Group 11_貓咪眞可愛

b07609015 張萬宏　b07610008 黃茹暄　b08203004 趙軒磊　b08608062 鄭達玄　b08705048 連冠豪

遊戲設計

程式架構

重要函式

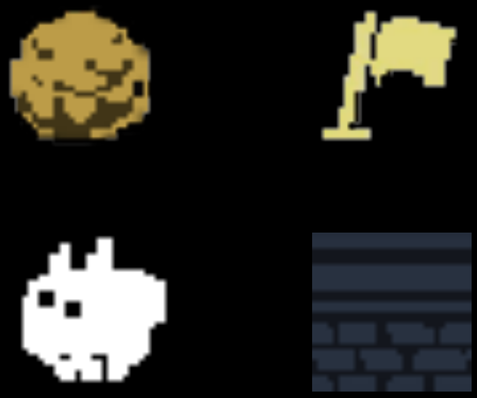
未來展望

遊戲設計

控制地圖上現有的文字方塊，透過改變他們的組合

就可以改變地圖的規則，最後利用 Object 通關

文字方塊



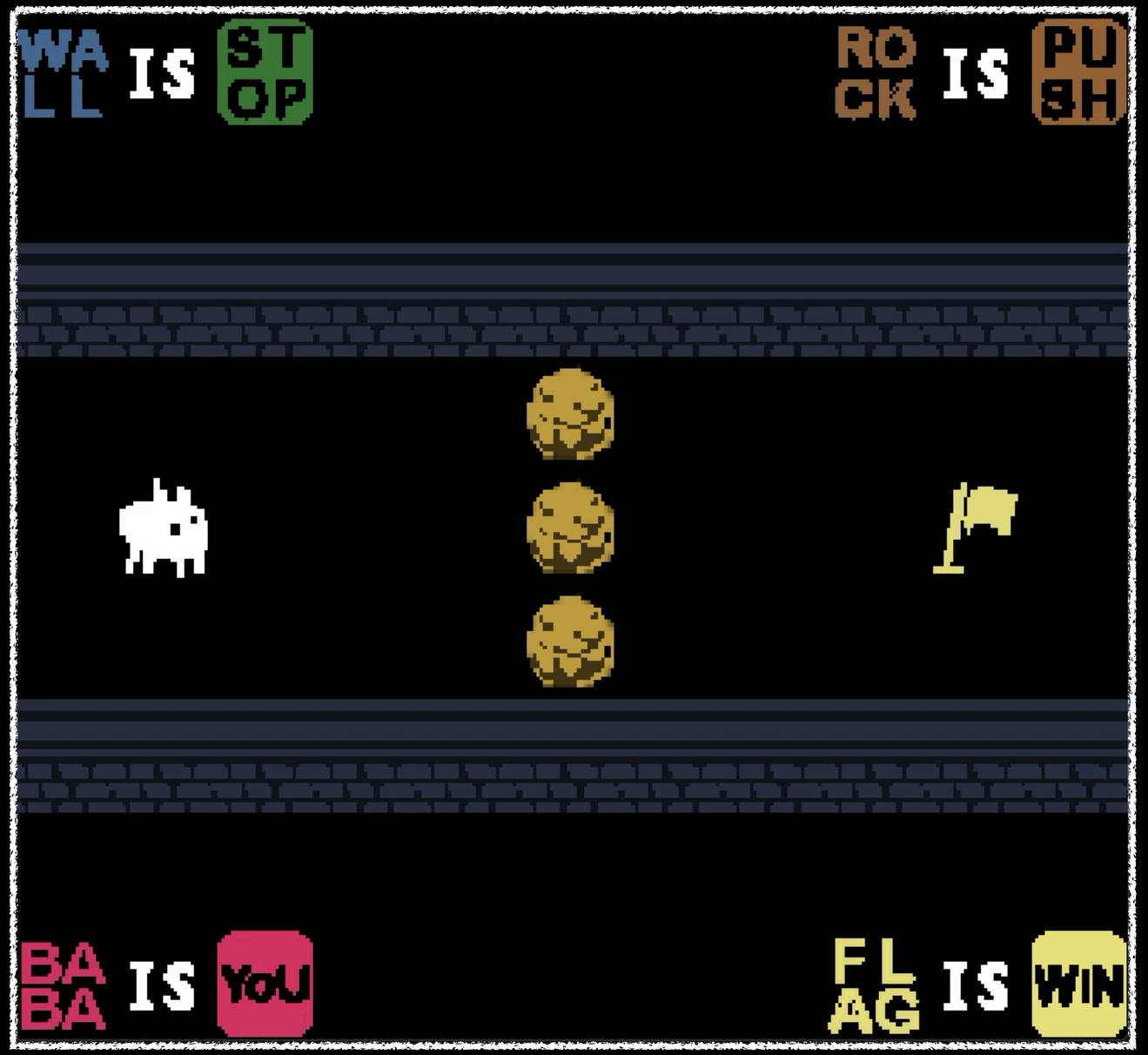
Object

遊戲設計

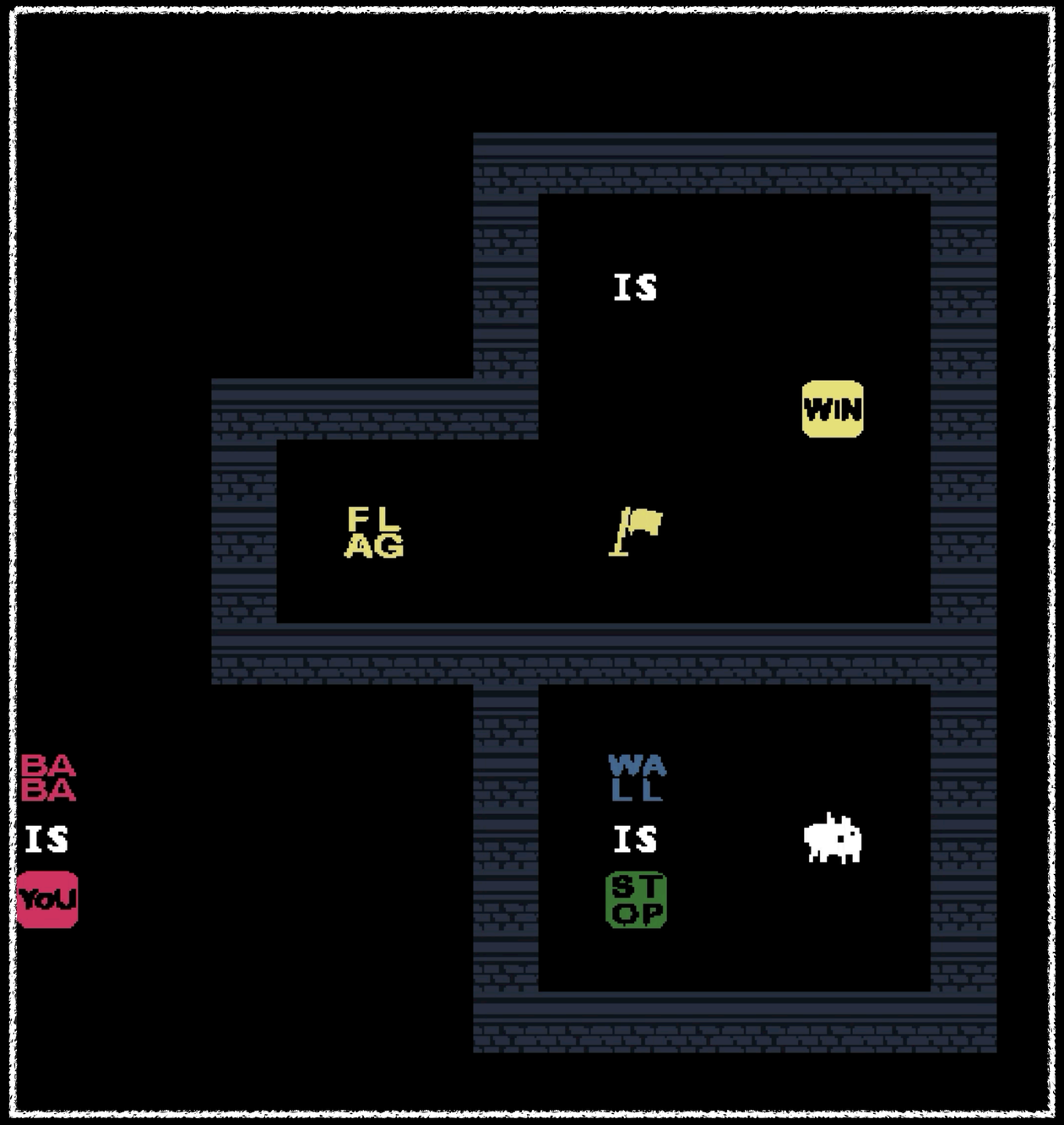
遊戲設計

遊戲設計

遊戲設計_Demo 1



遊戲設計_Demo 2



遊戲設計

程式架構

重要函式

未來展望

程式架構_Object


