

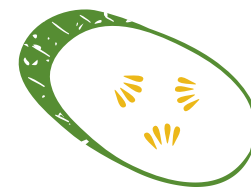
**CAUTION**

**FOOOD!!!**

# 食安小視窗

學號：111504503

姓名：陳姝蓉





# 目錄



- 動機
- 引入的函式庫
- 視窗介面 & 程式碼講解
- 未來改進
- 收穫與心得



# 動機



由於最近食安事件頻傳，有許多人因為食物中毒而就醫，甚至有人因為吃了一頓晚餐，就從此與親人天人永隔，所以我想在食安逐漸被重視的現在，做一個互動性的食安小視窗，希望藉由這個小視窗增進大家對於食品安全的意識，並在這個小視窗中結合了爬蟲的應用，讓資料可以及時得到更新。



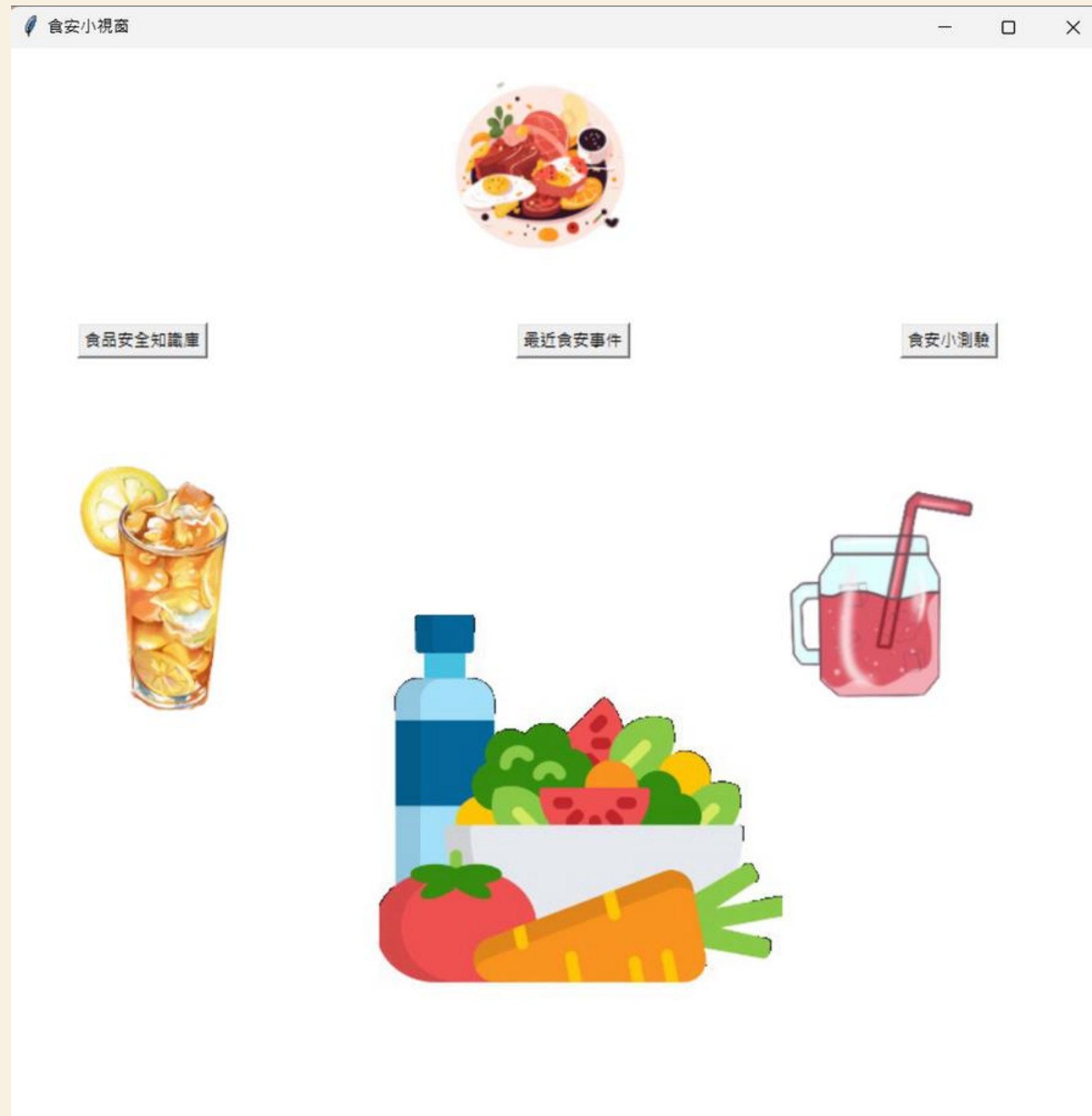
# 引入的函式庫

- **beautifulsoup4**
- **tkinter**



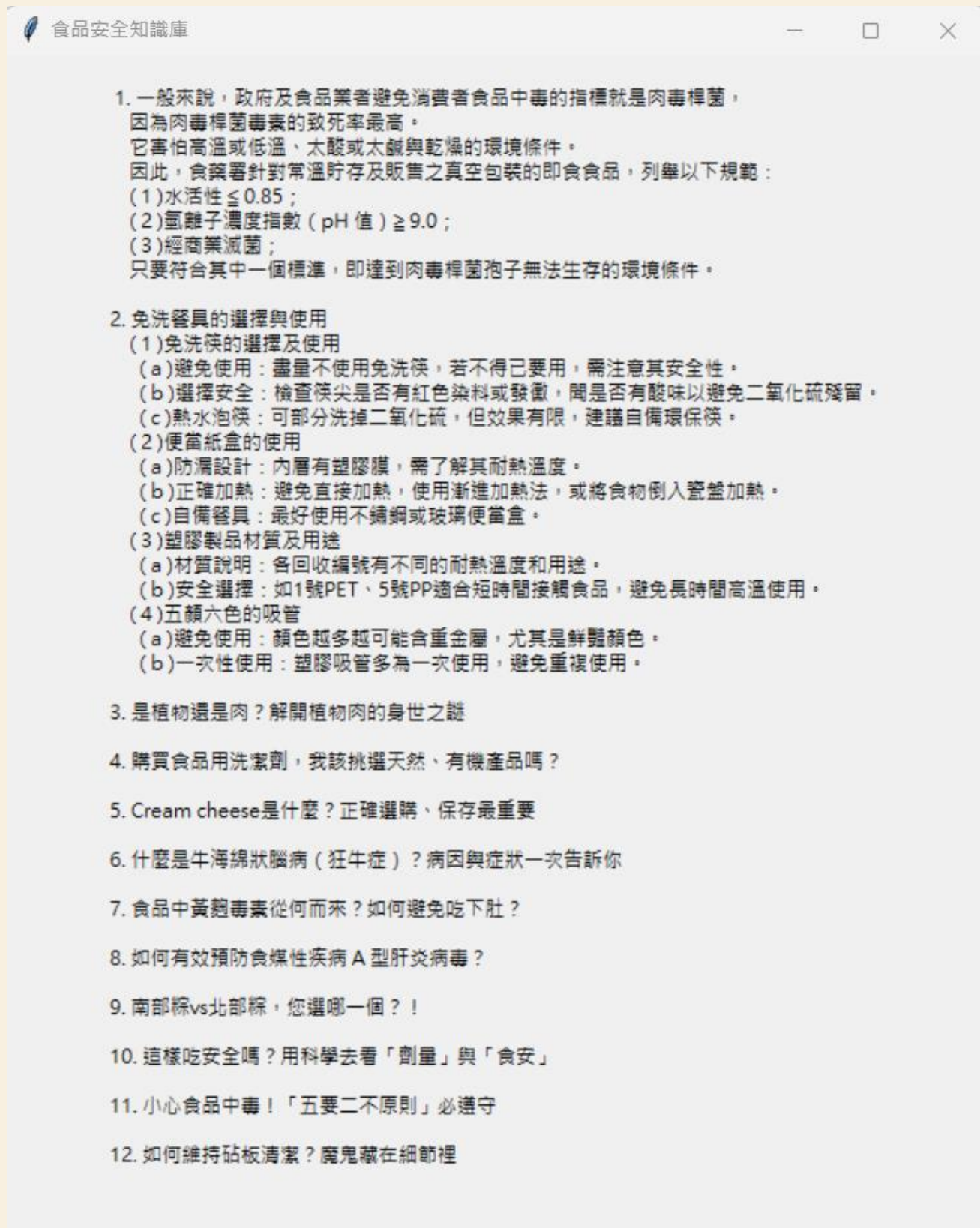


# 視窗介面(主畫面)



主畫面大小設定為800X800，包含三個按鍵，分別是「食品安全知識庫」、「最近食安事件」還有「食安小測驗」，按下這三個按鍵會跳出各自對應的小視窗。

# 視窗介面(食品安全知識庫)



「食品安全知識庫」內容的前2點是自己打上去的，從第3點之後是用爬蟲從「食藥好文網」抓下來的。

網址: <https://article->

[consumer.fda.gov.tw/subject\\_list.aspx?pid=1](https://article-consumer.fda.gov.tw/subject_list.aspx?pid=1)




# 視窗介面(最近食安事件)



「最近食安事件」的內容是從  
「風傳媒/食安」抓下來的。

網址:<https://www.storm.mg/category/k7762>

# 視窗介面(食安小遊戲)

 食安小遊戲


1.以下何者不是常溫之真空包裝的即食食品規範？

