壹、前言

一、研究動機

由於海洋污染日漸嚴重，逐漸影響人類生活，我們開始重視並呼籲對於海洋的保護。但近年來政府或民間團體的推動，對於海洋污染的改善都沒有產生極大的影響，就只像是在呼口號，我們認為或許是宣傳的方式與管道不夠貼近大眾的生活。

在資訊化的時代下，在平常就有許多人藉由玩遊戲紓解壓力，或是獲得樂趣，所以我們認為若是將一些小知識以遊戲的形式呈現，可以讓大家較不牴觸學習並且在學習的過程獲得娛樂的反饋。簡幸如(2005)的研究中指出「**遊戲能讓使用者或玩家在遊戲的過程中很容易沈浸其中，遊戲的挑戰性、不可預測性及競爭性是玩遊戲的動力來源，也可以引發玩家的好奇心與內在動機。**」因此，我們希望透過小遊戲的製作讓科技融入生活、感觸更多人，同時讓更多人知道海洋汙染的嚴重性。

二、研究目的

（一）學習並熟悉 python 語法
（二）運用 pygame 製作一款與海洋相關議題小遊戲
（三）學習 Krita 軟體，訓練電腦繪圖技巧
（四）學習 blender 建模軟體，了解建模物體的方法與技巧
（五）藉由此遊戲宣傳海洋保護理念

貳、文獻探討

一、Python 程式語言

Python 是一種高階編程語言，由於它易讀易學，且具有簡潔而清晰的語法，以及豐富的函式庫，使得它成為了一種流行的通用編程語言。楊東翰(2023)指出「**Python 是物件導向的直譯式語言，相較於 Java 和 C 語言，其程式碼敘述方式較為簡潔，故也較易學習。Python 支援多種程式設計範式。**」在石泓昇(2023)的研究也指出「**Python 具有豐富的函式庫和工具生態系統，使得開發人員可以 輕鬆地進行各種任務**」。

綜合上述，本研究整理了 python 優缺點比較表如下表一。

表一、python 優缺點比較表

優點	缺點
語法簡單	執行較為緩慢
工法完整	設計限制導致運行錯誤
應用廣泛	應用程序建構期間使用大量記憶體

表格來源：本研究製作

二、數位遊戲設計之教學模式

自古以來，學者們一直高度評價遊戲的價值，特別是在兒童及健康發展方面的促進作用。古代希臘的 Plato 和 Aristotle 已體認到遊戲對兒童教育的重要性。同樣，西方教育史上的學者，如 Rousseau、Pestalozzi、Froebel 也強調：「**遊戲有作為兒童天生學習工具的潛力**」。這項研究進一步細分遊戲為非數位化和數位化兩類；非數位化遊戲的重

點在於玩家自主創造，形式包括文字遊戲、棋盤和紙牌等。而數位化遊戲則是透過數位遊戲設計軟體或程式設計製作，這也隨著媒體和資訊科技的進步而愈加普及。

三、Blender 3D 建模

在眾多的 3D 建模軟體中，本研究找了 3 套軟體進行比較如下表二。大多數的 3D 建模軟體是需要付費，而 Blender 不僅提供大眾免費的使用，同時亦能擁用建模、UV、烘培、上色、引擎展示的功能。陳宏育(2023)在論文中提到在論文中提到：「**現實生活中，有許多物體將會以 3D 的模式呈現，以至於學習 3D 建模軟體的需求變大，同時透過 3D 建模的製作推廣 Blender。**」Blender 對於學生和獨立開發團隊而言是一套非常適合的程式。因此我們在實作中利用 Blender 製作海底下的世界，以此世界作為遊戲的背景，期望能真實的呈現海底世界。