我們以一個 class 存取物品的資訊

物品名稱

物品目前的State（可被推動、無法推動、Win）

物品的位置

程式架構_You

我們以一個 class 存取 You

一個指向的 Object 的 pointer

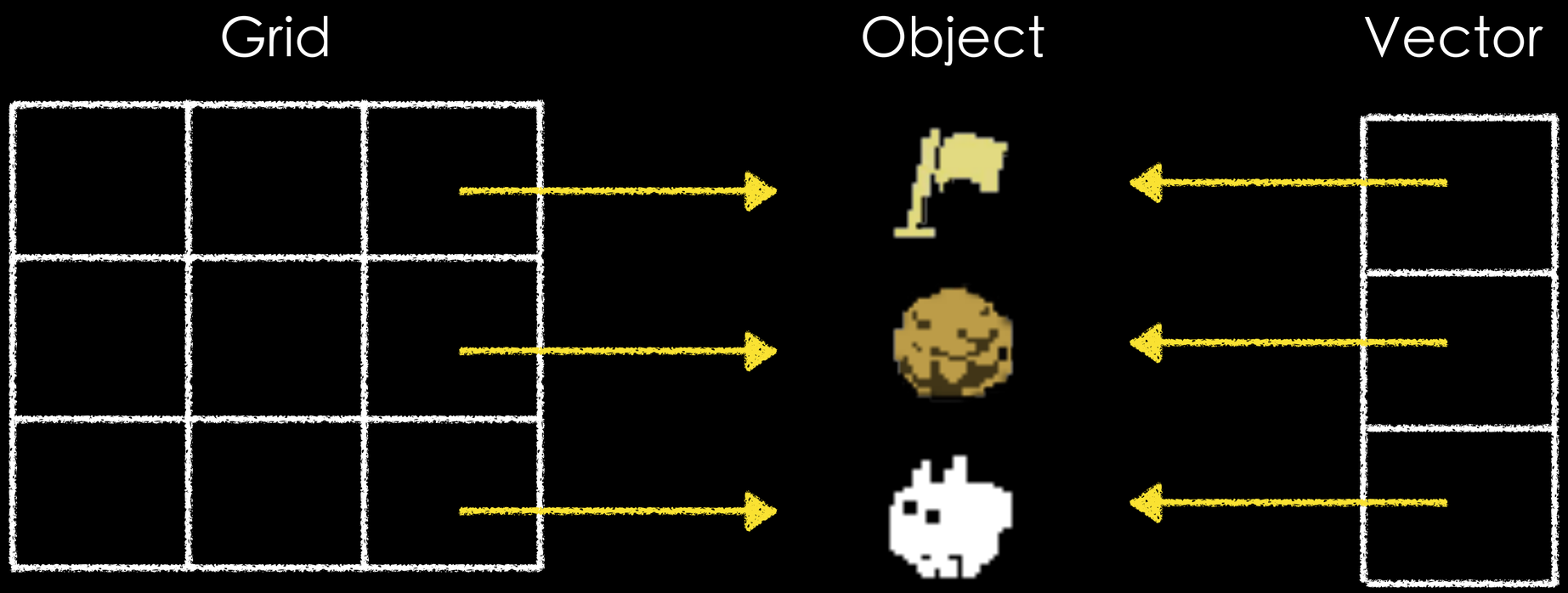
程式架構_地圖

我們以 pointer 為基礎

