

理財「遊」學

-運用 Unity 製作一個理財學習遊戲

研究者：游旭紳
指導老師：許瓊云

研究摘要

本研究旨在探討使用 Unity 遊戲引擎製作一個理財學習的遊戲，透過文獻閱讀理解與分析，整理出對國小學生重要的理財相關內容，同時參考書籍資料以及網路資料進行 Unity 程式設計的自學，並運用 Visual Studio 等程式與繪圖軟體製作遊戲圖像，完成遊戲的編寫與製作，最後，讓國小六年級的學生進行遊戲試玩，並填寫回饋問卷來審視試玩者對遊戲的滿意度及建議。

第一章 緒論

一、研究動機

我從國小三年級開始在家裡看書自學程式，一開始是運用 scratch 來製作遊戲，從四年級開始媽媽買了一本關於 python 的書，所以就開始繼續學程式語言，在過程中我發現我很喜歡寫程式，因為可以自由地做出東西，有種想做什麼都做得出來的感覺，並且能鍛鍊邏輯思考的能力，這次的獨立研究，從要訂定主題的時候，第一個冒出在腦袋的想法就是想要透過寫程式來做一個遊戲。

我玩過很多種的電動遊戲，也玩過教育用途的遊戲(例：Pagamo)，所以就思考可以有哪樣的主題作為我自製遊戲的主題及內容，最後整理出「理財」的這個在學校較少學習到的項目，將理財相關概念融入進我的遊戲中，並且在遊戲完成後，邀請同學進行試玩與回饋，從想法心得與建議中獲取可以更進步的目標。

二、研究目的

1. 了解並分析理財教育的相關內容及重要性
2. 運用 Unity 製作一個理財學習的遊戲

三、研究問題

1. 理財素養是什麼？
2. 現有的理財素養課程有哪些？
3. 現有的教育用途遊戲有哪些？
4. Unity 遊戲引擎是什麼？
5. 如何運用 Unity 進行理財素養遊戲程式設計？
6. 透過理財遊戲培養國小六年級生理財素養的成效如何？

四、名詞解釋

1. 遊戲引擎：

用於製作電腦遊戲的軟體(例如：Unity ,Unreal engine)，使用遊戲引擎可以加快製作遊戲的速度，降低遊戲設計的門檻，相較於不用遊戲引擎，可以節省非常多步驟，而且很多遊戲引擎都可以在網路上找到資源。

第二章 文獻探討

一、理財素養

在《年輕世代金融素養拉警報！金融素養教育即將大改革？》這篇文章中提到，隨著新興科技發展快速，商品的種類越來越多元，因此民眾的金融素養越來越重要，金融消費評議中心暨消基會董事長林盟翔曾經表示：「金融素養應從小開始建立」，金融素養的重要性可以由以下幾個面向來看：

- 償還和避免債務-了解背負債務的風險以及如何避免。
- 建立結構化預算-可以預先設想好未來的儲蓄和支出計畫，避免陷入財務困境。
- 策略性地分析資產-了解有關投資的技術知識。
- 計畫有保障的退休生活-進行目標投資或儲蓄規劃。
- 為緊急情況做好準備-準備緊急預備金，可以在意外事件發生時派上用場。
- 降低金融詐騙或不當交易的犯罪率-擁有金融素養的人較不容易被金融詐騙。

二、現有的理財素養教育案例

1. 日本

日本政府設立了「金融服務資訊中央委員會」，並在 2015 年制定「金融素養」地圖，根據不同年齡的學生設定不同目標，並規劃了八個學習主題，分別為以下：

- 了解商品與金錢的價值
- 了解商品與金錢有限以及金錢的重要性
- 釐清自己的需要與想要
- 了解金錢的來源以及賺錢這件事
- 建立預算與記帳的觀念
- 認識銀行以及養成儲蓄習慣
- 認識國內外的貨幣
- 認識保險與風險

2. 新加坡

新加坡金融管理局在 2003 年創立國家級教育計劃-「money sense」網站，提供相對應文章給國民可以閱讀學習，共分為七個主題：儲蓄、貸款和信貸、保險、投資、財產、退休、遺產規劃。

3. 臺灣

臺北市政府教育局於民國 110 年召集臺北市國教輔導團、專家、學者及教師組成研發團隊，編撰國小、國中到高中的財務智商試題，並且上架《酷課雲》，提供學生造訪並進行答題；111 學年度編撰財經素養探究任務教材，並透過時事文章，設計財經素養相關題目；臺北市政府教育局網站設有臺北市理財教育專區。

新北市立三重國小連秀霞老師設計了社會理財課程的大富翁遊戲，並且提供線上教材放置於「教育大市集」中，讓有需要的教師皆能夠參考與使用，課程遊戲中透過「收入」、「支出」及「機會/命運」牌來進行金錢的流動與使用。

臺灣金融監督管理委員會保險局為了建立人民的風險意識，設立了風險管理與保險教育推廣入口網，提供有需求的中小學師生能參考並執行，其中特別設有金融基礎教育專區，國小的金融教育架構有三大主題：金錢規劃、借貸與信用、風險與風險管理。

三、現有的教育用途遊戲

1. 期刊-《嚴肅遊戲之角色扮演與情境模擬對於學習成效之影響：以國小五年級碳足跡課程為例》

使用角色扮演遊戲及多媒體動畫，透過貼近生活的各種模擬關卡來進行遊戲化教學，用小考的方式來評估學生對碳足跡主題的了解程度，觀察遊戲化教學對學生對碳足跡的了解是否有影響。

2. 文章-《嚴肅遊戲不准笑？內容遊戲化的可能性》

提到了嚴肅遊戲的種類有兩種，分別是結構式遊戲化和內容式遊戲化，兩種各有不同的特點：

- 結構式遊戲化-成本較低，但遊戲的趣味性和接受度較低
- 內容式遊戲化-開發成本高，但是遊戲裡會加入更多遊戲的要素，接受度會比結構式遊戲化的接受度更高。

3. 文章-《理財媽媽實踐 STEAM 教學,把理財變得好好玩》

介紹了如何透過特殊的回饋機制來教導兒童儲蓄及投資，將理財知識融入我們的生活之中，有趣又好玩，例如：

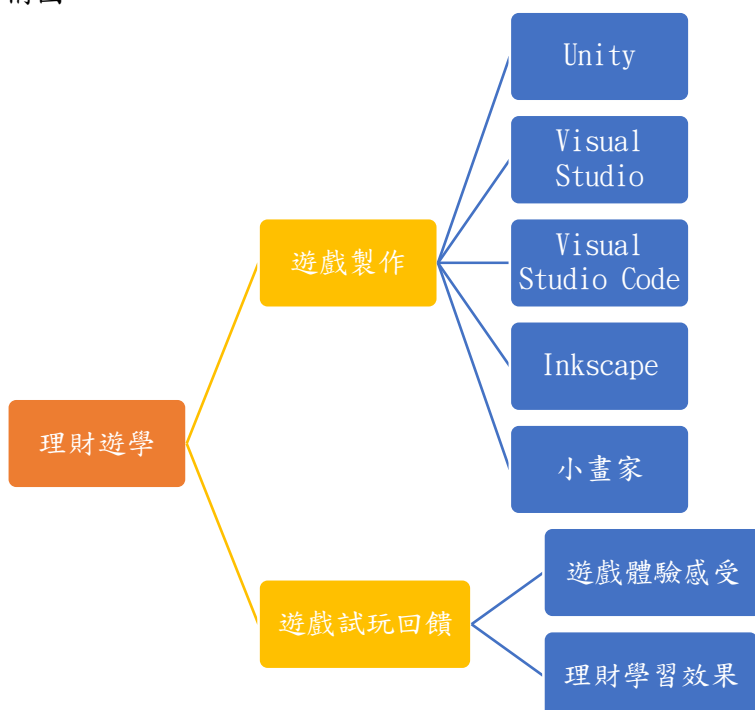
- 每個月收到零用錢時，將這筆錢分成幾個部分，包括「要花的錢」（消費）、「要存的錢」（儲蓄）、「要捐的錢」（幫助別人）。
- 365 存錢法，一年 365 天都要存錢，存錢金額為每天要增加 1 元，第一天 1 元、第二天 2 元、第三天 3 元、第四天 4 元，到第 365 天當天一日要存 365 元。

四、小結

綜合以上文獻閱讀與整理，我擷取三項理財素養重點作為我遊戲學習的主要目標，分別為「分配收入」、「儲蓄理財」、「消費習慣」，過程中加入利息或較低風險的投資機制，讓玩家學習儲蓄理財和投資的風險、技巧；並且透過角色扮演類型的遊戲，將知識內容遊戲化，讓學生/玩家能夠更有意願進行學習與參與。

第三章 研究方法

一、研究架構圖



二、研究方法

1. 遊戲劇情與關卡設計

我從文獻當中整理出我所要製作的遊戲學習目標為：「分配收入」、「儲蓄理財」、「消費習慣」、「投資風險」，故我從此目標進行發想，並且透過第一人稱的小明作為遊戲主角，設定故事並將後續的理財挑戰串聯，讓玩家在遊戲中一年的時間進行各項事件的選擇與應對，並在遊戲最後檢視是否達成理財目標。

2. 程式編寫

我透過 Unity 進行遊戲場景以及物件行為的製作，並且使用 Visual Studio Code 和 Visual Studio 進程式碼的編寫，並利用小畫家和 Inkscape 進行遊戲圖像的製作。

3. 問卷回饋調查

使用 google 表單進行遊戲回饋的問卷製作，透過量化數據以及遊戲體驗過程的經驗回饋進行綜合的整理。

問卷問題：

- (1) 請問你在遊戲結束時的結果是下列哪一個？
- (2) 你覺得在遊戲的操作過程中，順暢度如何？分數 1-5，5 分為非常順暢，1 分為非常不順暢。
- (3) 你覺得遊戲操作的直覺程度如何？分數 1-5，5 分為非常直覺，1 分為非常不直覺。
- (4) 你覺得這個遊戲的關卡難度如何？分數 1-5，5 分為非常難，1 分為非常簡單。
- (5) 你覺得這個遊戲能更幫助你知道如何管理金錢（理財）嗎？分數 1-5，5 分為非常有幫助，1 分為沒有幫助。
- (6) 你覺得透過這個遊戲，能夠讓你對於「理財」這件事更有興趣嗎？分數 1-5，5 分為非常能提升興趣，1 分為無法提升興趣。
- (7) 你學會了哪些理財的技巧？
- (8) 你在遊戲過程中所遇到最印象深刻的事件為何？
- (9) 承上題，為什麼這個事件讓你印象深刻呢？
- (10) 其他想法與回饋

回饋問卷連結：

<https://forms.gle/CermUXPREem2fMCK6>

三、研究工具

1. Unity engine

Unity engine 是 Unity 公司所開發的遊戲引擎，它有提供免費方案，所以使用 Unity 不需要付錢，但是使用免費方案做的遊戲收入每月不得超過二十萬美元。

2. Visual Studio

Visual Studio 是 Microsoft 公司所開發的 IDE(整合開發環境)，Visual Studio 的社群版提供開發者免費下載，有除錯、自動補全等功能，適合用來開發 C# 或 C++。