☐ A.氫離子濃度指數 ( pH 值 )  $\geq 9.0$

☐ B.經商業滅菌

☐ C.水活性  $\geq 0.85$

提交答案

 食安...


3.以下何者不是食品保存的正確方法？

☐ A.冷藏

☐ B.曝曬

☐ C.密封保存

提交答案

 食...


2.以下何者是食品中毒最高的指標？

☐ A.沙門氏菌

☐ B.大腸桿菌

☐ C.肉毒桿菌

提交答案

 食安小遊戲

4.以下何者是正確塑膠製品的材質及用途？

☐ A.1號PET可長時間高溫使用

☐ B.便當紙盒可以直接加熱

☐ C.熱水泡筷是為了洗掉二氧化硫

提交答案

總共有四題問題，按下「食安小遊戲」的按鍵，其中一題的小視窗會隨機出現。



# 視窗介面(食安小遊戲)

食... — □ ×

2.以下何者是食品中毒最高的指標？

☒ A.沙門氏菌


☐ B.大腸桿菌

☐ C.肉毒桿菌

提交答案

回答正確


回答結果 ×

 答對了！正確答案是 C.

確定

回答錯誤

回答結果 ×

 答錯了，正確答案是 C.

確定

# 程式碼(主畫面)

```
root = tk.Tk()  
root.geometry('800x800')  
root.title('Food Safety')
```

```
button1 = tk.Button(root, text="食品安全知識庫", command=open_new_window1)  
button1.place(x=50, y=200)
```

```
button2 = tk.Button(root, text="最近食安事件", command=open_new_window2)  
button2.place(x=370, y=200)
```

```
button3 = tk.Button(root, text="食安小測驗", command=open_game_window3)  
button3.place(x=650, y=200)
```

3個按鍵的設定



# 程式碼(主畫面)

```
# 載入第一張圖片
img = Image.open("Food.jpg")
img = ImageTk.PhotoImage(img)
label = tk.Label(root, image=img, width=290, height=290)
label.image = img # 保持對圖片的引用，避免被垃圾回收
label.place(x=270, y=400)

# 載入第二張圖片
img2 = Image.open("Diet.png")
img2 = ImageTk.PhotoImage(img2)
label2 = tk.Label(root, image=img2, width=170, height=170)
label2["image"] = img2
label2.place(x=300, y=0)
```

```
# 載入第三張圖片
img3 = Image.open("Drink.png")
img3 = ImageTk.PhotoImage(img3)
label3 = tk.Label(root, image=img3, width=120, height=180)
label3["image"] = img3
label3.place(x=50, y=300)

# 載入第四張圖片
img4 = Image.open("Drink2.png")
img4 = ImageTk.PhotoImage(img4)
label4 = tk.Label(root, image=img4, width=170, height=170)
label4["image"] = img4
label4.place(x=550, y=310)

root.configure(bg="white")
root.mainloop()
```