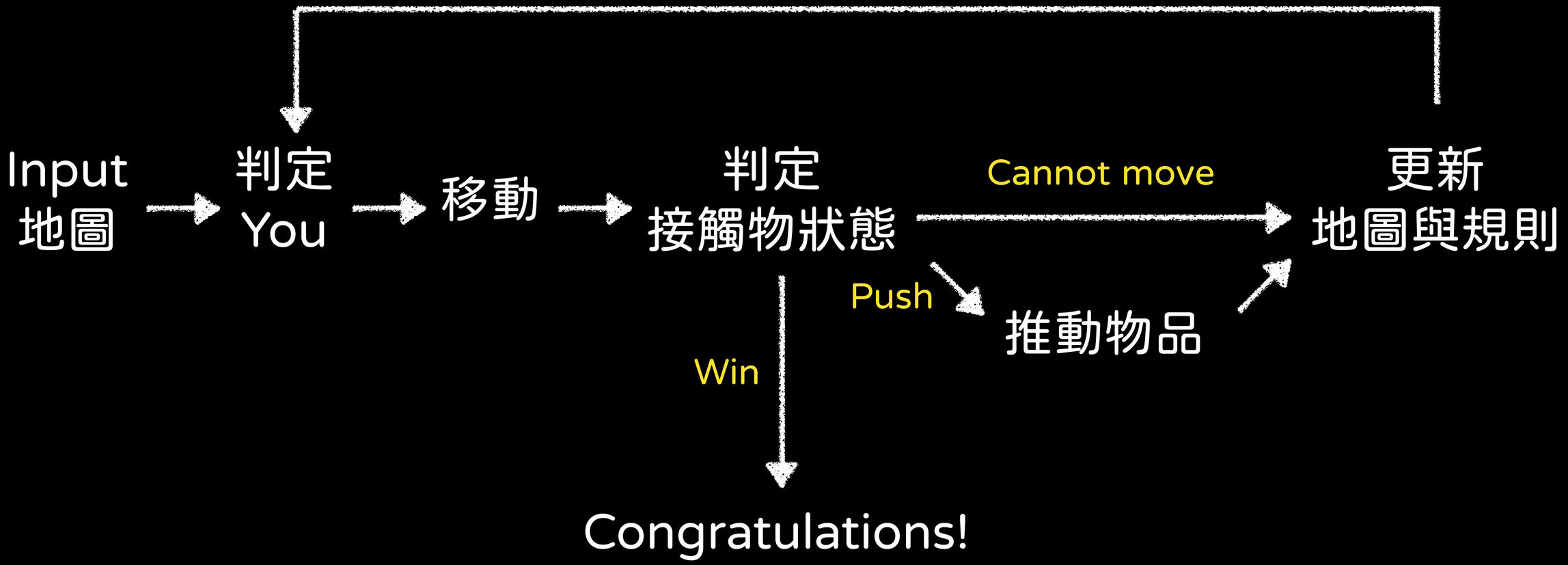
製成一個存取 *Object 的二維陣列(Grid)

當 Object 變換位置時，直接改變 pointer 的指向

此方式能夠大幅縮短更新地圖的速度



程式架構


Input
地圖
判定
You
移動
判定
接觸物狀態
Cannot move
更新
地圖與規則
Push
推動物品
Win
Congratulations!