3. Visual Studio Code

Visual Studio Code 是 Microsoft 公司開發的程式碼編輯器，是免費的輕量編輯器，有許多的擴充功能，支援各種語言。

4. Inkscape

Inkscape 是一個免費的向量圖編輯器，可以自由用在個人或商業用途。

5. 小畫家

小畫家是 Microsoft 公司開發的點陣圖繪圖軟體，可以免費使用。

第四章 研究結果

一、遊戲劇情與關卡設計

1. 遊戲故事背景

有一個大學剛畢業不久的年輕人小明因為家境貧困，爸爸因為公司倒閉而負債 120000 元，爸爸和媽媽每天都努力工作賺錢還債，生活過得很清寒，所以小明決定要離家過獨立自主的生活，小名找到了一份薪水穩定的打工之後，決定要開始過獨立的生活，利用打工得到的金錢維持生活，並盡快找到適合小明的的工作，獨自度過新生活的第一年。

2. 遊戲說明

進入遊戲後點擊“開始這一週”按鈕後會隨機跳出事件，可以在選擇類型事件進行選擇，遊戲中有三個數字，金錢、開心度和週數，遊戲一共進行 52 週，每個事件都會影響金錢和開心度，只要在最後的第 52 週賺到 120000 元且開心度大於 80 就算過關。

3. 遊戲連結：<https://asher2013.github.io/>



遊戲連結 QRcode：

(建議使用電腦網頁開啟，版面較完整)

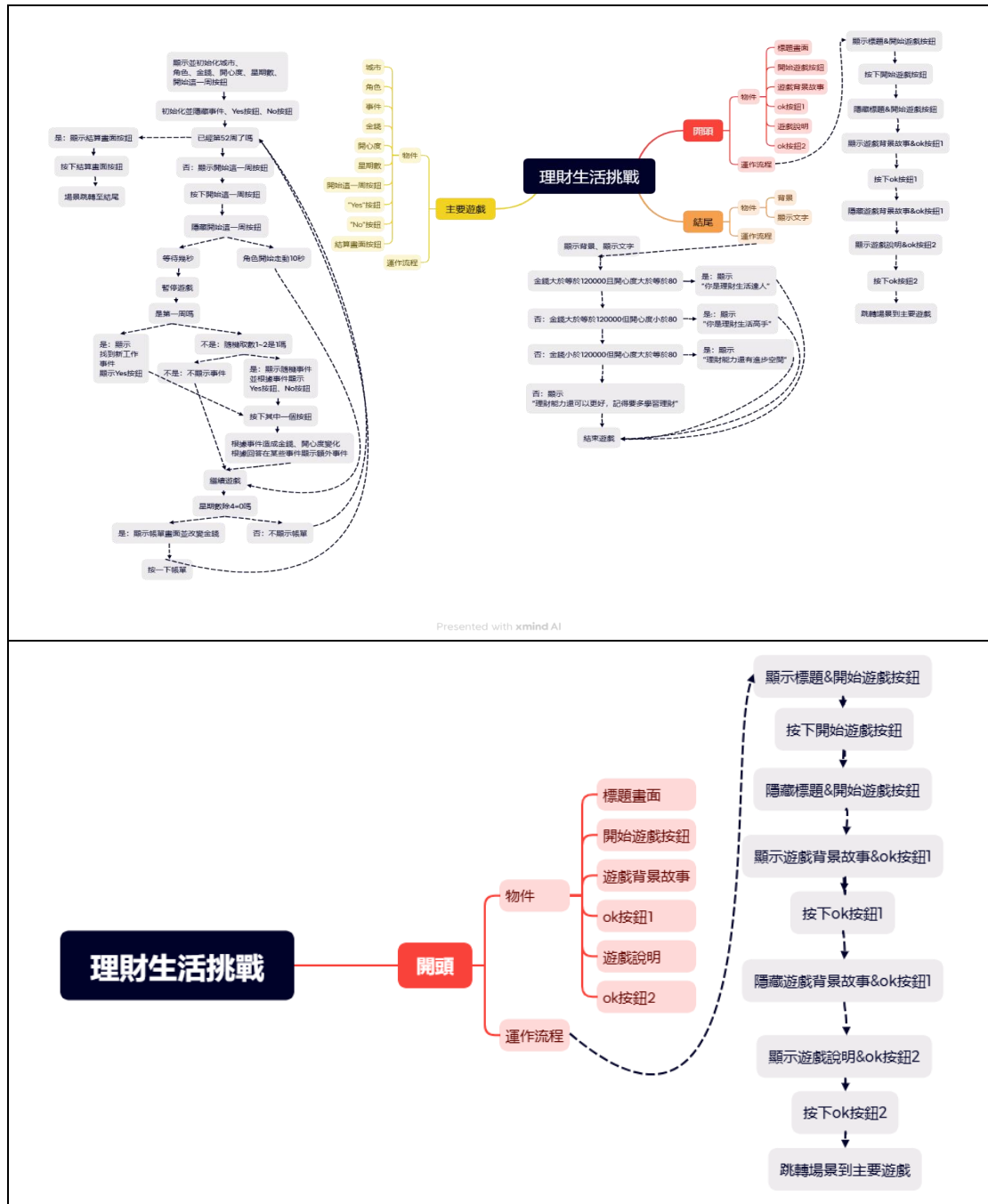
4. 理財遊戲過程事件

項目	金額	開心度	項目	金額	開心度
1. 獲得工作	每月+35000 元	+0	2. 弄丟悠遊卡	-200 元	-3
3. 參加抽獎	+500 元	+5	4. 桌球比賽獲勝	+200 元	+4
5. 朋友邀請投資	選擇	選擇	6. 新鞋購物	選擇	選擇
7. 弄丟鈔票	-100 元	-2	8. 朋友資助	+3000 元	+4
9. 參加抽獎活動	-300 元	-2	10. 幫朋友打掃房間	+1000 元	+0
11. 找到遺失鈔票	+2000 元	+3	12. 線上遊戲課金	-200 元	+3
13. 朋友請客	+300 元	+2	14. 夜市遊玩	-500 元	+5
15. 跳蚤市場擺攤	+2000 元	+3	16. 購買手機	選擇	選擇
17. 幫鄰居遛狗	+1000 元	+0	18. 賣手工餅乾賺錢	+3000 元	+2
19. 投資詐騙	-20000 元	-10	20. 投資簡訊	選擇	選擇
21. 限定公仔	選擇	選擇	22. 衛生紙不夠	選擇	選擇
23. 獲得利息	金錢乘以 0.7%	+0	24. 去動物園	選擇	選擇
25. 去看電影	-1000 元	+5	26. 開餅乾店	選擇	選擇
27. 想要捐款	選擇	選擇	28. 想買陶壺	選擇	選擇
29. 料理比賽	選擇	選擇	30. 買彩券	選擇	選擇

5. 理財遊戲過程額外事件

項目	金額	開心度	項目	金額	開心度
5-1 成功	-20000 元 +30000 元	+6	5-2 朋友邀請 投資_失敗	-20000 元 -20000 元	-10
6-1 買了新鞋	-3000 元	+6	6-2 沒買新鞋	+0 元	+0
16-1 買了手機	-6000 元	+10	16-2 沒買手機	+0 元	+0
20-1 投資簡訊_詐騙	-10000 元 -10000 元	-10	21-1 買公仔	-1000	+5
21-2 沒買公仔	+0 元	-3	22-1 上廁所衛生紙不夠	+0	-8
26-1 餅乾店獲利	-30000 元 +50000 元	+5	26-2 餅乾店虧損	-30000 元 -10000 元	-10
27-1 捐款	-200 元	+6	27-2 沒有捐款	+0 元	-2
28-1 賣出陶壺	-5000 元 +100000 元	+10	28-2 廉價陶壺	+0 元	-10
29-1 比賽失利	-1000 元	-8	30-1 彩券未中獎	-1000 元	-5
30-2 彩券中獎	-1000 元 +10000 元	+5			

6. 遊戲架構心智圖

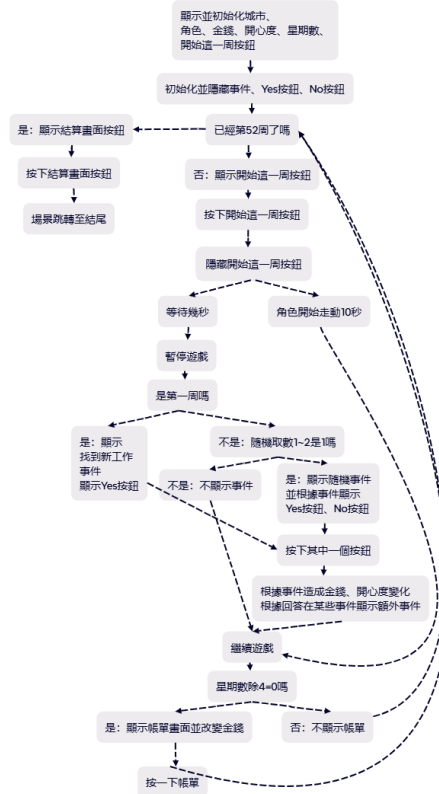


Presented with xmind AI

理財生活挑戰

主要遊戲

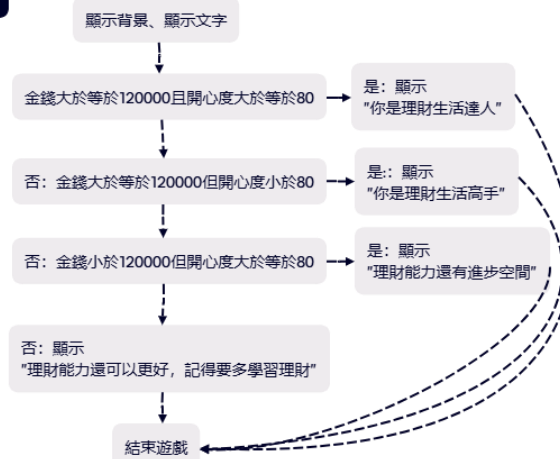
- 城市
- 角色
- 事件
- 金錢
- 開心度
- 星期數
- 開始這一周按鈕
- "Yes"按鈕
- "No"按鈕
- 結算畫面按鈕
- 運作流程



理財生活挑戰

結尾

- 物件
 - 背景
 - 顯示文字
- 運作流程



7. 遊戲程式編寫

- 製作過程發生的困難問題

有問題的程式碼	解決後的程式碼
<pre> void Start() { EventObjet.GetComponent<Image>().enabled = false; StartCoroutine(EventCycle()); } IEnumerator EventCycle() { while (true) { if (IsClicking.week <= 52) { Debug.Log(\$"[CYCLE CHECK] isclick: {IsClicking.isClick}, isRunning: {IsRunning}"); if (IsClicking.isClick && IsRunning) { IsRunning = true; IsClicking.isClick = false; yield return StartCoroutine(Main()); IsRunning = false; } yield return null; } else { SceneManager.LoadScene("結尾"); } } } </pre>	EndBtn.cs <pre> public class EndBtn : MonoBehaviour { // Start is called before the first frame update public GameObject Week; void Start() { this.gameObject.GetComponent<UnityEngine.UI.Image>().enabled = false; this.gameObject.GetComponent<UnityEngine.UI.Button>().enabled = false; this.gameObject.GetComponentInChildren<TMPPro.TMP_Text>().enabled = false; } // Update is called once per frame void Update() { if (Week.GetComponent<IsClicking>().week == 52) { this.gameObject.GetComponent<UnityEngine.UI.Image>().enabled = true; this.gameObject.GetComponent<UnityEngine.UI.Button>().enabled = true; this.gameObject.GetComponentInChildren<TMPPro.TMP_Text>().enabled = true; } } public void OnClick() { SceneManager.LoadScene("結尾"); } } </pre>
EventController.cs	
說明： 左圖是原本在第 52 周讓場景跳轉到結尾的程式，只要一到第 52 周就會將場景跳轉到結尾，但是這樣的做法會導致遊戲當機，所以我將功能改成在第 52 周時顯示一個”進入結算畫面”按鈕，並使用 EndBtn.cs 控制按鈕，讓”進入結算畫面”按鈕被按下時將場景跳轉到結尾，這個做法解決了遊戲當機的問題。	

