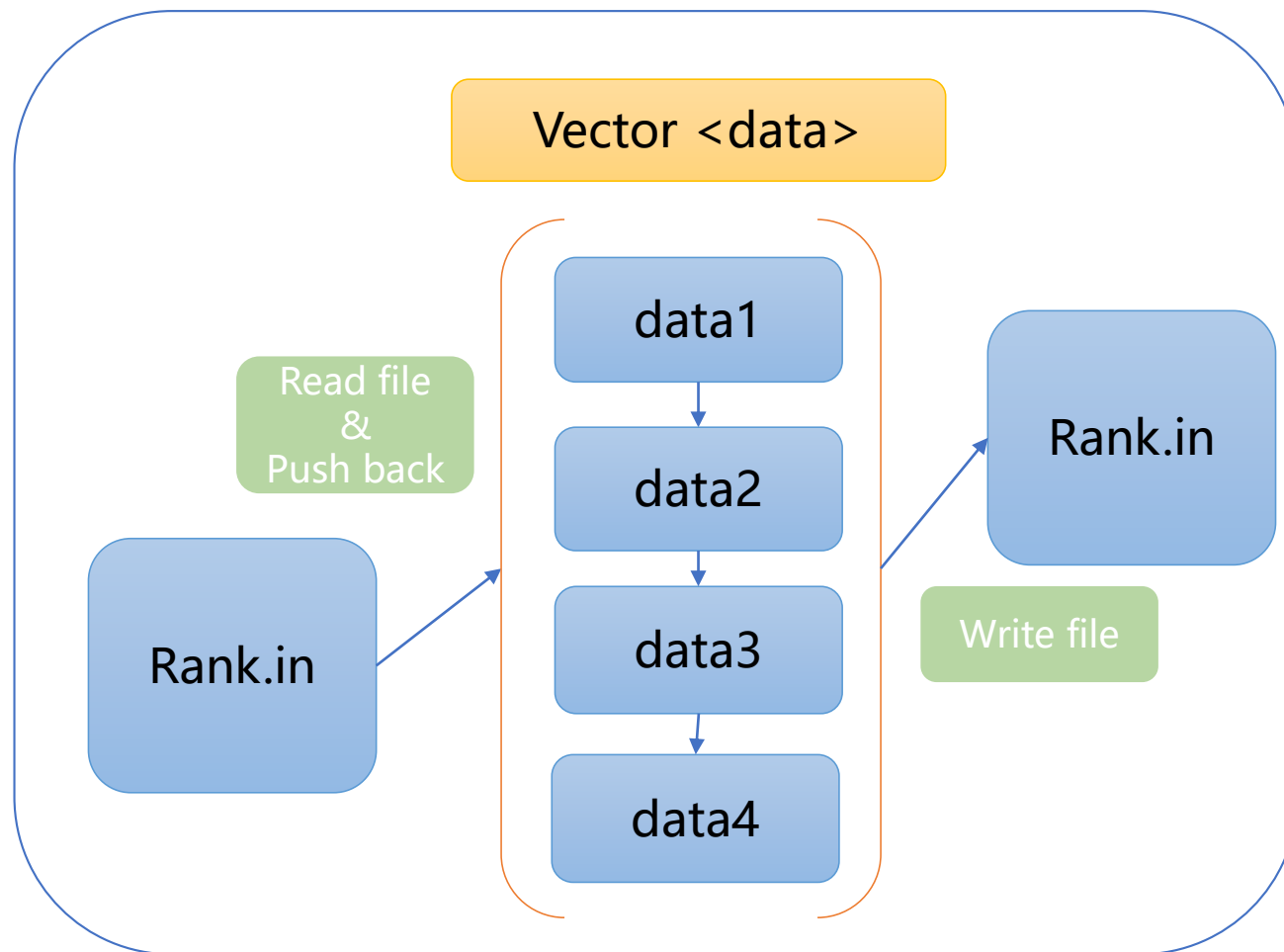
- $\text{Timex} = \text{now} - \text{start};$
- 遊戲限時兩分鐘



Rank

實做概念:

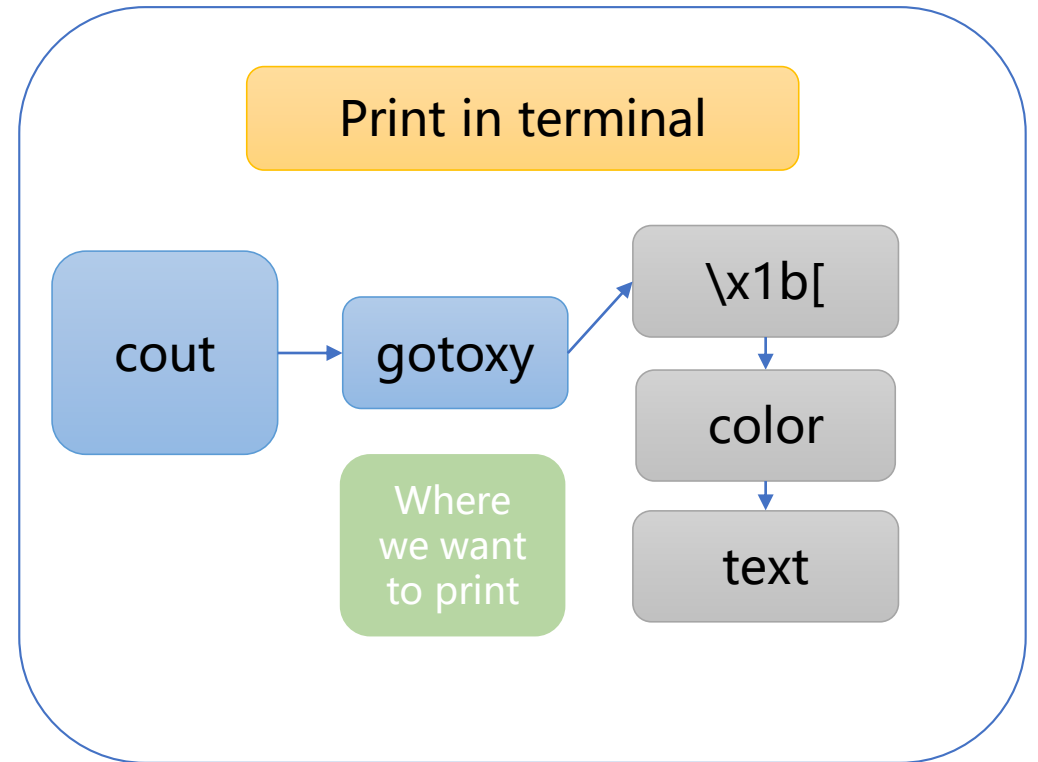
- ✓ 1. File Input / Output
 - 讀取歷來高分紀錄
 - 寫入該次遊戲的成績
- ✓ 2. STL Vector <data>
 - 用來將讀取到的資料暫存
- ✓ 3. STL Algorithm
 - 用來將vector裡的data按照compare function 排序順序



Window(gotoxy & print)

實做概念:

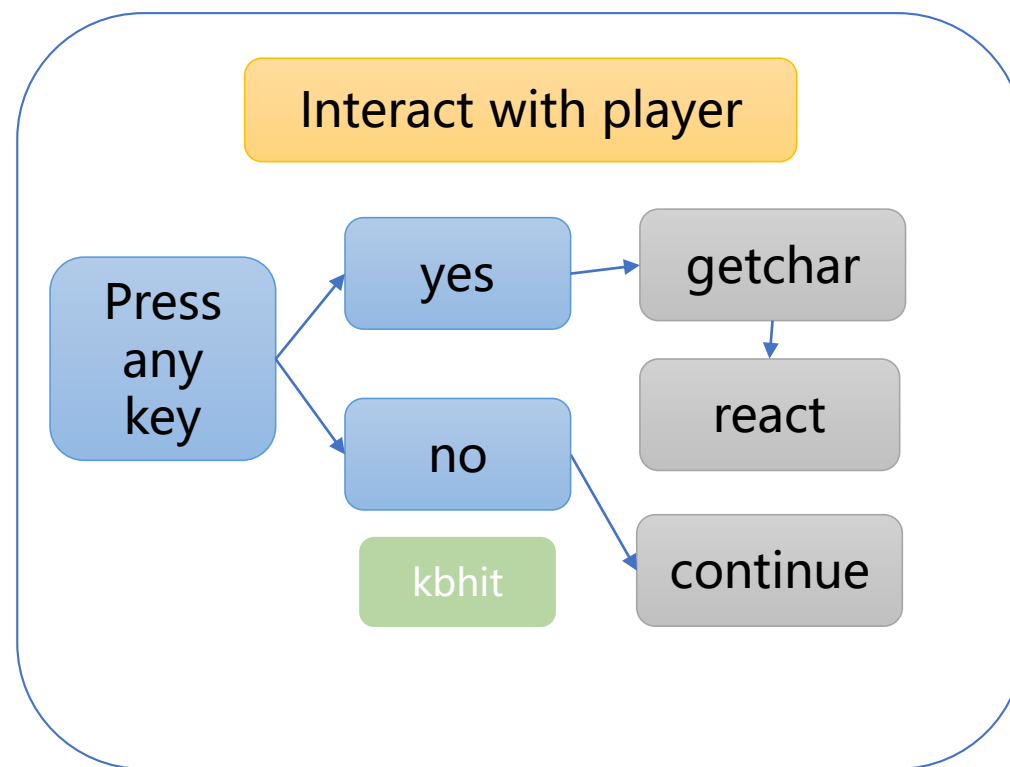
- ✓ 1. Linux終端控制符
 - 可以控制終端的顯示效果，
 - 如清屏，前景背景色設置
- ✓ 2. Operator overloading
 - Overload <<
- ✓ 3. Reference
 - <http://tcspecial.iteye.com/blog/2175865>