表二、3D 建模軟體比較表

軟體名稱	Blender	Autodesk Maya	Zbrush
作業系統	Windows、macOS、Linux	Windows、macOS、Linux	Windows、macOS
功能	• Pipeline 工具 • 渲染 • 影片編輯器	• 高階動畫處理 • 骨架綁定 • 動態模擬 • VFX	• Polypainting • 雕刻工具 • Dynamic Subdivision
適用	• 動畫與 3D 建模 • 3D 渲染 • 合成與視覺特效	• 角色動畫製作 • 遊戲與電影設計 • 大型專案	• 角色設計 • 概念藝術家 • 高品質雕塑
費用	免費	每月：235 USD 每年：1875 USD	每月：39 USD 每年：1359 USD

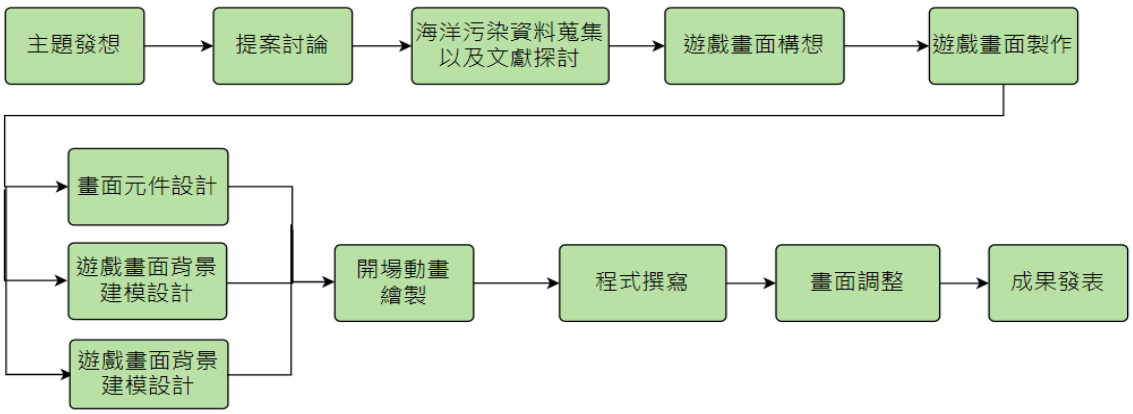
表格來源：本研究製作

參、研究方法

一、研究流程

在製作遊戲的前期，我們先進行主題的構想及討論，接著針對主題產出所需的原件及場景，接著我們透過 python 的程式撰寫製作遊戲，如下圖一。

圖一、研究流程圖



圖片來源：本研究繪製

二、研究工具

表三、研究工具列表

工具	使用方式	說明
VS code	程式撰寫	用於遊戲程式撰寫
Blender	3D 建模	用於繪製遊戲背景、增加遊玩實感
Krita	電腦繪圖	用於製作遊戲開始前動畫
GitMind	流程圖繪製	用於繪製流程圖，使報告能夠清楚呈現

表格來源：本研究製作

參、研究過程與結果

一、遊戲動畫背景及物件繪製

（一）建模背景

1、海洋光影

透過收集網路上的資料，本研究了解如何利用 Blender 簡單的程式來進行海洋光影的編輯。製作過程期初，本研究只把環境顏色調整成暗藍色而已，但發現看起來並不真實。後來，透過收集到的資料重新做出有如陽光灑落進海中的視覺效果。透過此建模瞭解到，光影是至關重要的，因為它會大大的影響整體的氛圍。