有問題的程式碼	解決後的程式碼
<pre> public void OnClick() { IsClicking.clickedYes = true; IsClicking.isClick = true; switch (eventIndex) { case 0: StartCoroutine(Waitstart()); IsClicking.money += 35000; IsClicking.hasWork = true; YesBtnController.startEvent = false; break; case 1: StartCoroutine(Waitstart()); IsClicking.money -= 200; IsClicking.happiness -= 3; YesBtnController.startEvent = false; break; case 2: StartCoroutine(Waitstart()); IsClicking.money += 500; IsClicking.happiness += 5; YesBtnController.startEvent = false; break; case 3: StartCoroutine(Waitstart()); IsClicking.money += 200; } } </pre>	<pre> public void OnClick() { IsClicking.clickedYes = true; IsClicking.isClick = true; switch (eventIndex) { case 0: StartCoroutine(Waitstart()); IsClicking.money += 35000; IsClicking.hasWork = true; YesBtnController.startEvent = false; break; case 1: StartCoroutine(Waitstart()); IsClicking.money -= 200; IsClicking.happiness -= 3; YesBtnController.startEvent = false; break; case 2: StartCoroutine(Waitstart()); IsClicking.money += 500; IsClicking.happiness += 5; YesBtnController.startEvent = false; break; case 3: StartCoroutine(Waitstart()); IsClicking.money += 200; IsClicking.happiness += 200; new WaitForSeconds(0.5f); if (UnityEngine.Random.Range(0, 2) == 0) { // ... } } } </pre>
說明： 原本將執行事件的程式碼直接寫在按鈕的程式碼中，但是會造成兩個按鈕的事件同時執行，造成異常增加或扣除金錢或開心度的情況，所以將執行事件的程式碼放到迴圈程式裡，利用條件判斷避免事件同時執行，讓改變金錢和開心度的程式碼正常運作。	

二、程式編寫說明

1. 主系統程式

玩家 PlayerController.cs

```
1  using System.Collections;
2  using UnityEngine;
3  using UnityEngine.UI;
4
5  public class PlayerController : MonoBehaviour
6  {
7      public GameObject PlayerObject;
8      public float speed = 3f;
9
10     public Animator animator;
11     public int dining;
12     public int live;
13     public int house;
14     public GameObject BillObject;
15
16
17     // Update is called once per frame
18     void Update()
19     {
20         if (animator.GetBool("walk") == true)
21         {
22             speed = 3f;
23             PlayerObject.transform.position += new Vector3(0, 0, speed * Time.deltaTime);
24         }
25     }
26
27
28     public void Onclick()
29     {
30         StartCoroutine(WalkCoroutine());
31     }
32
33
34     IEnumerator Bill()
35     {
36         if (IsClicking.week % 4 == 0)
37         {
38             dining = Random.Range(9000, 11000);
39             live = Random.Range(5000, 6000);
40             house = 10000;
41
42             BillObject.SetActive(true);
43             BillObject.GetComponent<Image>().enabled = true;
44             BillObject.GetComponentInChildren<TMPPro.TMP_Text>().enabled = true;
45
46             BillObject.GetComponentInChildren<TMPPro.TMP_Text>().text = $"餐飲費花了{dining}元\n生活費花了{live}元\n房租費花了{house}元";
47             if (IsClicking.week % 4 == 0 && EventController.haveWork)
48             {
49                 IsClicking.money += 35000;
50             }
51             yield return new WaitForSeconds(10f);
52             IsClicking.money -= house + dining + live;
53             BillObject.SetActive(false);
54         }
55     }
56     private IEnumerator WalkCoroutine()
57     {
58         animator.SetBool("walk", true);
59         IsClicking.isClick = true;
60         IsClicking.isShowed = false;
61         yield return new WaitForSeconds(10f);
62         IsClicking.isClick = false;
63         IsClicking.isShowed = true;
64         animator.SetBool("walk", false);
65         yield return StartCoroutine(Bill());
66         IsClicking.week += 1;
67     }
68 }
```

說明：

1-3 列引入 UnityEngine 跟相關函式庫。

7-14 列 宣告各式物件跟數值。

18 列 Update()，每幀都會被呼叫一次，如果變數 Walk 的值是 true，讓角色移動。

29 列 Onclick()，當被點擊時，開始執行 WalkCoroutine()。

35 列 Bill(), 如果當前是「第 4 的倍數週」(即每個月結束), 每月的餐飲費、生活費是隨機產生一個範圍, 房租是固定 10000, 打開帳單畫面(裡面的文字也顯示), 如果是每個月週(4 的倍數)並且玩家有工作的話, 領到 35000 薪水, 等待玩家按下滑鼠左鍵(Mouse0), 扣掉繳房租、餐飲費、生活費, 把帳單 UI 關閉。
56 列 WalkCoroutine(), 是玩家點擊後走一段路的流程, 行走結束後, 呼叫 Bill(), 檢查要不要繳費, 每走一次+1 週數。

鏡頭 CameraController.cs

```
1 using UnityEngine;
2
3 public class CameraController : MonoBehaviour
4 {
5
6     public GameObject Camera;
7     public GameObject Player;
8     public float distance;
9     public float StartDistance;
10
11
12
13     void Start()
14     {
15         StartDistance = Camera.transform.position.z - Player.transform.position.z;
16     }
17
18
19
20     void Update()
21     {
22         distance = Camera.transform.position.z - Player.transform.position.z;
23         Camera.transform.Translate(0, 0, StartDistance - distance);
24     }
```

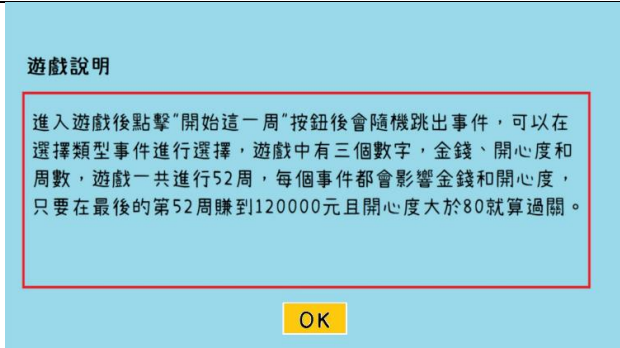
說明：
依照 Player 跟隨。

城市 CityController

```
1 using UnityEngine;
2
3 public class CityController : MonoBehaviour
4 {
5     // Start is called before the first frame update
6     public GameObject CityObject;
7     public GameObject Player;
8     void Start()
9     {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16         float distance = Player.transform.position.z - CityObject.transform.position.z;
17         float size = 476f;
18         Debug.Log(size);
19         if (distance > size)
20         {
21             CityObject.transform.position = new Vector3(CityObject.transform.position.x,
22             CityObject.transform.position.y, CityObject.transform.position.z + size * 2);
23         }
24     }
```

說明：
玩家走完一個 size (476f) 時將城市物件前移。

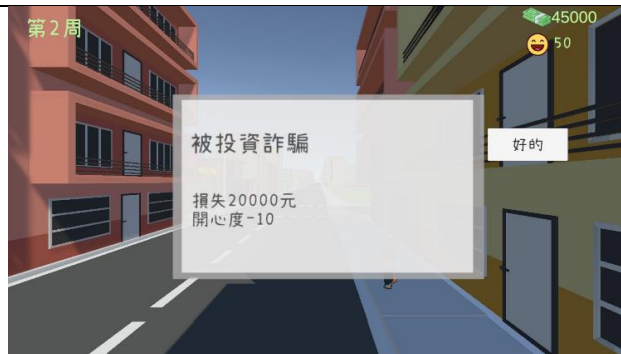
2. 主遊戲流程

開始畫面 	程式碼 StartButtonController.cs <pre> 1 using UnityEngine; 2 3 public class StartButtonController : MonoBehaviour 4 { 5 // Start is called before the first frame update 6 7 public GameObject Title; 8 public GameObject Btn; 9 public void Start() 10 { 11 12 } 13 void Click() 14 { 15 Destroy(Title); 16 Destroy(Btn); 17 } </pre>
說明： 按下 Start 後，刪除 Title/Btn	
故事背景&遊戲說明 	程式碼 OkButton2Controller1.cs <pre> 1 using UnityEngine; 2 using UnityEngine.SceneManagement; 3 4 public class OkButtonController2 : MonoBehaviour 5 { 6 // Start is called before the first frame update 7 public GameObject Description; 8 public GameObject OkBtnTwo; 9 void Start() 10 { 11 12 } 13 public void OnClick() 14 { 15 Destroy>Description; 16 Destroy(OkBtnTwo); 17 SceneManager.LoadScene("Game"); 18 } </pre>
說明： 按下 OK 後，刪除 OK 轉場，切換至 Game 場景	
遊戲開始 	月花費畫面 

突發事件畫面



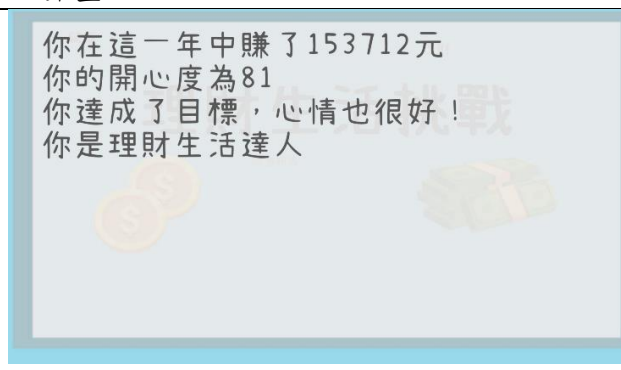
突發事件畫面



結算前畫面



結算畫面



程式碼 EventController.cs

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4  using UnityEngine.UI;
5  using UnityEngine.Mathematics;
6
7
8
9  0 個參考
10 public class EventController : MonoBehaviour
11 {
12     4 個參考
13     public Sprite[] eventImages;
14     7 個參考
15     public GameObject EventObject;
16     0 個參考
17     public GameObject BillObject;
18     0 個參考
19     public GameObject YesBtn;
20     0 個參考
21     public GameObject NoBtn;
22     8 個參考
23     public static int eventIndex = -1;
24     2 個參考
25     public static bool haveWork = false;
26     5 個參考
27     private Image uiImage;
28     10 個參考
29     public static bool isRunning = false;
30     3 個參考
31     private bool isExtraEventProcessed = false;
32     0 個參考
33     public GameObject WeekObject;
34
35
36
37     1 個參考
38     public Sprite[] eventImagesSpecial;
39
40     0 個參考
41     void Awake()
42     {
43         uiImage = GetComponent<Image>();
44     }
45
46     0 個參考
47     void Start()
48     {
49         EventObject.GetComponent<Image>().enabled = false;
50         StartCoroutine(EventCycle());
51     }
52
53     60 個參考
54     private IEnumerator Waitstart()
55     {
56         yield return new WaitUntil(() => YesBtnController.startEvent);
57     }
58
59     1 個參考
60     IEnumerator EventCycle()
61     {
```



