11 打造你的冒險遊戲

11-1 挑戰題目: 過去還是未來

我們可以利用if條件敘述，創立一個冒險遊戲。 遊戲讓你透過故事，挑選自己要做什麼，而不同的決定將導致不同的結局。 為了幫助你開始，可按照使用下面的程式碼定義遊戲：

```
#|echo: true
#|eval: false
#|includes: false

name = input("Enter your name: ")

## 遊戲說明
print(f"Welcome to {name}'s Choose Your Own Adventure game!
As you follow the story, you will be presented with choices that decide your fate.
Take care and choose wisely! Let's begin.")

## 遊戲開始
print(f"You find yourself in a dark room with 2 doors. The first door is red, the second is

##  做出你的選擇(雙向巢狀)
door_choice = input("Which door do you want to choose? Red or White:  ")

if door_choice == "Red":
    print("Great, you walk through the red door and are now in the future! You meet a scien

    choice_one = input("What do you want to do? Accept or Decline:  ")
    if choice_one == "Accept":
        print("""_ _ _ _ _ SUCCESS _ _ _ _ _ : You will helped the scientist save the world!In
        and sends you home! """)
    else:
        print("""_____GAME OVER_____ : Too bad! You declined the sientist's offer and
        now you are stuck in the future!""")

else:
    print("Great, you walked through the white door and now you are in the past! You meet a

    quest_choice = input("Do you want to accept her offer and go on the request, or do you

    if quest_choice == "Accept":
        print("""_ _ _ _ _ SUCCESS _ _ _ _ _ : Princess thanks you for accepting her offer. Yo
    else:
        print("""_____GAME OVER _____ : Well, I guess your story ends here!""")
```

11-2 Capstone專題：打造你的冒險遊戲

11-2-1 文字腳本

文字冒險遊戲，又稱互動故事 (interactive fiction)。

使用者在遊戲過程中做決定、收集物品，或回答問題，是個練習if敘述式和巢狀條件式的好設計。

- 這只是範例，你可以把它變成自己的冒險遊戲，擴充物品、選擇和角色。用這個概念打造更複雜的冒險遊戲是練習Python程式設計技巧的好方法。
- 用流程圖可以讓這個專題更容易規劃和設計程式，每一步都根據流程圖來加入程式碼。每一步完成後都要執行程式碼來確認沒有問題。

故事線：使用者在山中健行，聽到一個聲音。後來迷路了，必須做決定把遊戲破關才能安全回家。

```
歡迎來到聖塔克魯茲山上冒險遊戲
**************************************************
你現在在加州聖塔克魯茲。你在傍晚一個人爬山。
你可以帶一項物品上路：
地圖（m）、手電筒（f）、巧克力（c）、繩子（r）或棒子（s）：
你選哪個？：c
你聽到嗡嗡聲。
要尋找聲音來源嗎？輸入y或n：n
好主意。你沒有冒險。
你開始走回起點。
你發現自己迷路了！
身後的聲音愈來愈大。你開始驚慌！
要開始跑（r）或停下來打電話（c）？：c
電話不通。
要跑（r）或再打一次（c）？：c
電話不通。
要跑（r）或再打一次（c）？：r
你跑很快。聲音變很大。
一個女人騎電動機車從後面接近你。
她問：「我最喜歡的電腦程式語言是什麼？」：PYTHON
她說：「沒錯，Python是我最喜歡的程式語言。如果你有巧克力，我可以幫你。」
幸運的是，你當初選對了！
你把巧克力給她。
她幫助你回家。
恭喜！你成功脫身。你破關了。
```

請同學先看腳本，這是還沒有開始寫程式之前，就該有的部分。也就是，解決問題的運算思維與程式設計部分。如果要將之轉成Python程式碼，一個簡單的方法就是直接在腳本上做修改。換句話說，你腦中的想法已經用中文寫出來了，現在把它轉成Python的語法，就如同轉成英文一樣。寫程式可能更為簡單，只是在關鍵的地方變成Python的語法。

```
歡迎來到聖塔克魯茲山上冒險遊戲
=== === === === ===
你現在在加州聖塔克魯茲。
你在傍晚一個人爬山。
```