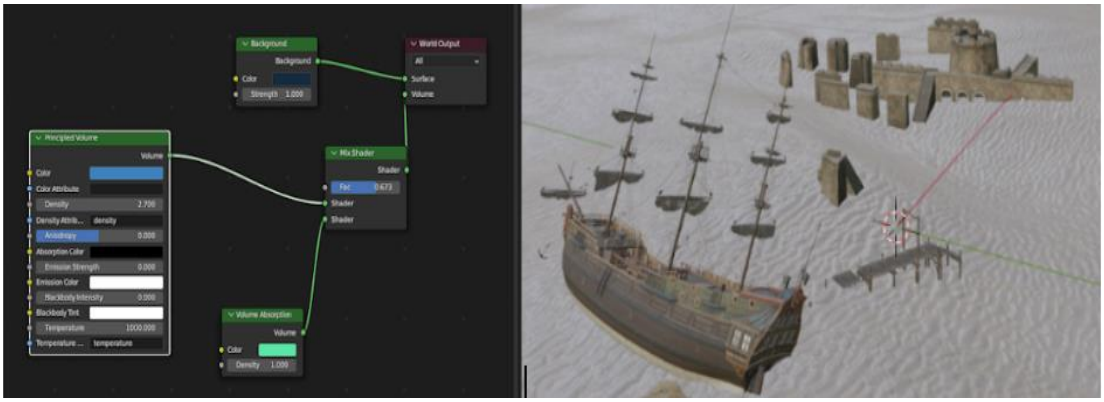
2、製作船隻與背景模型

在參考網路上的教學影片中，要先設置環境背景與設計元件。在製作完成後，需將完成的建模船隻與建築物合併至同一份檔案中，再使用 Poly Haven 中的免費材質庫來編輯模型的材質。接著，調整背景光影。最後利用截圖的方式放到遊戲中做為背景，增加玩家的實感。

圖二、海底場景產出流程圖

圖片來源：本研究繪製

圖三、海底場景產出部分過程



圖片來源：本研究繪製

（二）物件繪圖

1、 電繪元件設計

為了避免顏色與背景顏色相似造成體感不佳，我們在顏色選擇上，以較為明顯、飽和的色彩來呈現。

2、開場動畫繪製

由於海洋的背景大多為冷色，角色整體的配色為亮色系在一開始，本研究是使用手繪的方法進行繪製，之後再將筆稿拍照上傳至手機中的繪圖軟體，接著提取線稿後配色，最後再依照初始設定形象更改完善，以電繪的方式完成角色的最終設計。

圖四、遊戲物件繪製-泡泡、魚、塑膠袋

圖五、動畫部分畫面

圖片來源：本研究繪製

二、遊戲設計模組

（一）模組運用

1、Python 模組(Pygame)

表四、Python 模組列表

功能	模組名稱	說明
圖片及音源載入	**pygame.image.load()** **pygame.mixer.Sound()**	載入遊戲中所需元件的圖片
中文字體存放	**pygame.font.Font()**	從一個字體文件新增一中字體
視窗創建	**pygame.display.set_mode()**	創建一個新的視窗
視窗更新	**pygame.display.update()**	更新視窗內容
遊戲時鐘建立與計時	**pygame.time.Clock()** **clock.tick(FPS)**	定一一個時鐘 一秒鐘更新幾幀畫面，可透過 FPS 數值的改動幀率
遊戲事件存放與取出	**pygame.event**	事件存放
	pygame.event.get()	事件取出
鍵盤輸入	**pygame.key**	與鍵盤相關的模組
	pygame.KEYUP	偵測玩家在放開鍵盤後是否要結束玩家等待的指示
	pygame.K_RIGHT	

	pygame.K_LEFT	偵測鍵盤中左鍵與右鍵是否被按下
遊戲結束	**pygame.quit**	停止載入 pyame 模
對象管理	**pygame.sprite.Sprite**	管理遊戲對象的函式
	pygame.sprite.Group	不同對象的儲存處
增加碰裝偵測準確度	**pygame.sprite.collide_circle**	將系統預設的矩形碰撞判斷改為圓形碰撞判斷
	pygame.draw.circle	畫出實際圓形方便找出適當的圓形偵測半徑

表格來源：本研究製作

2、Python 模組(random)

在海洋遊戲中透過 random 完成以下功能：

物件隨機掉落位置、旋轉、速度：利用 **random.randranged(x,y)**在固定時間內在 x 到 y 之間產生隨機數，使物體從視窗上端寬度中隨機選擇掉落位置、物件旋轉角度、速度。

3、Python 模組(os)

在海洋遊戲中透過 os 完成以下功能：

導入物件路徑連結：利用 **os.path.join()**連接一個或多個路徑段，快速地將物件從正確的資料夾中導入程式內。

三、遊戲設計步驟與程式碼解說

1、步驟一：Pygame 初始化設定

利用 **import** 匯入所需的模塊。接著，設定視窗大小、定義顏色的常數，並用 **pygame.display.set_mode()**建立視窗；用 **pygame.display.sey_caption()**設置視窗標題。最後，設定 **pygame.time.Clock()**變數創建時鐘及其幀率。如圖六所示。