Interact (kbhit)

實做概念:

- ✓ 1. `int kbhit(void)`
 - 可以不用因等待輸入而導致程式停擺
- ✓ 2. `getchar()`
 - 讀取字元
- ✓ 3. Reference
 - <https://blog.csdn.net/lanmanc k/article/details/5823562>



物件整理表

Compose

Score

Game
Operating
system

TimeX

Interface (Pure virtual function)

boundary

Pool

Shape

Pool
boundary

Next
boundary

Hold
boundary

Block

Friend

利用繼承 擴充遊戲 不同模式

人數

- 單人模式
- 雙人模式

模式

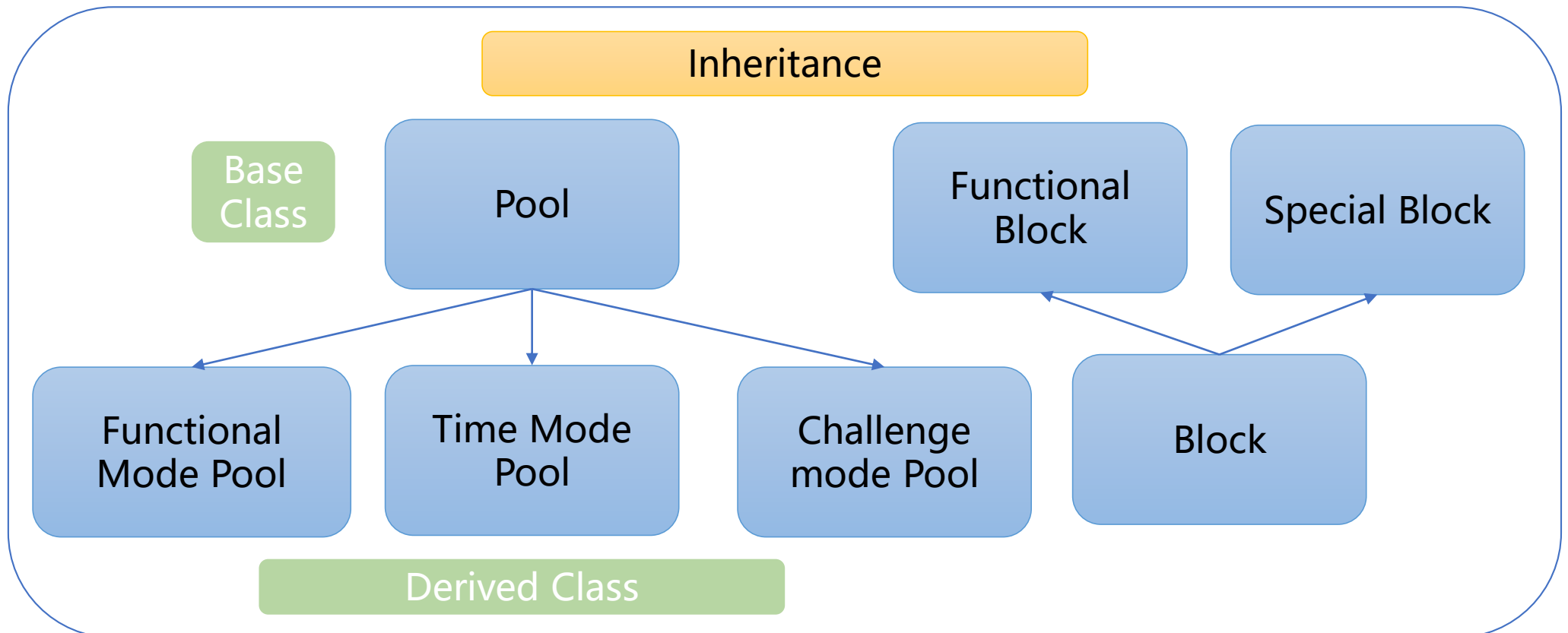
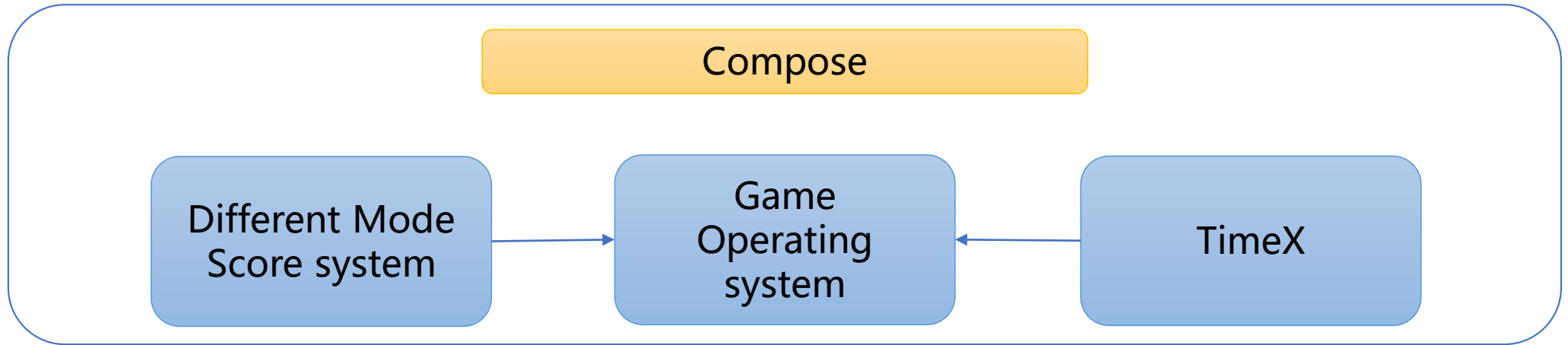
- 一般
- 計時
- 鏡像
- 極限
- 道具