~Question 1：爬山帶什麼？ - 什麼選項？

你可以帶一項物品上路：
地圖(map)、手電筒(flash)、巧克力(chocolate)、繩子(rope)或棒子(stick)
你選哪一個？:**chocoloate**

~Question 2：爬山遇險 - 聽到怪聲，要不要冒險？
你聽到嗡嗡聲。
要尋找聲音來源嗎？輸入yes或no：

選擇一：no
好主意。你沒有冒險。
你開始走回起點。
你發現自己迷路了！
身後的聲音越來越大。你開始驚慌！

選擇二：yes
你向聲音接近。
聲音突然停止。
你迷路了！...
你嘗試打電話，但沒有訊號！

=== === === === ===
~**選擇no + 後續故事第1集**

要開始跑(run)或停下來打電話(call)?: **call**
電話不通。
要跑(run)還是再打一次(call)？：**call**
電話還是不通。
要跑(run)還是再打一次(call)？：**run**
你拼命地快跑，而聲音越來越大。

~**選擇no + 後續故事第2集**

一個女人騎電動機車從後面接近你。
她問：「我最喜歡的電腦程式語言是什麼？」：

答案I. Python
她說：「沒錯，*Python* 是我最喜歡的程式語言。如果你有巧克力，我可以幫你。」

(1)幸運的是，你當初選對了！
你把巧克力給她。
她幫助你回家。
恭喜！你成功脫身。你破關了！

(2)沒有巧克力，當初應該選巧克力才對。
    她騎車走了，留下孤單的你。
    你輸了。

答案II. 其他答案，答錯了

她不喜歡你的回答。
他騎車走了，留下迷路的你。
你輸了。
=== === === === ===

～**選擇yes + 後續故事**

Q: 迷路電話不通，選擇前進方向

你要往那個方向前進？ 北(NORTH)、南(SOUTH)、東(EAST)或西(WEST)：

(1) 後續故事第1種：若選北NORTH
你抵達廢棄小屋。
若選地圖，你用地圖找到回家之路。
恭喜！你破關了。

～如果你有地圖，就能找到從這裡回家的路。
---你還在迷路！你輸了。---

(2) 後續故事第2種，若選南SOUTH
你抵達有斷橋的河流。
若選繩子或棒子，你選了可以修好橋樑的物品。
你修好橋樑、過橋，並找到回家的路。
恭喜！你破關了。

～如果你有繩子或棒子，就能修好橋樑。
---你還是在迷路。你輸了。---

(3)後續故事第3種，若選西WEST
你走路時被傾倒的樹絆倒。
你腳受傷了。你坐下等待救援。

～這可能要花很久時間。你還在迷路。
---你輸了。---

(4) 後續故事第4種，若選東EAST，即其他選項
你抵達公路邊。很暗。
若選手電筒，你用手電筒發出訊號。
一輛車停下來，載你回家。
恭喜你！你破關了，安全回到家。

～如果你有手電筒，就能發出求救訊號。
---你還是在迷路。你輸了---

1-2-2 按部就班寫程式

第1步：加入簡介並要使用者做出冒險之旅的決定

- 簡介

```
## 冒險遊戲說明，顯示簡介
print("歡迎來到聖塔克魯茲山上冒險遊戲！")
print("*****************************************")
print("你現在在加州聖塔克魯茲。")
print("你在傍晚一個人爬山。")
print("你可以帶一項物品上路：")
print("地圖(MAP)、手電筒(FLASH)、巧克力(CHOCOLATE)、繩子(ROPE)、或棒子（STICK): ")

## 從使用者取得物品選擇
## Question1:爬山帶什麼
item = input("你選擇那個？ ")
```