2、步驟二：多媒體檔案導入

初始化設定完成後，將遊戲畫面中會出現的圖片 、音源、字體檔案導入至 VS CODE。利用 os 模塊較有效率地找到檔案所在位置並利用 **os.path.join()**將其導入至 VS CODE 中。如圖七所示。

```
from typing import Any
import pygame
import random
import os
```

```
# 載入音樂
bubble_sound = pygame.mixer.Sound(os.path.join("泡泡音效.mp3"))
plastic_sound = pygame.mixer.Sound(os.path.join("塑膠袋音效.mp3"))
false_sound = pygame.mixer.Sound(os.path.join("錯誤音效.mp3"))
pygame.mixer.music.load(os.path.join("背景音樂.wav"))

#中文字體設定
font_name = os.path.join("NotoSansTC-Regular.ttf")
```

```python
# 遊戲初始化 and 建立視窗
pygame.init()
pygame.mixer.init()
screen = pygame.display.set_mode((wide, height))
pygame.display.set_caption("大海的呼喚")
clock = pygame.time.Clock()  # 創建物件(時間管理及操控)
```

圖六、Pygame 遊戲初始化設定

```python
# 載入圖片
background_img = pygame.image.load(os.path.join("專題簡報製作.jpg")).convert()
bubble_img = pygame.image.load(os.path.join("泡泡.png")).convert()
bottle_img = pygame.image.load(os.path.join("寶特瓶.png")).convert()
pygame.display.set_icon(bottle_img)
player_img = pygame.image.load(os.path.join("水管.png")).convert()
plastic_img = pygame.image.load(os.path.join("塑膠袋.png")).convert()
fish_img = pygame.image.load(os.path.join("魚.png")).convert()
```

圖七、多媒體檔案導入

3、步驟三：計分顯示系統

運用定義 **draw_text()**作為計分顯示系統。其中設定包含分數顯示所在位置、字體大小、顏色、建立在哪個平面。利用 **surf.blit()**將此分數(text)顯示於畫面(surface)上。如圖八所示。

4、步驟四：遊戲初始畫面

運用定義 **draw_init()**作為遊戲初始畫面。首先，導入利用 Blender 繪製出的海底世界。接著，將要顯示在封面的遊戲標題、規則放入 **draw_text()**中，使文字顯示。最後，設定文字大小並定位於螢幕長與寬的比例。如圖九所示。

```python
# 分數顯示設定
def draw_text(surf, text, size, x, y):
    font = pygame.font.Font(font_name, size)
    text_surface = font.render(text, True, White)
    text_rect = text_surface.get_rect()
    text_rect.centerx = x
    text_rect.top = y
    surf.blit(text_surface, text_rect)
```

圖八、分數設置

```python
# 遊戲初始畫面設計
def draw_init():
    screen.blit(background_img, (0, -40))
    draw_text(screen, '大海的呼喚', 150, wide / 2, height / 3)
    draw_text(screen, '← → 可移動飛船 ; Space 可發射子彈', 55, wide / 2, height * 9 / 15)
    draw_text(screen, '泡泡碰到塑膠袋+10分', 55, wide / 2, height * 10 / 15)
    draw_text(screen, '泡泡碰到海洋生物-20分', 55, wide / 2, height * 11 / 15)
    draw_text(screen, '按任意鍵即可開始遊戲', 50, wide / 2, height * 12 / 15)
```

圖九、遊戲初始畫面設計

5、步驟五：泡泡物件功能與樣式設計

定義**__init__()**作為建立物體的方法，包含泡泡的圖片導入、大小調整、去背、圓形碰撞判斷半徑、定位、速度。定義 **update()**判斷鍵盤上的按鍵作用，內容包含按下左右件時泡泡的移動速度。在程式的最後也設定了泡泡元件的左右不可超出了畫面的寬度，如圖十所示。