```

45     while (true)
46     {
47
48         Debug.Log($"[CYCLE CHECK] isClick: {IsClicking.isClick}, isRunning: {isRunning}");
49         if (IsClicking.isClick && !isRunning)
50         {
51             isRunning = true;
52             IsClicking.isClick = false;
53             yield return StartCoroutine(Main());
54
55             isRunning = false;
56         }
57         yield return null;
58     }
59
60
61 }
62
63
64
65

```

1 個參考

```

66 IEnumerator Main()
67 {
68     Debug.Log($"[MAIN START] isClick: {IsClicking.isClick}, isRunning: {isRunning}");
69     isExtraEventProcessed = false;
70     Time.timeScale = 1f;
71     yield return new WaitForSeconds(UnityEngine.Random.Range(0f, 7f));
72
73     List<int> validIndexes = new List<int>();
74     for (int i = 0; i < eventImages.Length; i++)
75     {
76         if (eventImages[i] != null)
77         {
78             validIndexes.Add(i);
79         }
80     }
81
82     if (validIndexes.Count == 0)
83     {
84         Debug.Log("[MAIN] No available event images.");
85         isRunning = false;
86         yield break;
87     }
88
89     if (!haveWork)
90     {
91         eventIndex = 0;
92     }
93     else
94     {
95         if (UnityEngine.Random.Range(0, 2) != 1)
96         {
97             Debug.Log("[MAIN] Random chose not to show event.");
98             isRunning = false;
99             yield break;

```

```

100     }
101
102     validIndexes.Remove(0);
103     if (validIndexes.Count == 0)
104     {
105         Debug.Log("[MAIN] No non-zero event left.");
106         isRunning = false;
107         yield break;
108     }
109
110     eventIndex = validIndexes[UnityEngine.Random.Range(0, validIndexes.Count)];
111 }
112
113 Debug.Log($"[MAIN] Showing event index: {eventIndex}");
114 EventObject.SetActive(true);
115 uiImage.sprite = eventImages[eventIndex];
116 EventObject.GetComponent<Image>().enabled = true;
117 Debug.Log("[SHOW EVENT] EventObject should be visible now.");
118 Time.timeScale = 0f;
119 yield return new WaitUntil(() => IsClicking.clickedYes || IsClicking.clickedNo);
120 if (IsClicking.clickedYes)
121 {
122     switch (eventIndex)
123     {
124     case 0:
125         StartCoroutine(Waitstart());
126         IsClicking.money += 35000;
127         IsClicking.hasWork = true;
128         YesBtnController.startEvent = false;
129         break;
130     case 1:
131         StartCoroutine(Waitstart());
132         IsClicking.money -= 200;
133         IsClicking.happiness -= 3;
134         YesBtnController.startEvent = false;
135         break;
136     case 2:
137         StartCoroutine(Waitstart());
138         IsClicking.money += 500;
139         IsClicking.happiness += 5;
140         YesBtnController.startEvent = false;
141         break;
142     case 3:
143         StartCoroutine(Waitstart());
144         IsClicking.money += 200;
145         IsClicking.happiness += 4;
146         YesBtnController.startEvent = false;
147         break;
148     case 4:
149         StartCoroutine(Waitstart());
150         IsClicking.money -= 20000;
151         new WaitForSeconds(0.5f);
152         if (UnityEngine.Random.Range(0, 2) == 0)
153         {
154             YesBtnController.extraEventIndex = 0;
155             YesBtnController.extraEventMoney = 30000;

```

```

156         YesBtnController.extraEventHappiness = 6;
157         YesBtnController.startEvent = false;
158         break;
159     }
160     else
161     {
162         YesBtnController.extraEventIndex = 1;
163         YesBtnController.extraEventMoney = -20000;
164         YesBtnController.extraEventHappiness = -10;
165         YesBtnController.startEvent = false;
166         break;
167     }
168     case 5:
169         StartCoroutine(Waitstart());
170         IsClicking.money -= 3000;
171         YesBtnController.extraEventIndex = 2;
172         YesBtnController.extraEventHappiness = 6;
173         YesBtnController.startEvent = false;
174         break;
175     case 6:
176         StartCoroutine(Waitstart());
177         IsClicking.money -= 100;
178         IsClicking.happiness -= 2;
179         YesBtnController.startEvent = false;
180         break;
181     case 7:
182         StartCoroutine(Waitstart());
183         IsClicking.money += 3000;
184         IsClicking.happiness += 4;
185         YesBtnController.startEvent = false;
186         break;
187     case 8:
188         StartCoroutine(Waitstart());
189         IsClicking.money -= 300;
190         IsClicking.happiness -= 2;
191         YesBtnController.startEvent = false;
192         break;
193     case 9:
194         StartCoroutine(Waitstart());
195         IsClicking.money += 1000;
196         YesBtnController.startEvent = false;
197         break;
198     case 10:
199         StartCoroutine(Waitstart());
200         IsClicking.money += 2000;
201         IsClicking.happiness += 3;
202         YesBtnController.startEvent = false;
203         break;
204     case 11:
205         StartCoroutine(Waitstart());
206         IsClicking.money -= 200;
207         IsClicking.happiness += 3;
208         YesBtnController.startEvent = false;
209         break;
210     case 12:

```

```

211         StartCoroutine(Waitstart());
212         IsClicking.money += 300;
213         IsClicking.happiness += 2;
214         YesBtnController.startEvent = false;
215         break;
216     case 13:
217         StartCoroutine(Waitstart());
218         IsClicking.money -= 500;
219         IsClicking.happiness += 5;
220         YesBtnController.startEvent = false;
221         break;
222     case 14:
223         StartCoroutine(Waitstart());
224         IsClicking.money += 2000;
225         IsClicking.happiness += 3;
226         YesBtnController.startEvent = false;
227         break;
228     case 15:
229         StartCoroutine(Waitstart());
230         IsClicking.money -= 6000;
231         YesBtnController.extraEventIndex = 3;
232         YesBtnController.extraEventHappiness = 10;
233         YesBtnController.startEvent = false;
234         break;
235     case 16:
236         StartCoroutine(Waitstart());
237         IsClicking.money += 1000;
238         YesBtnController.startEvent = false;
239         break;
240     case 17:
241         StartCoroutine(Waitstart());
242         IsClicking.money += 3000;
243         IsClicking.happiness += 2;
244         YesBtnController.startEvent = false;
245         break;
246     case 18:
247         StartCoroutine(Waitstart());
248         IsClicking.money -= 20000;
249         IsClicking.happiness -= 10;
250         YesBtnController.startEvent = false;
251         break;
252     case 19:
253         StartCoroutine(Waitstart());
254         IsClicking.money -= 10000;
255         YesBtnController.extraEventIndex = 4;
256         YesBtnController.extraEventMoney = -10000;
257         YesBtnController.extraEventHappiness = -10;
258         YesBtnController.startEvent = false;
259         break;
260     case 20:
261         StartCoroutine(Waitstart());
262         IsClicking.money -= 1000;
263         YesBtnController.extraEventIndex = 5;
264         YesBtnController.extraEventHappiness = 5;
265         YesBtnController.startEvent = false;

```

```

266         break;
267     case 21:
268         StartCoroutine(Waitstart());
269         IsClicking.money -= 200;
270         YesBtnController.startEvent = false;
271         break;
272     case 22:
273         StartCoroutine(Waitstart());
274         IsClicking.money += (int)math.round(IsClicking.money * 0.007f);
275         YesBtnController.startEvent = false;
276         break;
277     case 23:
278         StartCoroutine(Waitstart());
279         IsClicking.money -= 300;
280         YesBtnController.startEvent = false;
281         break;
282     case 24:
283         StartCoroutine(Waitstart());
284         IsClicking.money -= 1000;
285         IsClicking.happiness += 5;
286         YesBtnController.startEvent = false;
287         break;
288     case 25:
289         StartCoroutine(Waitstart());
290         IsClicking.money -= 30000;
291         if (UnityEngine.Random.Range(0, 5) == 0)
292         {
293             YesBtnController.extraEventIndex = 6;
294             YesBtnController.extraEventMoney = 50000;
295             YesBtnController.extraEventHappiness = 5;
296             YesBtnController.startEvent = false;
297             break;
298         }
299         else
300         {
301             YesBtnController.extraEventIndex = 7;
302             YesBtnController.extraEventMoney = -20000;
303             YesBtnController.extraEventHappiness = -10;
304             YesBtnController.startEvent = false;
305             break;
306         }
307     case 26:
308         StartCoroutine(Waitstart());
309         IsClicking.money -= 200;
310         YesBtnController.extraEventIndex = 8;
311         YesBtnController.extraEventMoney = 0;
312         YesBtnController.extraEventHappiness = 6;
313         YesBtnController.startEvent = false;
314         break;
315     case 27:
316         StartCoroutine(Waitstart());
317         IsClicking.money -= 5000;
318         if (UnityEngine.Random.Range(0, 6) == 0)
319         {
320             YesBtnController.extraEventIndex = 9;
321             YesBtnController.extraEventMoney = 100000;

```

```

322         YesBtnController.extraEventHappiness = 10;
323         YesBtnController.startEvent = false;
324         break;
325     }
326     else
327     {
328         YesBtnController.extraEventIndex = 10;
329         YesBtnController.extraEventHappiness = -10;
330         YesBtnController.extraEventMoney = 0;
331         YesBtnController.startEvent = false;
332         break;
333     }
334     case 28:
335         StartCoroutine(Waitstart());
336         IsClicking.money -= 1000;
337         YesBtnController.extraEventIndex = 11;
338         YesBtnController.extraEventHappiness = -8;
339         YesBtnController.extraEventMoney = 0;
340         YesBtnController.startEvent = false;
341         break;
342     case 29:
343         StartCoroutine(Waitstart());
344         IsClicking.money -= 1000;
345         if (UnityEngine.Random.Range(0, 20) == 0)
346         {
347             YesBtnController.extraEventIndex = 12;
348             YesBtnController.extraEventHappiness = -5;
349             YesBtnController.extraEventMoney = 0;
350             YesBtnController.startEvent = false;
351             break;
352         }
353         else
354         {
355             YesBtnController.extraEventIndex = 13;
356             YesBtnController.extraEventMoney = 10000;
357             YesBtnController.extraEventHappiness = 5;
358             YesBtnController.startEvent = false;
359             break;
360         }
361     default:
362         Debug.Log("Default case");
363         break;
364 }
365 }
366 else if (IsClicking.clickedNo)
367 {
368     switch(eventIndex)
369     {
370     case 0:
371         StartCoroutine(Waitstart());
372         YesBtnController.startEvent = false;
373         break;
374     case 1:
375         StartCoroutine(Waitstart());
376         YesBtnController.startEvent = false;

```