- 遇到危險，決定是否要冒險（if條件選擇）

```
## 冒險之旅
print("你聽到嗡嗡聲。")

## 冒險旅程中選擇
## Question2: 遇險 - 聽到怪聲
choice1 = input("要尋找聲音的來源嗎？ 請輸入YES 或 NO: ")
if choice1 == "YES":
    print("你向聲音接近。")
    print("聲音突然停止。")
    print("你迷路了！...")
    print("你嘗試打電話，但沒有訊號！")
else:
    print("好主意。你沒有冒險。")
    print("你開始走回起點。")
    print("你發你發現自己迷路了！")
    print("身後聲音越來越大，你開始驚慌！")
```

第2步：加入更多內容

建立yes/no兩種選擇的可能性後，我們要加入更多程式碼來延伸故事。

- 這個步驟，我們先要**擴充else區塊**的故事內容（選擇一：no）。
  - 我們讓使用者選擇跑或打電話求救。
  - **透過while迴圈**，生動地描述打電話還是跑的雙向(重複)選擇。

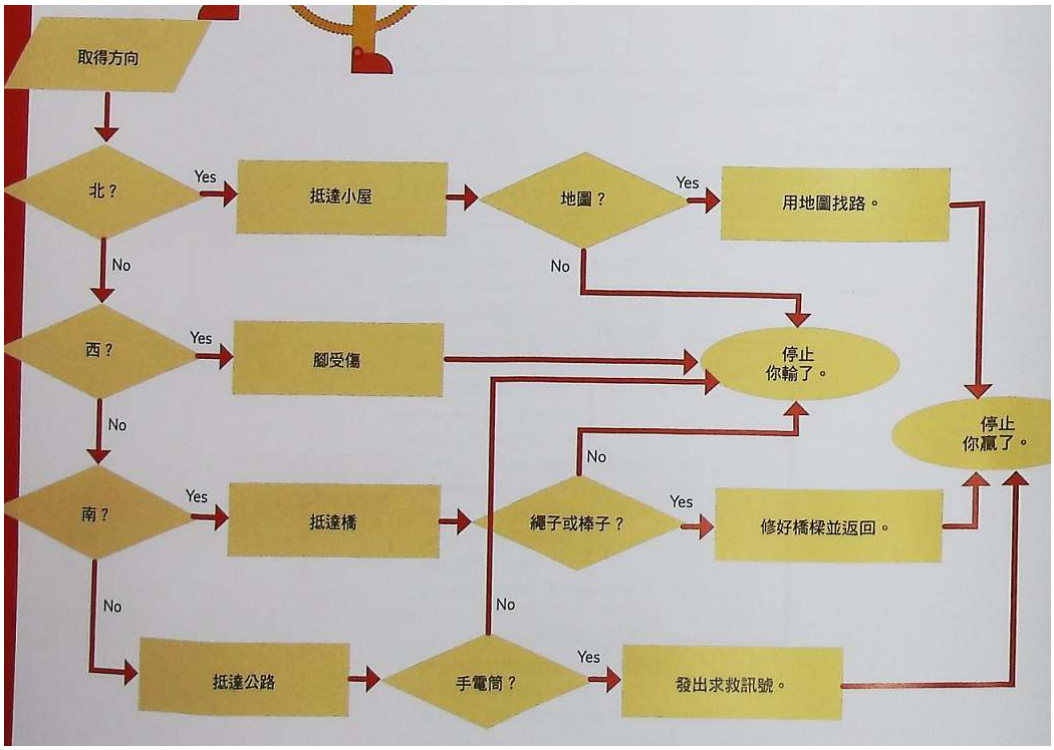
```
## 擴充else區塊的故事內容，在print("身後聲音越來越大，你開始驚慌！")後，加入以下程式碼：
action = input("開始跑或停下來打電話？ 選擇RUN或CALL: ")

## 條件迴圈
while action == "CALL":
    print("電話不通。")
    action = input("要跑或再打一次電話？ 選擇RUN或CALL： ")
print("你拼命地快跑，而聲音越來越大。")
```