6、步驟六：塑膠袋與海洋生物物件功能與樣式設計

定義**__init__()**作為建立物體的方法，包含塑膠袋的圖片導入、去背、圓形碰撞判斷半徑、定位、速度。中間部分以定義 **rotate()**旋轉圖片 。詳細內容包含旋轉角度、角度限制及旋轉中心點。下半部以定義 **update()**更新速度、位置等，如圖十一所示。

```python
class Bubble(pygame.sprite.Sprite):
    def __init__(self, bullets):
        pygame.sprite.Sprite.__init__(self)
        self.image = pygame.transform.scale(bubble_img, (220, 414))
        self.image.set_colorkey(Black)
        self.rect = self.image.get_rect()
        self.radius = 155
        #pygame.draw.circle(self.image, Yellow,self.rect.center, self.radius)
```

```python
class Plastic(pygame.sprite.Sprite):
    def __init__(self):
        pygame.sprite.Sprite.__init__(self)
        pygame.sprite.Sprite.__init__(self)
        self.image = plastic_img
        self.image_ori = plastic_img
        self.image_ori.set_colorkey(White)
        self.image = self.image_ori.copy()
        self.rect = self.image.get_rect()
        self.radius = 70
        #pygame.draw.circle(self.image, Yellow,self.rect.center, self.radius)
def rotate(self):
    self.total_degree += self.rot_degree
    self.total_degree = self.total_degree % 360
    self.image = pygame.transform.rotate(self.image_ori, self.total_degree)
    center = self.rect.center
    self.rect = self.image.get_rect()
    self.rect.center = center
```

```python
def update(self):
    key_pressed = pygame.key.get_pressed()
    if key_pressed[pygame.K_RIGHT]:
        self.rect.x += self.speedx
    if key_pressed[pygame.K_LEFT]:
        self.rect.x -= self.speedx

    if self.rect.right > wide:
        self.rect.right = wide
    if self.rect.left < 0:
        self.rect.left = 0
```

圖十、泡泡物件功能與樣式設計

```python
def update(self):
    self.rotate()
    self.rect.y += self.speedy
    self.rect.x += self.speedx
    if self.rect.top > height or self.rect.left > wide or self.rect.right < 0:
        self.rect.x = random.randrange(0, wide - self.rect.width)
        self.rect.y = random.randrange(-100, -40)
        self.speedy = random.randrange(2, 5)
        self.speedx = random.randrange(-3, 3)
```

圖十一、塑膠袋與海洋生物物件功能與樣式設計

7、步驟七：遊戲主迴圈

當此迴圈會被執行(**while running:**)，如果遊戲初始畫面顯示(**show_init:**)後，玩家立即將遊戲關閉(**close**)，程式將中斷執行(**break**)。當 **show_init=False** 時，即不再初始化。迴圈中變數在每次初始畫面顯示及遊戲開始時都會被初始化。如圖十二所示。

8、步驟八：取得輸入與遊戲更新

使用 **for event in pygame.event.get():**從 Pygame 中建立活動(遊戲)。當得分數(**score**)超過 150 分，畫面將會初始化(**show_init=True**)，分數也會歸零。再迴圈跑完之後，整個遊戲執行更新(**update**)。如圖十三所示。

```python
while running:
    if show_init:
        close = draw_init()
        if close:
            break
        show_init = False
        all_sprites = pygame.sprite.Group()
        rocks = pygame.sprite.Group()
        creature = pygame.sprite.Group()
        bullets = pygame.sprite.Group()
```

圖十二、遊戲主迴圈

```python
# 取得輸入
for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
if score >= 150:
    show_init = True
    score = 0

# 更新遊戲
all_sprites.update()
```

圖十三、取得輸入與遊戲更新

9、步驟九：判斷塑膠袋與泡泡相撞

當程式被執行時，圓形碰撞判斷會被執行。當塑膠袋與泡泡相撞時，塑膠袋音效會被撥放、分數會加 10 分(**score += 10**)；被撞到的塑膠袋消失後，會立馬新增一個新的塑膠袋(**all_sprites.add(r)**)。當分數小於等於 0 分時，遊戲畫面會被初始化(**show_init=True**)。如圖十四所示。