```

377         break;
378     case 2:
379         StartCoroutine(Waitstart());
380         YesBtnController.startEvent = false;
381         break;
382     case 3:
383         StartCoroutine(Waitstart());
384         YesBtnController.startEvent = false;
385         break;
386     case 4:
387         StartCoroutine(Waitstart());
388         YesBtnController.startEvent = false;
389         break;
390     case 5:
391         StartCoroutine(Waitstart());
392         YesBtnController.extraEventIndex = 14;
393         YesBtnController.startEvent = false;
394         break;
395     case 6:
396         StartCoroutine(Waitstart());
397         YesBtnController.startEvent = false;
398         break;
399     case 7:
400         StartCoroutine(Waitstart());
401         YesBtnController.startEvent = false;
402         break;
403     case 8:
404         StartCoroutine(Waitstart());
405         YesBtnController.startEvent = false;
406         break;
407     case 9:
408         StartCoroutine(Waitstart());
409         YesBtnController.startEvent = false;
410         break;
411     case 10:
412         StartCoroutine(Waitstart());
413         YesBtnController.startEvent = false;
414         break;
415     case 11:
416         StartCoroutine(Waitstart());
417         YesBtnController.startEvent = false;
418         break;
419     case 12:
420         StartCoroutine(Waitstart());
421         YesBtnController.startEvent = false;
422         break;
423     case 13:
424         StartCoroutine(Waitstart());
425         YesBtnController.startEvent = false;
426         break;
427     case 14:
428         StartCoroutine(Waitstart());
429         YesBtnController.startEvent = false;
430         break;
431     case 15:

```

```

432         StartCoroutine(Waitstart());
433         YesBtnController.extraEventIndex = 15;
434         YesBtnController.startEvent = false;
435         break;
436     case 16:
437         StartCoroutine(Waitstart());
438         YesBtnController.startEvent = false;
439         break;
440     case 17:
441         StartCoroutine(Waitstart());
442         YesBtnController.startEvent = false;
443         break;
444     case 18:
445         StartCoroutine(Waitstart());
446         YesBtnController.startEvent = false;
447         break;
448     case 19:
449         StartCoroutine(Waitstart());
450         YesBtnController.startEvent = false;
451         break;
452     case 20:
453         StartCoroutine(Waitstart());
454         YesBtnController.extraEventIndex = 16;
455         YesBtnController.extraEventHappiness = -3;
456         YesBtnController.extraEventMoney = 0;
457         YesBtnController.startEvent = false;
458         break;
459     case 21:
460         StartCoroutine(Waitstart());
461         YesBtnController.extraEventIndex = 17;
462         YesBtnController.extraEventHappiness = -8;
463         YesBtnController.extraEventMoney = 0;
464         YesBtnController.startEvent = false;
465         break;
466     case 22:
467         StartCoroutine(Waitstart());
468         YesBtnController.startEvent = false;
469         break;
470     case 23:
471         StartCoroutine(Waitstart());
472         YesBtnController.startEvent = false;
473         break;
474     case 24:
475         StartCoroutine(Waitstart());
476         YesBtnController.startEvent = false;
477         break;
478     case 25:
479         StartCoroutine(Waitstart());
480         YesBtnController.extraEventIndex = 18;
481         YesBtnController.startEvent = false;
482         break;
483     case 26:
484         StartCoroutine(Waitstart());
485         YesBtnController.extraEventIndex = 18;
486         YesBtnController.extraEventHappiness = -2;

```



```

487         YesBtnController.extraEventMoney = 0;
488         YesBtnController.startEvent = false;
489         break;
490     case 27:
491         StartCoroutine(Waitstart());
492         YesBtnController.startEvent = false;
493         break;
494     case 28:
495         StartCoroutine(Waitstart());
496         YesBtnController.startEvent = false;
497         break;
498     case 29:
499         StartCoroutine(Waitstart());
500         YesBtnController.startEvent = false;
501         break;
502     }
503 }
504
505 Debug.Log("[MAIN] Yes clicked");
506 IsClicking.clickedYes = false;
507 IsClicking.clickedNo = false;
508 IsClicking.isClick = false;
509 isRunning = false;
510 EventObject.GetComponent<Image>().enabled = false;
511 uiImage.sprite = null;
512 eventImages[eventIndex] = null;
513 Time.timeScale = 1f;
514 if (eventIndex == 0)
515 {
516     haveWork = true;
517 }
518 IsClicking.clickedYes = false;
519 IsClicking.isClick = false;
520 isRunning = false;
521 if (YesBtnController.extraEventIndex != -1)
522 {
523
524     if (isExtraEventProcessed)
525     {
526         Debug.Log("[EXTRA] 額外事件已處理過，跳過這一輪。");
527         yield break; // 正確結束 Coroutine
528     }
529     Debug.Log("[EXTRA] 顯示追加事件圖片");
530
531     uiImage.sprite = eventImagesSpecial[YesBtnController.extraEventIndex];
532     EventObject.SetActive(true);
533     EventObject.GetComponent<Image>().enabled = true;
534     IsClicking.clickedYes = false;
535     IsClicking.clickedNo = false;
536     Time.timeScale = 0f;
537     yield return new WaitUntil(() => IsClicking.clickedYes || IsClicking.clickedNo);
538     Time.timeScale = 1f;
539     isExtraEventProcessed = true;
540     YesBtnController.extraEventIndex = -1;
541     EventObject.GetComponent<Image>().enabled = false;
542     uiImage.sprite = null;
543
544     IsClicking.money += YesBtnController.extraEventMoney;
545     IsClicking.happiness += YesBtnController.extraEventHappiness;
546     Debug.Log($"[EXTRA EFFECT] 金錢變動: {YesBtnController.extraEventMoney}, 幸福變動: {YesBtnController.extraEventHappiness}");
547     YesBtnController.extraEventHappiness = 0;
548     YesBtnController.extraEventMoney = 0;
549     YesBtnController.extraEventIndex = -1;
550
551     IsClicking.clickedYes = false;
552     IsClicking.clickedNo = false;
553 }
554 }
555
556 }
557
558 }
559 }

```

說明：

1-6 列基本 Unity 函式庫等

11 列一組圖片陣列，每一張對應一個普通事件。

12 列事件顯示用的物件。

13 列帳單畫面（有時候事件結束會顯示）。

eventIndex：目前進行到哪一個事件

haveWork：玩家有沒有工作

uiImage：自己身上的 Image 元件（讓 eventObject 顯示圖）

isRunning：控制是否有事件正在進行中，避免重複觸發

34 列開始時，先把事件畫面關閉，同時啟動 EventCycle() 協程，不斷檢查玩家有沒有點擊事件。

43 列 EventCycle(), 持續檢查是否超過一年, 如果玩家點擊過並且目前沒有正在跑事件, 開始 Main() 主要事件流程, 如果已經超過 52 週時顯示一個“進入結算畫面”按鈕。

66 列處理一次事件的流程，隨機等待 0~7 秒後才出現事件（讓節奏不要太固定）

73 列找出還有圖片的事件編號，避免顯示已經用過的，如果沒有可用事件了，直接結束，不繼續做，如果還沒找到工作，強制顯示 `eventIndex=0`（找工作事件），有工作的人，會有 50% 機率「今天不出現事件」，移除「找工作」這個選項，如果刪除後沒剩下可以選的，就直接跳出。

110 列隨機選一個事件編號。

114 列顯示事件圖片，暫停遊戲（時間停止，等玩家選 Yes/No），重置所有點擊狀態，關掉圖片，清空這個 event 的圖片（防止重複），恢復時間流動。

124 列如果是找工作的事件，玩家標記為已找到工作。

130 列如果是其他事件，依照事件編號執行事件，改變金錢及開心度。

521 列如果 YesBtn 有觸發「追加事件」，顯示追加事件圖片，再次暫停，等玩家處理，加上追加事件的金錢、開心度變化，清除追加事件資訊，整個 Main() 流程結束。

YesBtnController.cs

```

1 using System.Collections;
2 using System.Net.Security;
3 using TMPro;
4 using UnityEngine;
5 using UnityEngine.Events;
6 using UnityEngine.UI;
7 using UnityEngine.SceneManagement;
8
9 public class YesBtnController : MonoBehaviour
10 {
11     public GameObject YesButtonObject;
12     public GameObject EventObject;
13     public static bool startEvent = false;
14     public static int extraEventIndex = -1; // -1 代表没有追加事件
15     public static int extraEventMoney = 0;
16     public static int extraEventHappiness = 0;
17
18     public IEnumerator WaitStart()
19     {
20         yield return new WaitForSeconds(1);
21     }
22
23     public void Start()
24     {
25         HideYesButton();
26     }
27
28     public void Update()
29     {
30         bool ok = EventController.eventIndex == 0 || EventController.eventIndex == 1 || EventController.eventIndex == 2 || EventController.eventIndex == 3 || EventController.eventIndex == 6 || EventController.eventIndex == 7 || \
31             EventController.eventIndex == 8 || EventController.eventIndex == 9 || EventController.eventIndex == 10 || EventController.eventIndex == 11 || EventController.eventIndex == 12 || EventController.eventIndex == 13 || \
32             EventController.eventIndex == 14 || EventController.eventIndex == 16 || EventController.eventIndex == 17 || EventController.eventIndex == 18 || EventController.eventIndex == 22 || EventController.eventIndex == 24 || (extraEventIndex != -1 && EventObject != null && EventObject.GetComponent<Image>().enabled == true);
33         bool yes = EventController.eventIndex == 4 || EventController.eventIndex == 5 || EventController.eventIndex == 15 || EventController.eventIndex == 19 || EventController.eventIndex == 20 || EventController.eventIndex == 21 || \
34             EventController.eventIndex == 23 || EventController.eventIndex == 25 || EventController.eventIndex == 26 || EventController.eventIndex == 27 || EventController.eventIndex == 28 || EventController.eventIndex == 29;
35         Image eventImage = EventObject.GetComponent<Image>();
36         if (eventImage != null && eventImage.enabled && ok)
37         {
38             ShowYesButton("好的");
39         }
40         else if (eventImage != null && eventImage.enabled && yes)
41         {
42             ShowYesButton("要");
43         }
44     }
45 }

```

```

50     }
51     else
52     {
53         HideYesButton();
54     }
55 }
56
57
58 2 個參數
59 void ShowYesButton(string showString)
60 {
61     YesButtonObject.GetComponent<Image>().enabled = true;
62     YesButtonObject.GetComponent<Button>().enabled = true;
63     TMP_Text tmpText = YesButtonObject.GetComponentInChildren<TMP_Text>();
64     if (tmpText != null)
65     {
66         tmpText.enabled = true;
67         tmpText.text = showString;
68     }
69 }
70
71 2 個參數
72 void HideYesButton()
73 {
74     YesButtonObject.GetComponent<Image>().enabled = false;
75     YesButtonObject.GetComponent<Button>().enabled = false;
76     TMP_Text tmpText = YesButtonObject.GetComponentInChildren<TMP_Text>();
77     if (tmpText != null)
78     {
79         tmpText.enabled = false;
80     }
81 }
82
83 0 個參數
84 public void OnClick()
85 {
86     IsClicking.clickedYes = true;
87     IsClicking.isClick = true;
88 }

```

說明：

1-7 列基本 Unity 函式庫等

11 列 YesButtonObject：指向畫面上的「Yes」按鈕

EventObject：指向畫面上正在顯示的事件圖片物件

startEvent：有沒有觸發新的事件

extraEventIndex：有沒有「追加事件」（-1 代表沒有）

extraEventMoney、extraEventHappiness：追加事件給的獎懲

20 列等到 startEvent == true 再繼續

25 列一開始的時候，把 Yes 按鈕隱藏起來

32 列 Update()，每一幀檢查，判斷什麼時候要出現 OK 按鈕，這些事件是屬於「一般確定」型態，例如，找工作、撿到錢之類的，什麼時候要出現 Yes 按鈕，這些事件是「需要作選擇」型態，例如「要不要投資」「要不要冒險」等等。如果目前事件正在顯示，而且屬於 ok 類型 → 顯示「好的」，如果屬於 yes 類型 → 顯示「要」，否則隱藏 Yes 按鈕。

58 列顯示按鈕，打開文字，並改成指定文字（「好的」或「要」）。

70 列把按鈕和文字全部關閉。

81 列最關鍵：按下「Yes」的處理，當玩家按下「Yes」這個按鈕時觸發，通知整個遊戲：

「玩家已經點了 Yes！」

NoBtnController.cs