載入四張圖片並調整圖片位置

調整主畫面背景顏色



# 程式碼(食品安全知識庫)

## 「知識庫」的程式碼:

```
def open_new_window1():  
    new_window = tk.Toplevel(root)  
    new_window.geometry('600x800')  
    new_window.title("食品安全知識庫")  
    url = "https://article-consumer.fda.gov.tw/subject_list.aspx?pid=1"  
    text = text_add + getData1(url)
```

前2點

```
    label = tk.Label(new_window, text=text, wraplength=500, justify="left")  
    label.pack()
```

第3點之後call `getData1()`，用爬蟲去爬資料

```
button1 = tk.Button(root, text="食品安全知識庫", command=open_new_window1)  
button1.place(x=50, y=200)
```

# 程式碼(食品安全知識庫)

## 前2點的程式碼:

```
text_add = "\n 1. 一般來說，政府及食品業者避免消費者食品中毒的指標就是肉毒桿菌，\n\
因為肉毒桿菌毒素的致死率最高。 \n\
它害怕高溫或低溫、太酸或太鹹與乾燥的環境條件。 \n\
因此，食藥署針對常溫貯存及販售之真空包裝的即食食品，列舉以下規範： \n\
( 1 )水活性  $\leq 0.85$ ； \n      ( 2 )氫離子濃度指數 ( pH 值 )  $\geq 9.0$ ； \n      ( 3 )經商業滅菌； \n\
只要符合其中一個標準，即達到肉毒桿菌孢子無法生存的環境條件。 \n\
\n2. 免洗餐具的選擇與使用 \n\
( 1 )免洗筷的選擇及使用 \n\
    ( a )避免使用：盡量不使用免洗筷，若不得已要用，需注意其安全性。 \n\
    ( b )選擇安全：檢查筷尖是否有紅色染料或發黴，聞是否有酸味以避免二氧化硫殘留。 \n\
    ( c )熱水泡筷：可部分洗掉二氧化硫，但效果有限，建議自備環保筷。 \n\
( 2 )便當紙盒的使用 \n\
    ( a )防漏設計：內層有塑膠膜，需了解其耐熱溫度。 \n\
    ( b )正確加熱：避免直接加熱，使用漸進加熱法，或將食物倒入瓷盤加熱。 \n\
    ( c )自備餐具：最好使用不鏽鋼或玻璃便當盒。 \n\
( 3 )塑膠製品材質及用途 \n\
    ( a )材質說明：各回收編號有不同的耐熱溫度和用途。 \n\
    ( b )安全選擇：如1號PET、5號PP適合短時間接觸食品，避免長時間高溫使用。 \n\
( 4 )五顏六色的吸管 \n\
    ( a )避免使用：顏色越多越可能含重金屬，尤其是鮮豔顏色。 \n\
    ( b )一次性使用：塑膠吸管多為一次使用，避免重複使用。 \n"
```

# 程式碼(食品安全知識庫)

## 第3點之後的程式碼:

```
def getData1(url):  
    #抓取網頁原始碼(HTML)  
    import urllib.request as req  
    #建立一個Request物件，附加Request Headers的資訊  
    request = req.Request(url,headers = {  
        "cookie":"over18=1",  
        #讓網頁擁有者以為是正常使用者連線進入，而非電腦  
        "User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)  
    })  
    with req.urlopen(request) as response:  
        data = response.read().decode("utf-8")  
    #解析原始碼，取得每篇文章的標題  
    import bs4  
    root = bs4.BeautifulSoup(data,"html.parser")#讀BeautifulSoup 協助我們解析HTML 格式文件  
    titles = root.find_all("div",class_="subtitle")#尋找所有 class="subtitle"的div標籤  
    links = root.find_all("a",target="")  
    count = 3  
    text = "\n"  
    for title in titles:  
        if title.a is not None: # 如果包含a標籤(沒有被刪除).印出來  
            text += str(count)+". "+title.a.string + "\n"+ "\n" # 將每篇文章標題加入文字中  
            count+=1  
    # for link in links:  
    #     href = link.get("href")  
    #     if href:  
    #         text += href + "\n" # 將連結加入文字中  
  
    return text|
```