10、步驟十：判斷魚與泡泡相撞

當程式被執行時，圓形碰撞判斷會被執行。當魚與泡泡相撞時，系統錯誤音效會被撥放、分數會減 20 分(**score -= 20**)；被撞到的魚消失後，會立馬新增一隻新的魚(**all_sprites.add(c)**)。當分數小於等於 0 分時，遊戲畫面會初始化(**show_init=True**)。如圖十五所示。

```python
# 判斷塑膠袋與泡泡相撞
hits1 = pygame.sprite.spritecollide(player, rocks, True, pygame.sprite.collide_circle)
for hit in hits1:
    plastic_sound.play()
    plastic_sound.set_volume(0.1)
    score += 10
    r = Plastic()
    all_sprites.add(r)
    rocks.add(r)
    player.life += 10
    if player.life <= 0:
        show_init = True
```

圖十四、判斷塑膠袋與泡泡相撞

```python
# 判斷魚與泡泡相撞
hits2 = pygame.sprite.spritecollide(player, creature, True, pygame.sprite.collide_circle)
for hit in hits2:
    false_sound.play()
    false_sound.set_volume(0.7)
    score -= 20
    c = Creature()
    all_sprites.add(c)
    creature.add(c)
    player.life -= 20
    if player.life <= 0:
        show_init = True
```

圖十五、判斷魚與泡泡相撞

11、步驟十一：畫面顯示

利用 **screen.blit()**將利用 Blender 製作的背景圖顯示再畫面當中；利用 **draw_text()**將分數定位於距離畫面頂部 8 像素、位於寬度的中間點、字體大小 50 像素。最後，利用 **pygame0display.update()**將所有撰寫內容被更新。如圖十六所示。

12、步驟十二：結束遊戲

在遊戲迴圈的外部呼叫 **pygame.quit()**，使整個遊戲結束。如圖十七所示。

```python
# 畫面顯示
screen.fill(White)  # 頁面為白色
screen.blit(background_img, (0, -40))
all_sprites.draw(screen)
draw_text(screen, str(score), 50, wide/2, 8)
pygame.display.update()
```

圖十六、畫面顯示

```python
pygame.quit()
```

圖十七、結束遊戲

四、遊戲成果展現

（一）遊戲介紹

遊戲的主題為「大海的呼喚」，希望能夠透過小遊戲呼籲海洋保育相關議題。遊玩時，泡泡接觸海洋生物時扣較多分，代表海洋生物受到的傷害極為嚴重；泡泡接觸垃圾時加較少分，代表垃圾的累積量過多，較難解決也較難達成海洋零垃圾的目標。