```
1 using System.Collections;
2 using TMPro;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 〇 標註
7 public class NoBtnController : MonoBehaviour
8 {
9     〇 標註
10     public GameObject NoButtonObject;
11     〇 標註
12     public GameObject EventObject;
13     〇 標註
14     public GameObject WeekObject;
15
16     〇 標註
17     private IEnumerator Waitstart()
18     {
19         yield return new WaitUntil(() => YesBtnController.startEvent);
20     }
21
22     〇 標註
23     void Start()
24     {
25         HideNoButton();
26     }
27
28     〇 標註
29     void Update()
30     {
31         bool show = EventController.eventIndex == 4 || EventController.eventIndex == 5 || EventController.eventIndex == 15 || EventController.eventIndex == 19 || EventController.eventIndex == 20 || EventController.eventIndex == 21 || \
32         EventController.eventIndex == 23 || EventController.eventIndex == 25 || EventController.eventIndex == 26 || EventController.eventIndex == 27 || EventController.eventIndex == 28 || EventController.eventIndex == 29;
33         Image eventImage = EventObject.GetComponent<Image>();
34         if (eventImage != null && eventImage.enabled && show && YesBtnController.extraEventIndex == -1)
35         {
36             ShowNoButton("不要");
37         }
38         else
39         {
40             HideNoButton();
41         }
42     }
43
44     〇 標註
45     void ShowNoButton(string showString)
46     {
47         NoButtonObject.GetComponent<Image>().enabled = true;
48         NoButtonObject.GetComponent<Button>().enabled = true;
49         TMP_Text tmpText = NoButtonObject.GetComponentInChildren<TMP_Text>();
50         if (tmpText != null)
51         {
52             tmpText.enabled = true;
53             tmpText.text = showString;
54         }
55     }
56
57     void HideNoButton()
58     {
59         NoButtonObject.GetComponent<Image>().enabled = false;
60         NoButtonObject.GetComponent<Button>().enabled = false;
61         TMP_Text tmpText = NoButtonObject.GetComponentInChildren<TMP_Text>();
62         if (tmpText != null)
63         {
64             tmpText.enabled = false;
65         }
66     }
67
68     〇 標註
69     public void OnClick()
70     {
71         IsClicking.clickedNo = true;
72         IsClicking.isClick = true;
73     }
74 }
```

說明：

與 YesBtnController 基本完全相同架構，不同的是在於「可選擇事件」出現時，才會顯現。

isClicking.cs

```
1 using UnityEngine;
2
3 public class IsClicking : MonoBehaviour
4 {
5
6     public static bool isClick = false;
7     public static bool isShown = true;
8     public static int money = 10000;
9     public static int happiness = 50;
10    public static bool hasWork = false;
11    public static int week = 1;
12    public static bool clickedYes = false;
13    public static bool clickedNo = false;
14
15    public void OnClick()
16    {
17        Debug.Log("[CLICK] 點擊了事件觸發!");
18        isClick = true;
19    }
20
21 }
22 }
```

說明：

是這個遊戲最關鍵的靜態變數

6 列玩家是否正在點擊事件，被 EventController、YesBtnController、NoBtnController 各種地方讀取與控制。

7 列目前是否允許顯示「開始按鈕」或其他 UI，例如：正在走路的時候，按鈕會自動隱藏 (isShown = false)。

8 列玩家初始金錢，開始時預設 10000，後續各種事件、工資、消費會影響這個值，顯示在畫面上透過 moneycontroller 更新。

9 列玩家初始開心度，預設是 50 分，被各種事件增減（例如工作太累會扣、娛樂活動會加），顯示在畫面上透過 moodcontroller 更新。

10 列玩家一開始是失業的，透過某些事件（比如 eventIndex=0）找到工作後，變成 true，工作會影響領薪水、開銷計算（比如 Bill() 裡有檢查）。

11 列遊戲的時間進度（第幾週），開始是第一週，每次行走一輪或處理一次事件，week 就加 1，最後玩到第 52 週跳出結尾按鈕。

12 列記錄玩家最近一次點的是 Yes 還是 No，由 YesBtnController 和 NoBtnController 來設定，EventController 那邊會等到這兩個變數其中之一變成 true，才繼續下一步。

15 列點擊事件觸發當玩家手動按了一個「啟動事件」的按鈕（可能是 Start 按鈕等）時呼叫，會把 isClick 變成 true，讓 EventController 進入新一輪事件流程。

Moneycontroller.cs

```
1 using UnityEngine;
2 using UnityEngine.UI;
3
4 public class moneycontroller : MonoBehaviour
5 {
6     // Start is called before the first frame update
7     public GameObject moneyObject;
8     void Start()
9     {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16         moneyObject.GetComponent<Text>().text = IsClicking.money.ToString();
17     }
18 }
```

說明：

每一幀即時更新玩家金錢顯示，在 Update() 裡不斷從 IsClicking.money 取得最新值，並轉成文字顯示。

Moodcontroller.cs

```
1 using UnityEngine;
2 using UnityEngine.UI;
3
4 public class moodcontroller : MonoBehaviour
5 {
6     // Start is called before the first frame update
7     public GameObject moodObject;
8     void Start()
9     {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16         moodObject.GetComponent<Text>().text = IsClicking.happiness.ToString();
17     }
18 }
```

說明：

幾乎和 moneycontroller 一模一樣，只是顯示的是開心度 happiness，它也不斷更新 UI 上對應的文字欄位。

WeekController.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5
6 public class WeekController : MonoBehaviour
7 {
8     // Start is called before the first frame update
9     void Start()
10    {
11    }
12 }
13
14 // Update is called once per frame
15 void Update()
16 {
17     this.gameObject.GetComponent<Text>().text = "第" + IsClicking.week.ToString() + "周";
18 }
19 }
20
```

說明：

每一幀都會把目前週數顯示出來。

EndScreenController.cs

```
1 using System.Collections;
2 using System.Collections.Generic;
3 using UnityEngine;
4 using UnityEngine.UI;
5 using TMPro;
6 public class EndScreenController : MonoBehaviour
7 {
8     // Start is called before the first frame update
9
10    public int Money = IsClicking.money;
11    public int Happiness = IsClicking.happiness;
12    public GameObject EndScreen;
13
14    void Start()
15    {
16    }
17
18    // Update is called once per frame
19    void Update()
20    {
21    }
22
23    Money = IsClicking.money;
24    Happiness = IsClicking.happiness;
25    EndScreen.SetActive(true);
26    EndScreen.GetComponent<Image>().enabled = true;
27    if (Money >= 120000 && Happiness >= 80)
28    {
29        EndScreen.GetComponentInChildren<TMP_Text>().text = $"你在這一年中賺了{Money}元\n你的開心度為{Happiness}\n你達成了目標。心情也很好！\n你是理財生活達人！\n";
30    }
31    else if (Money >= 120000 && Happiness < 80)
32    {
33        EndScreen.GetComponentInChildren<TMP_Text>().text = $"你在這一年中賺了{Money}元\n你的開心度為{Happiness}\n你達成了目標。但心情不太好！\n你是理財生活高手！\n";
34    }
35    else if (Money < 120000 && Happiness >= 80)
36    {
37        EndScreen.GetComponentInChildren<TMP_Text>().text = $"你在這一年中賺了{Money}元\n你的開心度為{Happiness}\n你沒有達成目標。但心情很好！\n你的理財能力還有進步空間！\n";
38    }
39    else
40    {
41        EndScreen.GetComponentInChildren<TMP_Text>().text = $"你在這一年中賺了{Money}元\n你的開心度為{Happiness}\n你沒有達成目標。心情也不好！\n你的理財能力還可以更好！\n記得要多學習理財囉！\n";
42    }
43 }
```