遊戲設計

程式架構

重要函式

未來展望

Determine_State

```
int Object::Determine_State(){
    座標上是否 Empty?
      是 -> return 1
      否 -> Get_State檢視狀態
        1.Push
        2.Stop
        3.Win
        4.else
string Object::Get_State()
    vector WORDBLOCK_SUB 中，找對應的文字方塊
    取x, y值
    以grid判斷周圍是否有 is 跟 Adj
```

當物品要被推動時呼叫

將物品分成三種狀態

可被推動、無法被推動、勝利

物品的狀態由被推動的方向是否有另一物品而定

Determine_State

```
int Object::Determine_State(){
    座標上是否 Empty?
      是 -> return 1
      否 -> Get_State檢視狀態
        1.Push
        2.Stop
        3.Win
        4.else
string Object::Get_State()
    vector WORDBLOCK_SUB 中，找對應的文字方塊
    取x, y值
    以grid判斷周圍是否有 is 跟 Adj
```

當物品要被推動時呼叫

將物品分成三種狀態

可被推動、無法被推動、勝利

物品的狀態由被推動的方向是否有另一物品而定

Rock: BE STOPED

Determine_State

```
int Object::Determine_State(){
    座標上是否 Empty?
      是 -> return 1
      否 -> Get_State檢視狀態
        1.Push
        2.Stop
        3.Win
        4.else
string Object::Get_State()
    vector WORDBLOCK_SUB 中，找對應的文字方塊
    取x, y值
    以grid判斷周圍是否有 is 跟 Adj
```

當物品要被推動時呼叫

將物品分成三種狀態

可被推動、無法被推動、勝利

物品的狀態由被推動的方向是否有另一物品而定



Rock: BE PUSHED

Determine_State

```
int Object::Determine_State(){
    座標上是否 Empty?
      是 -> return 1
      否 -> Get_State檢視狀態
        1.Push
        2.Stop
        3.Win
        4.else
string Object::Get_State()
    vector WORDBLOCK_SUB 中，找對應的文字方塊
    取x, y值
    以grid判斷周圍是否有 is 跟 Adj
```

當物品要被推動時呼叫

將物品分成三種狀態

可被推動、無法被推動、勝利

物品的狀態由被推動的方向是否有另一物品而定

MOVEABLE

```
function (包含W, A, S, D) (旨在確定移動總次數)
  if(接下來將抵達的位置為空)
     return 1
  if(接下來將抵達的物件不可被移動)
     return 一極小值
  else if(接下來將抵達的物件可被移動)
     return moveable(下一物件指標) + 1
     //代表總移動次數
```

當物品要被推動時呼叫

先偵測此物品前方是否有物品

如有，則看前方物品是否能被推動

如沒有，則移動

MOVEABLE

```
function (包含W, A, S, D) (旨在確定移動總次數)
  if(接下來將抵達的位置為空)
     return 1
  if(接下來將抵達的物件不可被移動)
     return 一極小值
  else if(接下來將抵達的物件可被移動)
     return moveable(下一物件指標) + 1
     //代表總移動次數
```

當物品要被推動時呼叫

先偵測此物品前方是否有物品

如有，則看前方物品是否能被推動

如沒有，則移動

Rock2: BE STOPED

MOVEABLE

```
function (包含W, A, S, D) (旨在確定移動總次數)
  if(接下來將抵達的位置為空)
     return 1
  if(接下來將抵達的物件不可被移動)
     return 一極小值
  else if(接下來將抵達的物件可被移動)
     return moveable(下一物件指標) + 1
     //代表總移動次數
```

當物品要被推動時呼叫

先偵測此物品前方是否有物品

如有，則看前方物品是否能被推動

如沒有，則移動

Rock2: BE PUSHED

move

```
move function (包含up, down, left, right) {
  if(接下來將抵達的位置不合法)
     不動作
  if(接下來將抵達的位置為win)
     宣告勝利
  if(接下來將推動的物件終將停止)
     不動作
  if(接下來可合法推動物件)
     將第n格和最後一個互換指標及位置   n = 1 ~
    總共推動方塊數+1
     將YOU指定為下一個位置的物件指標
     更改朝向
}
```

若物品確定可被移動時呼叫

將要移動的物品更新地圖上的位置

遊戲設計

程式架構

重要函式

未來展望

未來展望

- 做出一個演算法，以自動生成可破關的新地圖
- 做出倒退按鍵，以利玩家走錯時能夠回到上一步，不用關掉程式
- 將地圖上的 Object 改成 gif 檔，增加遊戲生動的感覺
- 新增不同 Object 互動時的音效
- 解決 Baba 和 Wall 互動時的呈現