第3步：加入方向選擇

第3步：加入方向選擇

建立yes/no兩種選擇的可能性後，我們要加入更多程式碼來延伸故事。

- **擴充if區塊**（選擇二：yes）。
  - 我們讓使用者在聲音停止時選擇前進方向。
    - 這是**巢狀選擇**的例子。
  - **用多向選擇，決定東西南北四個選項中的一個。
    - 各方向選項中，搭配前面攜帶什麼物品的選擇，決定最後的遊戲結果。



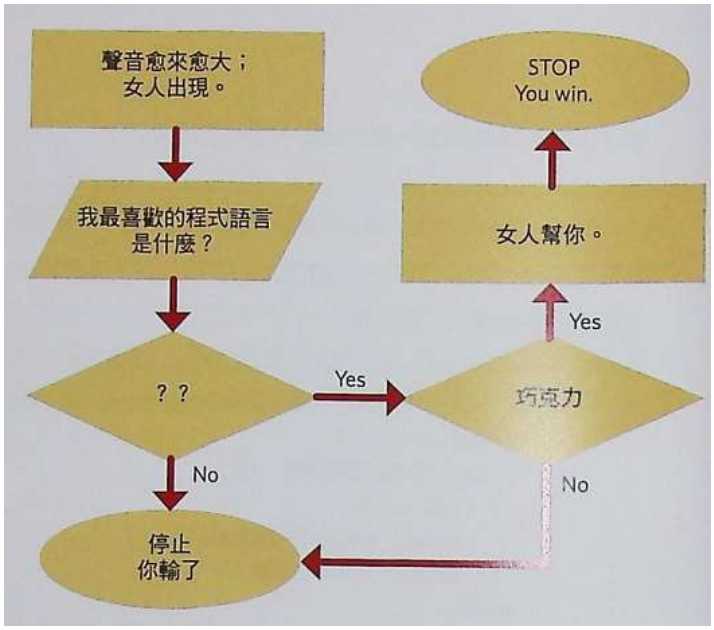

```
## **擴充if區塊**，在print("你嘗試打電話，但沒有訊號！")，加入以下的程式碼：
direction = input("你要往那個方向前進？ 北(NORTH)、南(SOUTH)、東(EAST)或西(WEST)： ")

if direction == "NORTH":
    print("你抵達廢棄小屋。")
    if item == "MAP":
        print("你用地圖找到回家之路。")
        print("恭喜！你破關了。")
    else:
        print("如果你有地圖，就能找到從這裡回家的路。")
        print("---你還在迷路！你輸了。---")
elif direction == "SOUTH":
    print("你抵達有斷橋的河流。")
    if item == "ROPE" or item == "STICK":
        print("你選了可以修好橋樑的物品。")
        print("你修好橋樑、過橋，並找到回家的路。")
        print("恭喜！你破關了。")
    else:
        print("如果你有繩子或棒子，就能修好橋樑。")
        print("---你還是在迷路。你輸了。---")
elif direction == "WEST":
```

```
    print("你走路時被傾倒的樹絆倒。")
    print("你腳受傷了。你坐下等待救援。")
    print("這可能要花很久時間。你還在迷路。")
    print("---你輸了。---")
else:
    print("你抵達公路邊。很暗。")
    if item == "FLASH":
        print("你用手電筒發出訊號。")
        print("一輛車停下來，載你回家。")
        print("恭喜你！你破關了，安全回到家。")
    else:
        print("如果你有手電筒，就能發出求救訊號。")
        print("---你還是在迷路。你輸了---")
```

第4步：加入其他選擇

建立兩種可能性後，我們要加入更多程式碼來延伸故事。

- 這個步驟，我們還要繼續**擴充else區塊**。
    - 我們讓使用者回答問題或謎題，答案正確之後，才能進行下一個動作。




```
## 繼續擴充else區塊，在print("你拼命地快跑，而聲音越來越大。")後，加入以下的程式碼：

## 兩人互動
print("一個路人騎著機車從後面接近你。")

answer = input("她說：『我最喜歡的電腦程式語言是什麼？』； ")
if answer == "PYTHON":
    print("她說：『沒錯，Python是我最喜歡的程式語言。』")
    print("『如果有巧克力，我就幫你。』")
    if item == "CHOCOLATE":
        print("很幸運地，你正好選擇了巧克力。")
        print("你把巧克力給她。")
```

```
        print("她幫助你回家。")
        print("恭喜！你安全脫身，你破關了！")
    else:
        print("沒有巧克力，當初應該選巧克力才對。")
        print("她騎車走了，留下孤單的你。")
        print("你輸了。")
else:
    print("她不喜歡你的回答。")
    print("他騎車走了，留下迷路的你。")
    print("你輸了。")
```