# 爬蟲(食品安全知識庫)

「食藥好文網」的網頁原始碼：

```
<div class="subtitle"><a href="subject.aspx?subjectid=1&id=3702" target="">是植物還是肉？解開植物肉的身世之謎</a></div><button class="dropdown" aria-expanded="false"><i class="fa fa-share-alt" aria-hidden="true"></i></button><ul class="dropdown-menu" aria-expanded="false"><i class="fa fa-share-alt" aria-hidden="true"></i></button><ul class="dropdown-menu" aria-expanded="false"><i class="fa fa-share-alt" aria-hidden="true"></i></button><ul class="dropdown-menu" aria-label="share"></ul></div>
```

class="subtitle" 的 div 標籤

# 程式碼(最近食安事件)

「最近食安事件」 的程式碼:

```
def open_new_window2():  
    new_window = tk.Toplevel(root)  
    new_window.geometry('700x700')  
    new_window.title("最近食安事件")  
    url = "https://www.storm.mg/category/k7762"  
    text1 = getData2(url)  
    label = tk.Label(new_window, text=text1, wraplength=600, justify="left")  
    label.pack()  
  
button2 = tk.Button(root, text="最近食安事件", command=open_new_window2)  
button2.place(x=370, y=200)
```

call getData2()



# 程式碼(最近食安事件)

「最近食安事件」 的程式碼(用爬蟲去抓資料):

```
def getData2(url):  
    #抓取網頁原始碼(HTML)  
    import urllib.request as req  
    #建立一個Request物件，附加Request Headers的資訊  
    request = req.Request(url,headers = {  
        "cookie":"over18=1",  
        #讓網頁擁有者以為是正常使用者連線進入，而非電腦  
        "User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)  
    })  
    with req.urlopen(request) as response:  
        data = response.read().decode("utf-8")  
    #解析原始碼，取得每篇文章的標題  
    import bs4  
    root = bs4.BeautifulSoup(data,"html.parser")#讀BeautifulSoup 協助我們解析HTML 格式文件  
    titles = root.find_all("h3", class_="card_title")  
    count = 1  
    text = "\n"  
    for t in titles:  
        if t is not None: # 如果包含a標籤(沒有被刪除).印出來  
            text += str(count)+". "+t.string + "\n"+"  
            count+=1  
    return text
```

尋找所有class="card\_title" 的 h3 標籤



# 爬蟲(最近食安事件)

## 「風傳媒/食安」的網頁原始碼:

```
<a href="https://www.storm.mg/lifestyle/5141906" class="card_link link_title">  
    <h3 class="card_title">知名礦泉水「驗出錳+細菌」急召回190萬瓶！電商平台龍頭中鏢 攝取過多恐神經病變</h3>  
    </a>  
<p class="card_info right">  
    <a href="/authors/385092/%E4%BE%AF%E8%A9%A0%E6%99%B4" class="card_link link_author" data-google-interstitial="false">  
        <span class="info_author">侯詠晴</span>  
    </a>  
    <span class="info_time">2024-05-31 14:58</span>  
</p>  
<a class="card_substance" href="https://www.storm.mg/lifestyle/5141906">  
    知名礦泉水品牌「斐濟水」(Fiji Natural Artesian Water) 先前針對產品進行檢測，卻驗出錳與3種細菌，今(2024)...
```

class="card\_title" 的 h3 標籤

# 程式碼(食安小遊戲)

## 「食安小遊戲」的程式碼:

```
class MCQGame:
    def __init__(self, master):
        self.master = master
        self.master.title("食安小遊戲")

    # 題目列表
    questions = [
        {
            "question": "1.以下何者不是常溫之真空包裝的即食食品規範?",
            "options": ["A.氫離子濃度指數 (pH 值)  $\geq 9.0$ ", "B.經商業滅菌", "C.水活性  $\geq 0.85$ "],
            "correct_answer": "C"
        },
        {
            "question": "2.以下何者是食品中毒最高的指標?",
            "options": ["A.沙門氏菌", "B.大腸桿菌", "C.肉毒桿菌"],
            "correct_answer": "C"
        },
        {
            "question": "3.以下何者不是食品保存的正確方法?",
            "options": ["A.冷藏", "B.曝曬", "C.密封保存"],
            "correct_answer": "B"
        },
        {
            "question": "4.以下何者是正確塑膠製品的材質及用途?",
            "options": ["A.1號PET可長時間高溫使用", "B.便當紙盒可以直接加熱", "C.熱水泡筷是為了洗掉二氧化硫"],
            "correct_answer": "C"
        }
    ]
```