說明：

10 列 Money 和 Happiness：在這邊從 IsClicking 抓目前的金錢與開心度。

12 列 EndScreen：指向結尾畫面。

20 列開啟結尾畫面，啟用圖片顯示（確保有顯示出來），確認數值在哪一個區間，並顯示相應的文字。

三、遊戲試玩回饋問卷資料整理

- 問卷形式：Google 表單線上問卷
- 問題數：10 題
- 試玩人數：14 人
- 回收問卷數：14 份

問題 1：請問你在遊戲結束時的結果是下列哪一個？

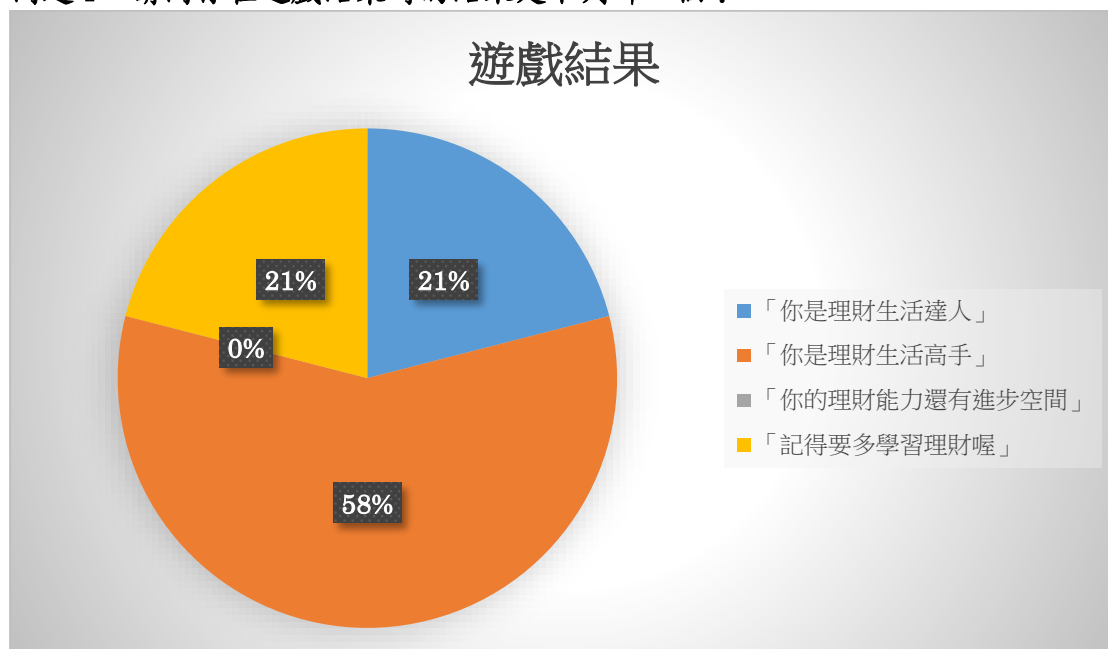


圖 1

由圖 1 可知，超過半數的 58% 試玩者的結果為「你是理財生活高手」，有 21% 的試玩結果為「你是理財生活達人」和「記得要多學習理財喔」。

問題 2：你覺得在遊戲的操作過程中，順暢度如何？ 5 分為非常順暢，1 分為非常不順暢。

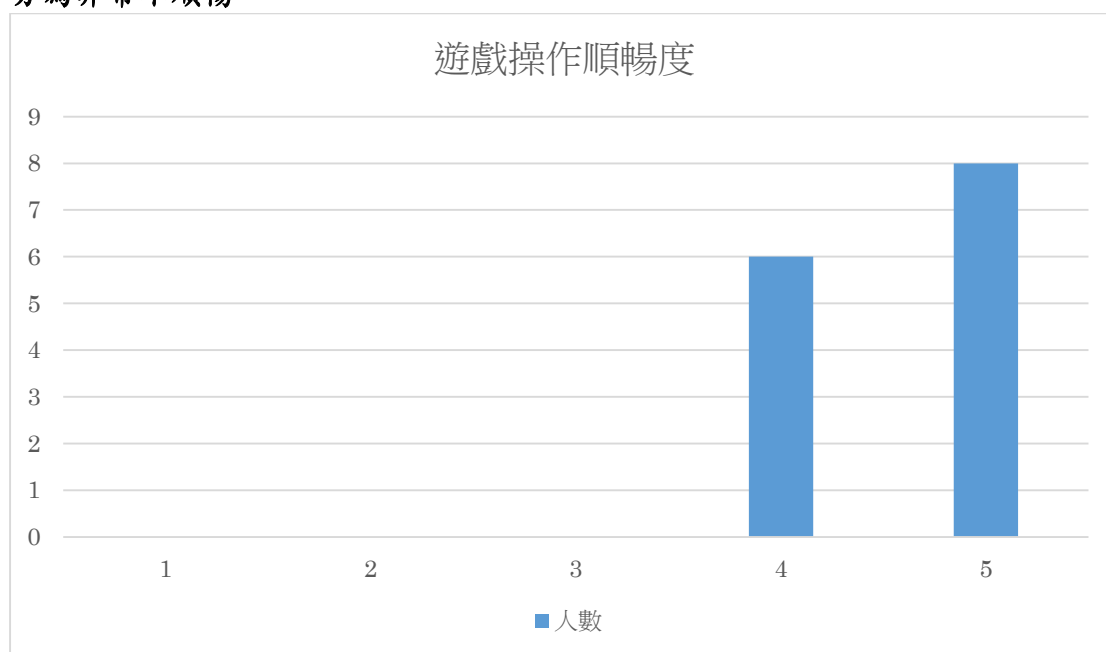


圖 2

由圖 2 可知，所有的試玩者認為遊戲進行是順暢的。

問題 3：你覺得遊戲操作的直覺程度如何？分數 1-5，5 分為非常直覺，1 分為非常不直覺。

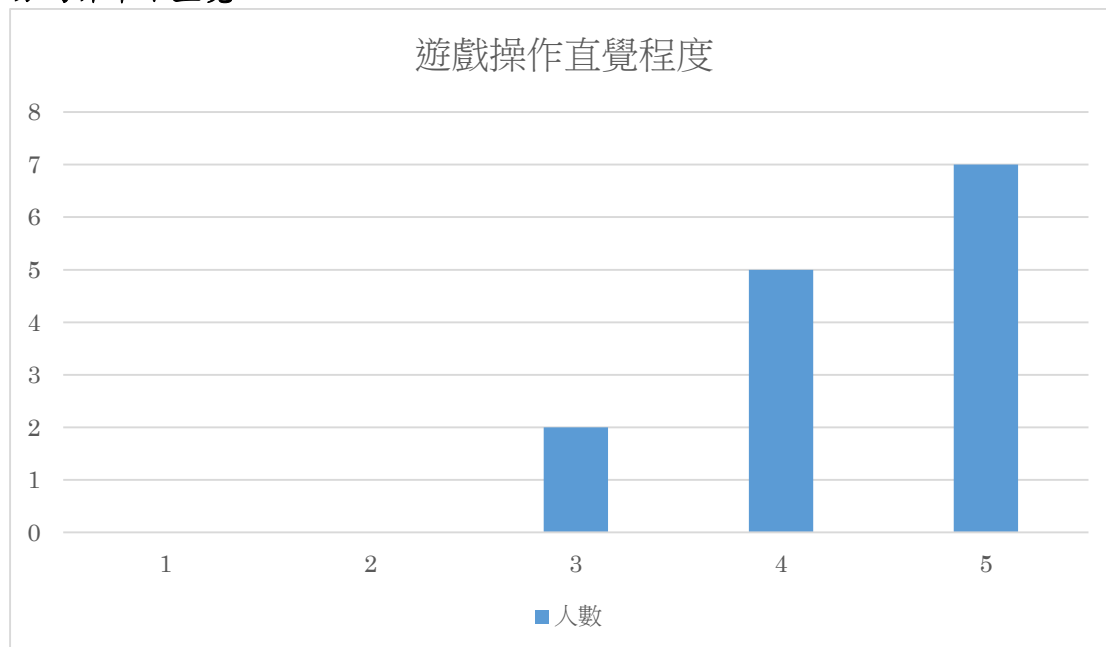


圖 3

由圖 3 可知，大多數試玩者認為遊戲操作偏直覺。

問題 4：你覺得這個遊戲的關卡難度如何？分數 1-5，5 分為非常難，1 分為非常簡單。

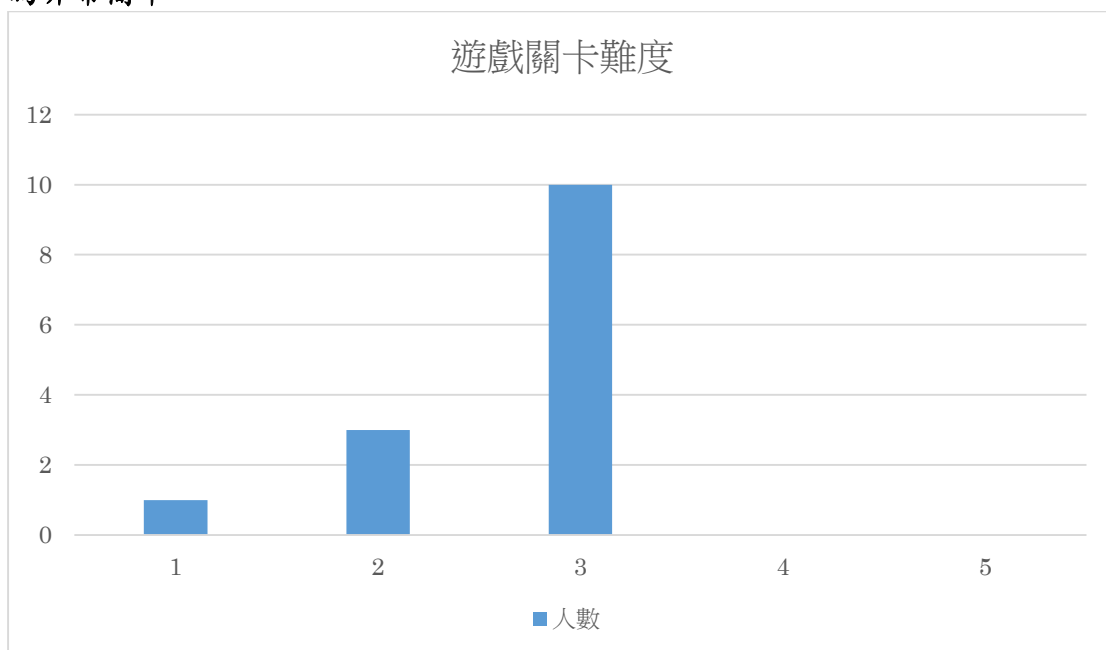


圖 4

由圖 4 可知，大多數試玩者認為遊戲難度普通偏易。

問題 5：你覺得這個遊戲能更幫助你知道如何管理金錢（理財）嗎？分數 1-5，5 分為非常有幫助，1 分為沒有幫助。

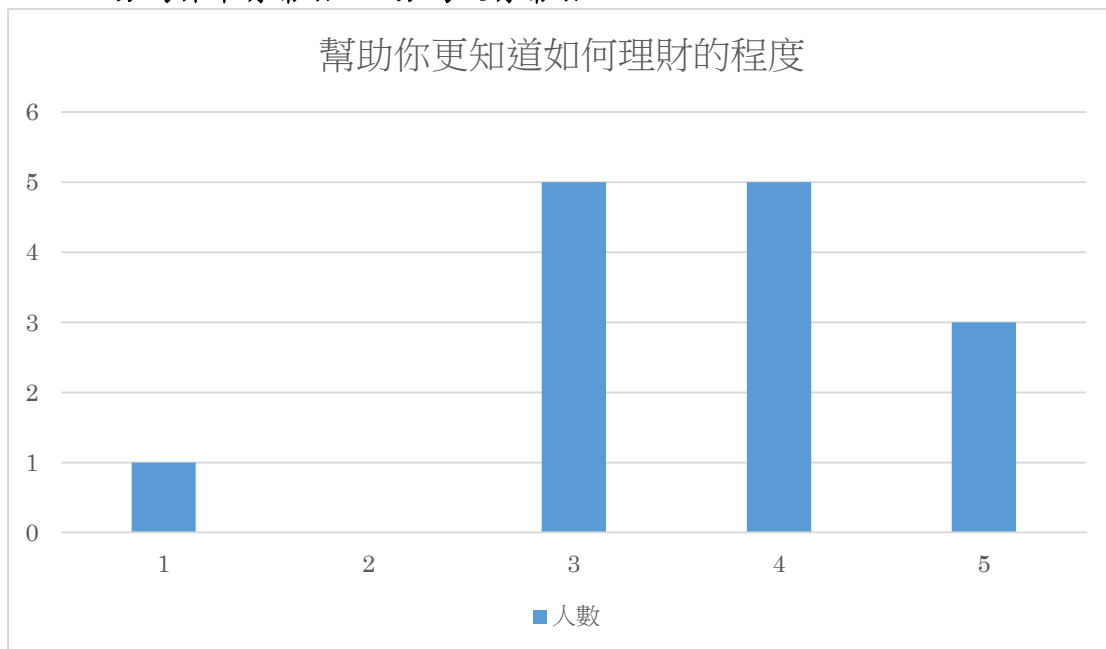


圖 5

由圖 5 可知，大多數的試玩者認為遊戲能幫助自己更了解理財。

問題 6：你覺得透過這個遊戲，能夠讓你對於「理財」這件事更有興趣嗎？分數 1-5，5 分為非常能提升興趣，1 分為無法提升興趣。

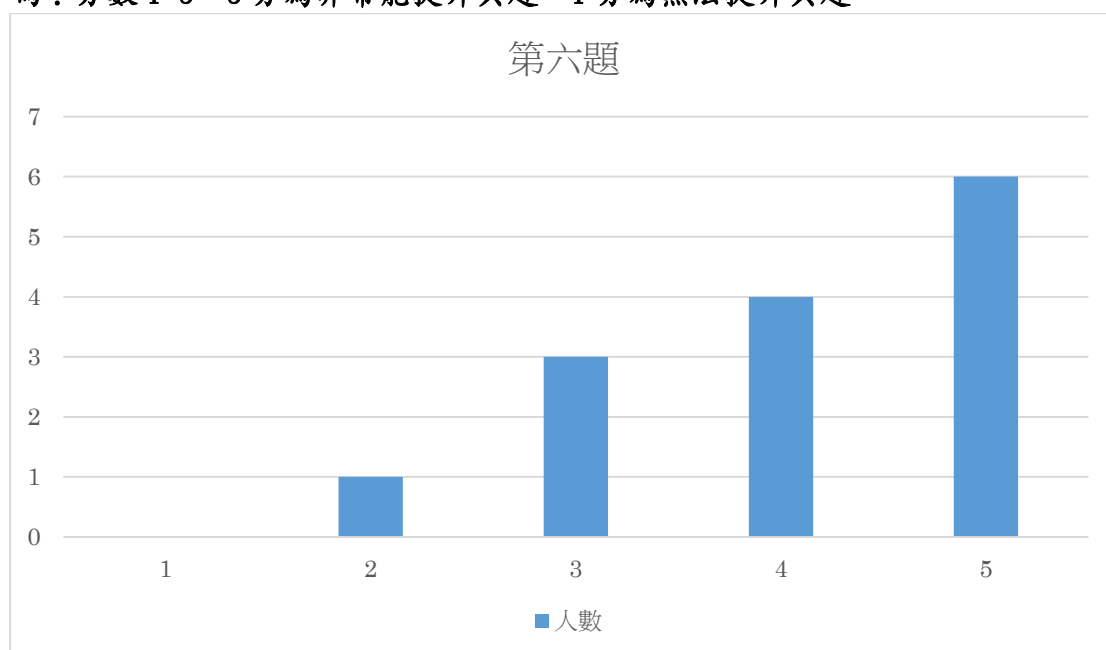


圖 6

從圖 6 可知，大部分的試玩者認為這個遊戲能增加對理財的興趣，部份的人則認為提升理財興趣程度為普通。

問題 7：你學會了哪些理財的技巧？

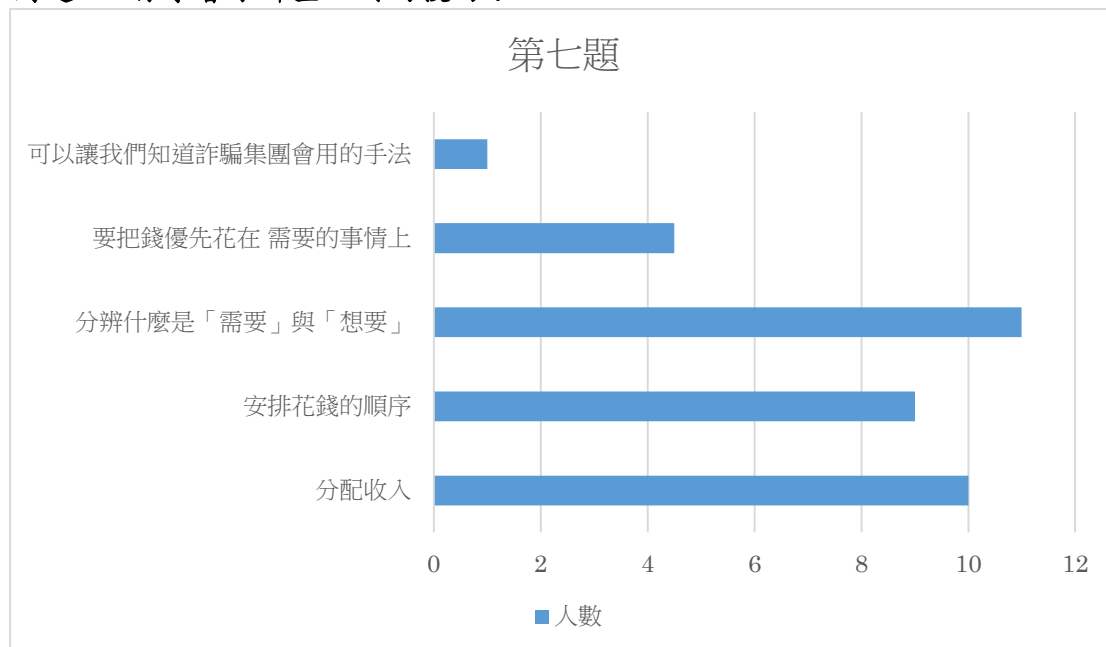


圖 7

由圖 7 可以知道，試玩者透過遊戲能學到「分辨需要與想要」、「分配收入」及「安排花錢順序」等技巧。

問題 8：你在遊戲過程中所遇到最印象深刻的事件為何？

問題 9：承上題，為什麼這個事件讓你印象深刻呢？

問題 8 回應	問題 9 回應
選擇是否要買東西	因為如果買了就會浪費錢, 但不買開心度就會變低
鈔票不見	因為我不會讓鈔票不見
有人在走路	有模擬人走路的樣子
朋友希望和你合作開店	一下損失 40000 元
第四十幾週的事件	因為我的心情值從 85 直接被扣到 77www
被詐騙	開心到 54 錢不夠 120000 也剩下沒幾週了
開餅乾店	這是一個很重要的比較需思考的問題
被投資詐騙 20000 元	因為太多錢了, 我不會隨便去投資網路上的東西
動物園	因為很少錢
被詐騙很多錢錢	因為被騙了很多錢
發票中獎 10000 元	因為增加了收入, 還讓心情變好
要同時滿足心情和需求	因為有一點困難
在第五周買的壺和理財的方法	這個壺讓我財富自由 WoW
拿到了 150000 元	沒有想過可以賺那麼多錢

從上表可知，大部分試玩者較印象深刻的事件是會讓自己一次賺到很多錢或少了很多錢的事件，原因是因為會造成大量金錢變化，影響後面的抉擇。

問題 10：你有其他想法、建議或是回饋想要和我說的，可以寫在這裡！

(自由填答)

無
做的很棒沒有建議
可以提高流暢度因為遊戲到一半有點卡 w
可以在難一點 事件可以增加 有時候會感到有些許的無聊 但還是蠻好玩的
增加事件出現率
沒有
人物好像有點卡 Bug
可以讓關卡再難一些
很有趣
我在第 20 周的時候就已經達成目標了 所以虛妄可以縮短一點 QoQ
有時候人偶會陷進地板，但不會影響多

從上表可看到，試玩者提出覺得事件出現率較低，遊戲的人物會有畫面問題，關卡難度低等建議。

第五章 結論與建議

一、研究結論

1. 理財素養是什麼？