1-3 把遊戲改的更好更有趣

1-3-1 降低錯誤的可能性

第5步：改善使用者輸入並加入錯誤檢查

我們已經有了堪用的遊戲了，繼續來做些改進吧！

- 我們**改善輸入PYTHON的步驟**。
  - 讓使用者能輸入大、小寫和大小寫混合，即Python、python、PYTHON都好。
  - **用or運算子** 讓選擇變成三選一。
  - **用lower()函式**將輸入全變成小寫python。
  - 用upper()轉成PYTHON。

```
## 改善輸入PYTHON的步驟
## 方法（一）：
if answer == "python" or answer == "Python" or answer == "PYTHON"

## 方法（二）：
if answer.lower() == "python":
```

- 我們加入錯誤檢查。
  - 當使用者輸入錯誤時，可以重新輸入YES或NO的選項。
  - 當使用者輸入的不是YES或NO的選項之一時，就重試，所以用not條件判斷。

```
## 加入第一個選擇的錯誤檢查
choice1 = input("要尋找聲音的來源嗎？ 請輸入YES 或 NO: ") # 冒險旅程中的第一個選擇
while not (choice1 == "YES" or choice1 == "NO"):
    choice1 = input("輸入無效，注意是輸入YES 或 NO二選一: ")
```

第6步：在故事加入暫停

為了讓遊戲更順暢，我們可以加入暫停來減緩輸出。這樣讓使用者有時間閱讀，也增加戲劇張力。

- 在遊戲開頭，告訴使用者他迷路了之後，可以先停頓幾秒再繼續。
  - **用time模組中的sleep()函式來插入這種暫停** 。

- 遊戲中許多地方都可以這樣做。

```
import time
time.sleep(3)# 暫停3秒

choice1 = input("要尋找聲音的來源嗎？ 請輸入YES 或 NO: ") #冒險旅程中的第一個選擇
if choice1 == "YES":
    print("你向聲音接近。")
    print("聲音突然停止。")
    time.sleep(3) #暫停3秒讓讀者閱讀
    print("你迷路了！...")
    time.sleep(3) #暫停3秒增加戲劇性
    print("你嘗試打電話，但沒有訊號！")
else:
    print("好主意。你沒有冒險。")
    time.sleep(3) #暫停3秒
    print("你開始走回起點。")
    print("你發你發現自己迷路了！")
    time.sleep(3) #暫停3秒
    print("身後聲音越來越大，你開始驚慌！")
```

1-3-2 改的更好更有趣

1. 進行任何程式設計專題，最好的方式都是一步一步推進。

- 不要一次寫太多程式碼。
- 一個段落運作沒有問題之後，可以隨時回來修改，讓它運作更順利。

2. 後續的延伸

方向一：你可以想一個更精彩的故事！

方向二：你也可以擴充這個遊戲，讓它更好！！

- 在開頭替換或加入更多可以選則攜帶的物品。
- 替換或加入更多路人問的問題和產生的動作。
- 加入更多問題或是可以贈送、交換、交易的物品，讓謎題更豐富。
- 加入更多的錯誤檢查，檢查是否所有的輸入是有效的。
- 加入更多的使用者回應方式，除了YES和NO之外。
- 加入在不同關卡會變化的能量變數。
- 用sleep()函式加入更多的暫停，讓遊戲更順暢。
- 加入一些文字圖片，讓輸入更美觀。

~ 在故事中加入更多複雜性和決策，可以讓它更精彩。

~ 用各種想法擴充這個遊戲，用你的創意和學到的Python程式碼。