難易度

- 容易
- 普通
- 困難
- 地獄

地圖

- 空白
- 心形
- 金字塔
- 連擊



分工

賴林鴻 50%

- Game OS
- 先寫好四格的俄羅斯方塊遊戲
- 移動與旋轉
- Rank.h
- Map.h
- PPT製作

林文約 50%

- 螢幕輸出與程式互動
- 修正、簡化程式
- 拓展五格的俄羅斯方塊遊戲
- PPT輔助製作

心得

- 這次的期末project我們真的非常用心做。在整個實作的過程當中，我們不斷思考要如何用老師上課教的物件導向的概念方式解決問題。像是在實做shape時，我們就採用compose的方式;而在shpae的特殊形狀，則用繼承的方式。而有趣的是在寫rank的時候，忽然想到期末考的第一題寫法，引進STL概念來儲存資料，確實讓程式漂亮許多。
- 當然在設計當中遇到了很多問題，包括了像是要如何克服邊界問題，旋轉，位置還有如何讓方塊印出來，如何與程式互動...等等，這些問題都是查了很多資料後才一一解決的。
- 此外，我們也建立在原本俄羅斯方塊的基礎下，建立起不同形狀的方塊，讓遊戲變得更好玩，更有挑戰性，我想這也是寫遊戲最大的樂趣之一。

心得

- 不過讓我遺憾的是，要把期末專題打到臻於完美真的需要投入大量的心力，儘管我們從五月中旬就已經開始打了，至今仍有許多問題還需要解決，還要投入更大的心力才能達到完美的目標。
- 但確實，在這過程中學到了很多技巧，並且發現物件導向的威力。
- 最後感謝老師與助教這學期的教學與幫助，讓我在程式"設計"方面成長很多，也感謝我的好partner，雖然只有兩個人，卻依然能盡全力的一起把程式打好。

• 0610831 賴林鴻

心得

- 平常上機或是考試時都因為程式簡單，無法體會到物件導向的好處，這次的期末專題讓我體會到了物件導向的威力。
- 使用物件導向的概念使我們大幅化簡了程式的複雜度，也減少了分工合作的難度。
- 最後我要謝謝我的partner，他神奇地運用了線性代數中旋轉矩陣的概念，旋轉了我們的方塊，如果不是用座標轉換與旋轉矩陣，真不知道這程式會有多複雜。