在questions存入所有題目



# 程式碼(食安小遊戲)

## 「食安小遊戲」的程式碼:

從questions隨機產生一題



```
# 隨機選擇一道題目
self.current_question = random.choice(questions)

self.question_label = tk.Label(master, text=self.current_question["question"])
self.question_label.pack()

self.selected_option = tk.StringVar()

# 添加選項
for option in self.current_question["options"]:
    option_radio = tk.Radiobutton(master, text=option, variable=self.selected_option, value=option.split(".")[0], anchor="w")
    option_radio.pack(anchor="w")

self.submit_button = tk.Button(master, text="提交答案", command=self.check_answer)
self.submit_button.pack()
```



# 程式碼(食安小遊戲)

## 檢查答案是否正確:

```
def check_answer(self):  
    user_answer = self.selected_option.get()  
  
    if user_answer == self.current_question["correct_answer"]:  
        messagebox.showinfo("回答結果", "答對了！正確答案是 {}".format(self.current_question["correct_answer"]))  
    else:  
        messagebox.showerror("回答結果", "答錯了，正確答案是 {}".format(self.current_question["correct_answer"]))
```

## 食安小遊戲的主程式(呼叫MCQGame()):

```
def open_game_window3():  
    game_window = tk.Toplevel(root)  
    game = MCQGame(game_window)
```

# 未來改進

**1.在「食品安全知識庫」的爬蟲部分，原本的設定是打出每一點的Title，每個Title底下提供連結，讓使用者可以按下連結去細讀食品安全的知識，但是實際去爬後發現會爬出一些亂碼，連結也沒辦法直接讓使用者在視窗上直接點選，這個部分仍待改善。**

# 未來改進

```
def getData(url):
    #抓取網頁原始碼(HTML)
    import urllib.request as req
    #建立一個Request物件，附加Request Headers的資訊
    request = req.Request(url,headers = {
        "cookie":"over18=1",
        #讓網頁擁有者以為是正常使用者連線進入，而非電腦
        "User-Agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.0.0 Safari/537.36"
    })
    with req.urlopen(request) as response:
        data = response.read().decode("utf-8")
    #解析原始碼，取得每篇文章的標題
    import bs4
    root = bs4.BeautifulSoup(data,"html.parser")#讀BeautifulSoup 協助我們解析HTML 格式文件
    titles = root.find_all("div",class_="subtitle")#尋找所有 class="subtitle"的div標籤
    links = root.find_all("a",target_="")
    count = 1
    text = "\n"
    # for title in titles:
    #     if title.a is not None: # 如果包含a標籤(沒有被刪除).印出來
    #         text += str(count)+". "+title.a.string + "\n"+" \n" # 將每篇文章標題加入文字中
    #         count+=1
    for link in links:
        href = link.get("href")
        if href:
            print(href)

pgURL = "https://article-consumer.fda.gov.tw/subject_list.aspx?pid=1"
getData(pgURL)
```

有試圖去抓連結，但是會抓到一些亂碼(下一頁)