理財素養是指個人具備理解與運用金融知識的能力，能有效管理金錢並做出正確決策。隨著科技與金融商品日益多元，理財素養越來越重要，應從小培養。其核心包括：懂得避免與償還債務、規劃預算與儲蓄、具備投資與資產分析能力、為退休與突發事件預做準備，以及提升防範金融詐騙的能力。

2. 現有的理財素養課程有哪些？

理財素養課程有很多種，例如：

- 學術整合：理財概念可與數學、社會研究或經濟學等課程結合，讓學生學會預算管理和利息計算。
- 實作教學：國外教師透過模擬日常生活的方式教導學生理財，包括管理零用錢和投資等基本概念。
- 專業課程：銀行等金融機構協助設計理財素養課程，針對生活情境介紹各類支付與投資工具。

3. 現有的教育用途遊戲有哪些？

透過遊戲化方式提升學習成效，可以有以下類型：

- 角色扮演與情境模擬遊戲-如《碳足跡課程》，結合角色扮演與多媒體動畫，讓學生在模擬關卡中學習環保知識，並透過小考評估學習成果。
- 嚴肅遊戲分類-分為結構式遊戲化（成本低、趣味性較弱）與內容式遊戲化（融入遊戲元素、開發成本高、接受度高）。
- 生活化理財遊戲-如「理財媽媽」教學案例，運用零用錢分配與365存錢法，引導兒童理解儲蓄、消費與捐款的概念，使理財教育變得有趣又實用。

4. Unity 遊戲引擎是什麼？

Unity 引擎是由 Unity 公司開發的遊戲引擎，可以讓使用者在較簡單的環境製作 3D 或 2D 遊戲，Unity 在網路上可以找到的資源也較多。

5. 如何運用 Unity 進行理財素養遊戲程式設計？

利用 Unity 設計理財素養遊戲必須先規劃遊戲內容，將遊戲的運作流程都規劃好後，要尋找或用繪圖軟體繪製遊戲圖像及遊戲裡的 3D 模型，再用 Unity 將 3D 模型和圖像搭建場景，並使用程式碼編輯器或 IDE 進行程式編輯，用 Unity 附加到各個物件上，在確保遊戲運行順暢。

6. 透過理財遊戲培養國小六年級生理財素養的成效如何？

根據問卷回覆內容，大多數試玩者的結果是金錢大於等於 120000 元，但是開心度小於 80，所以大多數試玩者較注重金錢。大多數試玩者認為遊戲體驗順暢度及直覺度良好，能增加對理財學習的興趣。大多數的試玩者認為遊戲的難度低，一次增加或減少很多金錢的事件會使試玩者印象深刻。

二、研究建議

因為我是第一次使用 Unity 遊戲引擎製作 3D 遊戲，所以品質不甚理想，所以有很多問題，例如：人物在走路時會下陷，人物有時會穿過部分物件，事件顯示比例不對等問題，所以希望如果可以改良這個遊戲，能把現有問題解決，並採用試玩者提出的意見，將角色走路時間縮短，將週數減少等，還可以重新設計遊戲圖像和場景，讓遊戲畫面更美觀。如果可以，還會增加事件的數量和遊戲要素，讓學習範圍增加，並增加特效或動畫等增加遊戲趣味性的要素，增加遊玩此遊戲及學習的意願。

三、研究心得

這次的研究耗費了我大量的時間與精力，因為我從未使用過 Unity 這個遊戲引擎，因此在開始創作遊戲之前，花了相當多的時間查閱各種資料，自學 Unity 引擎的操作與相關功能。由於缺乏經驗，我只能從最基本的知識學起，當遇到尚未學過的功能時，便需再花時間查詢更多資料。有時候會遇到難以解決的錯誤，必須花費許多時間逐步排除與嘗試修正，這些問題往往會延誤整體進度。此外，Unity 引擎在執行時會佔用大量 CPU 資源，導致電腦在製作過程中出現當機的情況，使我不得不暫停工作、重新開機，也讓原定的製作進度受到影響。

在遊戲製作的過程中，我經常無法如期完成各項階段任務，讓我感受到不小的壓力，甚至一度擔心無法在期限內完成這份獨立研究報告。不過，儘管面對困難，我仍希望能夠堅持完成這次的研究，因此花了大量的時間投入在每一個細節的修改與調整上。由於長時間盯著螢幕進行開發與測試，研究期間也造成視力的明顯下降，有時甚至會出現眼睛疲勞或頭痛的情形。總而言之，這次的研究歷程雖然十分艱辛，但透過不斷的學習與實作，我獲得了非常寶貴的經驗，也提升了自己在程式設計與問題解決上的能力。

參考文獻

一、書籍

1. 陳子安（譯）（2019）。Unity 遊戲設計育成攻略。臺北市：旗標科技股份有限公司。（北村愛実原著）
2. 盛介中、邱筱雅（2019）。UNITY 從 2D 到 3D:程式設計的第一哩路 X 從零到近戰遊戲。臺北市：五南圖書出版股份有限公司。
3. 邱勇標（2019）。Unity 3D 遊戲設計實戰（第三版）。臺北市：碁峰資訊股份有限公司。

二、期刊

1. 陳明秀、蔡仕廷、張基成（2016）。嚴肅遊戲之角色扮演與情境模擬對於學習成效之影響：以國小五年級碳足跡課程為例。《教育科學研究期刊》，61(4)，1-32。

三、網頁

- [illegible]