• 0610851 林文約

附錄 程式碼:

```
class Shape : public Cube, public Move
{
protected:
    unsigned int type;
    int NumberOfBlocks;
    Blocks *ptr;
public:
    Shape();
    Shape(unsigned int a, int b, int c, int d);
    Shape(const Shape &old);

    bool isbottom(Pool &in); // bool isObstacle
    bool isbottom2(Pool &in); // bool isRobot
    bool isLboundary(); // Ken: haven't finish
    bool isRboundary(); // Ken: haven't finish

    void show();
    void setcolor(int a);
    unsigned int gettype() {return type;}

    void up(int a = 1){};
    void down(int a = 1);
    void left(int a = 1);
    void right(int a = 1);

    void RotateR(int a = 1);
    void movetox(int a, int b);
    void movetobottom(Pool &in);
    void movetobottom2(Pool &in);

    friend class Pool;
};
```

```
class Blocks : public Move
{
    int x, y; // left one
    int color; // 40 ~ 47
public:
    Blocks(int xpos = 1, int ypos = 1, int col = 40): x(xpos), y(ypos), color(col) {}

    int getx() {return x;}
    int gety() {return y;}
    int getcolor() {return color;}
    void setxy(int xpos, int ypos);
    void setcolor(int a) { color = a; }
    void show() { cout << gotoxy(x, y) << "\x1b[" << color << "m \x1b[0m"; } // Ken: show

    void up(int a = 1) {y -= a;}
    void down(int a = 1) {y += a;}
    void left(int a = 1) {x -= a;}
    void right(int a = 1) {x += a;}

    void RotateR(int cx, int cy) {int lx = x, ly = y; x = cx - ly + cy; y = cy + lx - cx;}
    void RotateL(int cx, int cy) {} // Ken: can't understand ^

    friend class Shape;
};
```

```
class Move
{
public:
    virtual void up(int) = 0;
    virtual void down(int) = 0;
    virtual void left(int) = 0;
    virtual void right(int) = 0;
};
```

附錄 程式碼:

```
class Shape : public Cube, public Move
{
protected:
    unsigned int type;
    int NumberOfBlocks;
    Blocks *ptr;
public:
    Shape();
    Shape(unsigned int a, int b, int c, int d);
    Shape(const Shape &old);

    bool isbottom(Pool &in); // bool isLobstack
    bool isbottom2(Pool &in); // bool isRobstack
    bool isLboundary(); // Ken: haven't finish
    bool isRboundary(); // Ken: haven't finish

    void show();
    void setcolor(int a);
    unsigned int gettype() {return type;}

    void up(int a = 1){};
    void down(int a = 1);
    void left(int a = 1);
    void right(int a = 1);

    void Rotater(int a = 1);
    void movetoxy(int a, int b);
    void movetobottom(Pool &in);
    void movetobottom2(Pool &in);

    friend class Pool;
};
```

```
class Shape5 : public Shape // flag
{
    friend class Pool;
public:
    Shape5();
    Shape5(unsigned int t, int NOB, int x_pos, int y_pos);
    Shape5(const Shape5 &);
    static int fiveBlocksTimes;
};
```

附錄 程式碼:

```
void Rank::writefile()_{
    fstream rank_file;
    rank_file.open(rank_file_name.c_str(), ios::out | ios::app);
    if(!rank_file) cout << "failed";
    rank_file << user.name << " " << user.mode << " " << user.score;
    rank_file.close();
}

void Rank::readfile()
{
    fstream rank_file;
    rank_file.open(rank_file_name.c_str(), ios::in);
    if(!rank_file) cout<<"failed";

    string name; int m,s,h;
    while(rank_file >> name)
    {
        rank_file >> m >> s >> h;
        Point_vec.push_back(data(name, m, s, h));
        count++;
    }
}
```

```
class gotoxy
{
    int xx;
    int yy;
    friend ostream &operator<<(ostream &output, const gotoxy &a);
public:
    gotoxy(int x = 1, int y = 1):xx(x), yy(y){};
};

ostream &operator<<(ostream &output, const gotoxy &a){
    cout << "\x1b[" << a.yy << ";" << 2*a.xx << "H";
    return output;
}
```

附錄 程式碼:

```
class Rank
{
public:
    Rank(string file,int a = 0,data b = data("N",0,0,0))
    void readfile();
    void writefile();
    bool push_back_data(data a);
    void sorting_data();
    void show_rank(int mode);
    void getrank(data temp);
private:
    string rank_file_name;
    vector<data> Point_vec;
    data user;
    int count;
};
```