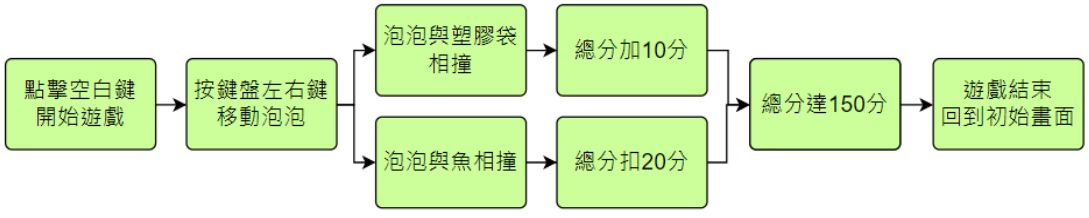
（二）遊玩流程

遊戲初始畫面顯示時，可以按下鍵盤上空白鍵已開始遊戲。

遊戲開始後，當泡泡與垃圾接觸可加 10 分，接觸海洋生物接觸扣 20 分。遊戲中，玩家可透過點及電腦鍵盤的左右鍵，避免泡泡與海洋生物接觸。

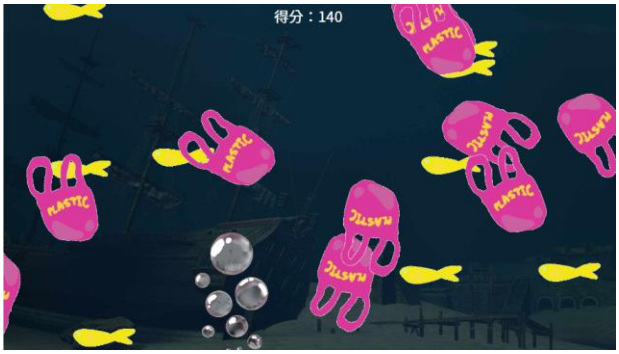
當玩家遊玩時分數達到 150 分，遊戲即結束，畫面將會跳回初始畫面。想傳達的是即便在過程中較為艱難，但在最後依舊能減少海洋的汙染。

圖十八、遊戲遊玩流程圖



圖片來源：本研究繪製

圖十九、遊戲初始畫面



圖二十、遊戲遊玩畫面

圖二十一、遊戲開場動畫

圖二十二、遊戲建模背景

伍、結論與建議

一、研究成果

本研究透過 Python 的 **Pygame** 模組開發了一個海洋遊戲；透過 random 模組使畫面更加生動、豐富；透過 **os** 模組更有效率地尋找檔案所在位置與導入檔案。透過遊戲的開發，可以更順暢地應用 Python 語法同時學習 Pygame 內建的函式。同時，也能夠更了解並熟悉遊戲的開發流程。

在建模與電繪方面，本研究透過 Krita 繪製出一個小短片，畫面為一女孩在海底中漂浮。另外，本研究也利用 3D 建模模擬海洋背景。透過小短片及模擬真實海洋背景的製作引導玩家進入遊戲情境與氛圍當中，以增加玩家的遊戲體驗。

二、發現問題

（一）在兩物件上未觸碰時，就已經執行加扣分以及消去物件的指令。
（二）程式撰寫完畢後，發現當遊戲在初始畫面就被關閉時，終端會顯示錯誤
（三）電繪動畫末期需要調整細節，因圖層較為混亂難以進行細部調整。

三、解決問題

（一）將系統預設的矩形碰撞偵測藉**pygame.sprite.collide_circle**改為圓形碰撞偵測。

（二）利用 ChatGPT 排除錯誤，同時找出更為保險的寫法，避免在邏輯上出現錯誤
（三）預先將所有圖層先進行分配，有利於最後調整細節。

四、建議

（一）增加關卡：在新增的不同關卡中更改速度、旋轉角度等的參數，並加入新型態的敵人與角色，增加遊戲難度、提升遊戲刺激感。

（二）增加元件種類：新增更多不同的海洋垃圾可以讓玩家了解更多關於海洋污染的相關知識。並且在新的海洋垃圾或生物出現時暫停，加入關於這個角色的相關知識背景，增加樂趣的同時也可以讓玩遊戲的人學習到新的知識。

（三）增加可玩性：新增海洋垃圾知識庫，每當移除一垃圾時，畫面可顯示垃圾的相關資訊，使玩家在遊玩的同時也可以吸收新的知識。

陸、參考文獻

一、楊東翰（2023）。**應用 Python 於公車路線票證資料分析-以臺中市公車 33 路為例**。逢甲大學運輸與物流學系：碩士論文。https://hdl.handle.net/11296/4u2696

二、石泓昇（2023）。**資料庫大數據分析結合 Python 系統控制**。國立虎尾科技大學電機工程系碩士在職專班：碩士論文。https://hdl.handle.net/11296/xgaxus

三、簡幸如（2005）。**數位遊戲設計之教學模式建構**。國立中央大學學習與教學研究所：碩士論文。https://hdl.handle.net/11296/pj4763

四、陳宏育（2023）。**次世代 3D 建模、拓撲技術-以 BLENDER 為例**。南臺科技大學多媒體與電腦娛樂科學系：碩士論文。https://hdl.handle.net/11296/825st9