# 未來改進

subject\_list.aspx?pid=1  
subject\_list.aspx?pid=1&parentid=0  
subject.aspx?subjectid=1&id=3702  
subject.aspx?subjectid=1&id=3702  
https://www.facebook.com/sharer.php?u=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3702  
https://plus.google.com/share?url=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3702  
http://line.naver.jp/R/msg/text/?是植物還是肉？解開植物肉的身世之謎 https://article-consumer.fda.gov.tw/subject.aspx?subjectid=1&id=3702  
subject.aspx?subjectid=1&id=3699  
subject.aspx?subjectid=1&id=3699  
https://www.facebook.com/sharer.php?u=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3699  
https://plus.google.com/share?url=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3699  
http://line.naver.jp/R/msg/text/?購買食品用洗潔劑，我該挑選天然、有機產品嗎？ https://article-consumer.fda.gov.tw/subject.aspx?subjectid=1&id=3699  
subject.aspx?subjectid=1&id=3697  
subject.aspx?subjectid=1&id=3697  
https://www.facebook.com/sharer.php?u=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3697  
https://plus.google.com/share?url=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3697  
http://line.naver.jp/R/msg/text/?Cream cheese是什麼？正確選購、保存最重要 https://article-consumer.fda.gov.tw/subject.aspx?subjectid=1&id=3697  
subject.aspx?subjectid=1&id=3687  
subject.aspx?subjectid=1&id=3687  
https://www.facebook.com/sharer.php?u=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3687  
https://plus.google.com/share?url=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3687  
http://line.naver.jp/R/msg/text/?什麼是牛海綿狀腦病（狂牛症）？病因與症狀一次告訴你 https://article-consumer.fda.gov.tw/subject.aspx?subjectid=1&id=3687  
subject.aspx?subjectid=1&id=3686  
subject.aspx?subjectid=1&id=3686  
https://www.facebook.com/sharer.php?u=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3686  
https://plus.google.com/share?url=https%3a%2f%2farticle-consumer.fda.gov.tw%2fsubject.aspx%3fsubjectid%3d1%26id%3d3686  
http://line.naver.jp/R/msg/text/?食品中黃麴毒素從何而來？如何避免吃下肚？ https://article-consumer.fda.gov.tw/subject.aspx?subjectid=1&id=3686

亂碼

正確的連結



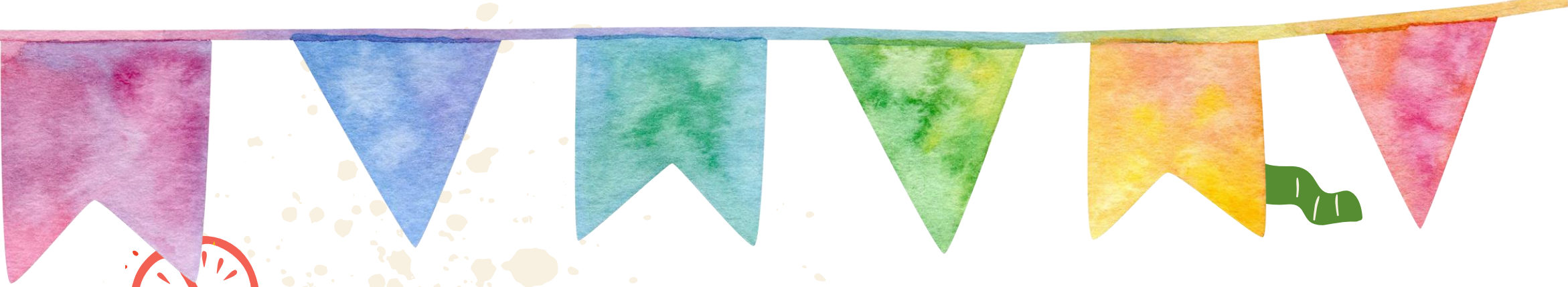
# 未來改進

**2.在「食安小遊戲」的部分，目前是以人工出題，再隨機從題庫選擇一題題目，未來希望可以做出一套生成題目的系統，依據爬蟲爬到的食安知識由系統自行出題，這樣就不需要透過人工出題，可以進而省下出題的時間與精力。**

# 收穫與心得

這次final project的題目是在學期中間，食安事件爆發時就定下來了。跟之前在系上修課時的final project不一樣的點在於，這次我的隊友不是人類，而是強大的classroomGPT，在整個做專題的過程中，他給予我很大的幫助，基本上只要指令有下對，他都能生成正確的程式碼給我，而這也讓我看見了生成式AI的強大。

未來可預見的是一個無論做甚麼都會碰上AI的時代，而身為本科系的同學，要如何靈活應用，而不成為人們口中的「碼農」，我想這是我必須嚴肅去思考的問題，也謝謝老師給我一個機會，讓我好好的利用生成式AI去完成一份期末專題，未來也希望能有機會一窺生成式AI背後的運作原理，知己知彼，讓AI能夠更好的成為我生活和工作中的好幫手。



# THANKS

*Feel-good food*

