第1次作業-作業-HW1

> 學號：112111225
> 姓名：林芷羽
> 作業撰寫時間：180 (mins，包含程式撰寫時間)
> 最後撰寫文件日期：2024/10/01

本份文件包含以下主題：(至少需下面兩項，若是有多者可以自行新增)

- ☑ 說明內容
- ☑ 個人認為完成作業須具備觀念

說明程式與內容

開始寫說明，該說明需說明想法， 並於之後再對上述想法的每一部分將程式進一步進行展現， 若需引用程式區則使用下面方法， 若為.cs檔內程式除了於敘述中需註明檔案名稱外， 還需使用語法```語言種類 程式碼 ```，其中語言種類若是要用python則使用py，java則使用java，C/C++則使用cpp， 下段程式碼為語言種類選擇csharp使用後結果：

```
public void mt_getResult(){
    ...
}
```

若要於內文中標示部分網頁檔，則使用以下標籤```html 程式碼 ```， 下段程式碼則為使用後結果：

```
<%@ Page Language="C#" AutoEventWireup="true" ...>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" ...>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
        </div>
    </form>
</body>
</html>
```

更多markdown方法可參閱https://ithelp.ithome.com.tw/articles/10203758

請在撰寫"說明程式與內容"該塊內容，請把原該塊內上述敘述刪除，該塊上述內容只是用來指引該怎麼撰寫內容。

1. 請解釋何謂git中下列指令代表什麼？並舉個例子，同時必須說明該例子的結果。其指令有add、commit、push、fetch、pull、branch、checkout與merge。

Ans:

```
add：將變動（新增、修改、刪除）準備好，加入暫存區
範例：git add a.txt
結果：a.txt 的變更已加入暫存區

commit：將暫存區中的變動提交至本地倉庫，創建新的提交紀錄
範例：git commit
結果：將暫存區中的 a.txt 變更提交到版本庫，創建一個新的提交

push：將本地分支的變動推送到遠端倉庫
範例：git push
結果：更新遠端倉庫，推送所有提交

fetch：從遠端倉庫獲取最新的提交，但不改變當前的工作分支
範例：git fetch
結果：下載最新的提交，但不改變當前的工作分支。可以檢查遠端分支的變更

pull：從遠端倉庫獲取最新的提交並合併到當前工作分支
範例：git pull
結果：將分支的最新提交下載並合併到當前的工作分支

branch：創建一個新的分支
範例：git branch new
結果：創建一個名為 new 的新分支

checkout：切換到指定的分支
範例：git checkout new
結果：當前工作分支將切換到 new

merge：將兩個或多個分支的變更整合到一起
範例：git merge new
結果：將 new 分支的變更合併到當前工作分支
```

2. 於專案下的檔案—**hw1.py**，撰寫註解，以說明該程式每列中之背後意義。

   該hw1.py題目如下：

   ```
   統計字母數。假設今天輸入一句子，句子中有許多單字，單字皆為英文字母小寫，
   請統計句子中字母出現的字數，輸出實需要照字母排序輸出，且若該字母為0則不輸出

   如輸入
   this is an apple
   輸出
   a: 2
   e: 1
   h: 1
   ```

```
i: 2
l: 1
n: 1
p: 2
s: 2
t: 1
```

Ans:

```python
# 引入List類型以便使用
from typing import List

def countLetters(sentence: str) -> List[int]:
    # 創建一個長度為26的列表，對應英文字母a-z，用於儲存每個字母的出現次數
    letterCount: List[int] = [0] * 26

# 檢查輸入的句子中的每個字符
    for char in sentence:
        # 檢查字符是否為英文字母
        if char.isalpha():
            # 計算其在字母表中的索引
            index = ord(char) - ord('a')
            letterCount[index] += 1  # 對應字母的計數加一

    return letterCount # 返回包含每個字母計數的列表
pass

#定義一個函數來列印字母計數，列表為參數
def printLetterCount(letterCount: List[int]) -> None:

# 檢查字母計數列表，總共26個字母
    for i in range(26):
        # 若該字母的計數大於0，則輸出該字母及其出現的次數
        if letterCount[i] > 0:
            print(f"{chr(i + ord('a'))}: {letterCount[i]}") # 輸出字母出現次數
pass

inputSentence: str = "this is an apple" # 定義句子
letterCount: List[int] = countLetters(inputSentence) # 計算字母出現次數
printLetterCount(letterCount) # 輸出各字母的計數結果
```

3. 請新增檔案**hw1_2.py，**輸入一個正整數(N)，其中$1\le N \le 100000$，請將該正整數輸出進行反轉

```
如輸入
1081

輸出
1801
```

```
如輸入
1000

輸出
1
```

Ans:

```python
def reverse_integer(n: int) -> int:
    reversed_n = 0
    while n > 0:
        digit = n % 10
        reversed_n = reversed_n * 10 + digit
        n //= 10
    return reversed_n

if __name__ == "__main__":

    input_number = int(input("請輸入一個正整數: "))


    if input_number > 0:

        output_number = reverse_integer(input_number)
        print(output_number)
    else:
        print("請確保輸入的是正整數。")
```

4. **[課外題]**：請找尋資料，說明何謂**單元測試**，請新增檔案**hw1_3.py**，並利用溫度計攝氏轉華氏撰寫單元測試。

Ans:

單元測試：單元測試（Unit Testing）是一種軟體測試方法，旨在驗證程式碼中的最小可測試單元（通常是函數或方法）的正確性。
單元測試通常由開發者在開發過程中編寫，以確保每個單元都能按照預期工作。這樣可以在早期階段及時發現和修正錯誤，提高程式碼的可靠性和可維護性。

單元測試的特點包括：

自動化：單元測試通常可以自動執行，開發者可以快速檢查多個測試案例。
獨立性：每個測試案例應獨立於其他測試，不應影響或依賴其他測試的結果。
可重複性：每次執行測試時，應該得到相同的結果，無論何時執行。
快速性：單元測試通常執行速度快，便於開發者在編碼過程中經常運行。

```python
def celsius_to_fahrenheit(celsius: float) -> float:
    """將攝氏溫度轉換為華氏溫度"""
    return (celsius * 9/5) + 32

if __name__ == "__main__":
    # 測試轉換
    celsius_temp = float(input("請輸入攝氏溫度: "))
    fahrenheit_temp = celsius_to_fahrenheit(celsius_temp)
    print(f"華氏溫度: {fahrenheit_temp}")
```

```python
import unittest
from hw1_3 import celsius_to_fahrenheit

class TestTemperatureConversion(unittest.TestCase):

    def test_positive_celsius(self):
        self.assertEqual(celsius_to_fahrenheit(0), 32)  # 0°C = 32°F
        self.assertEqual(celsius_to_fahrenheit(100), 212)  # 100°C = 212°F

    def test_negative_celsius(self):
        self.assertEqual(celsius_to_fahrenheit(-40), -40)  # -40°C = -40°F
        self.assertEqual(celsius_to_fahrenheit(-20), -4)  # -20°C = -4°F

    def test_non_integer_celsius(self):
        self.assertAlmostEqual(celsius_to_fahrenheit(25.5), 77.9, places=1)  #
25.5°C = 77.9°F

if __name__ == "__main__":
    unittest.main()
```

個人認為完成作業須具備觀念

開始寫說明，需要說明本次練習需學會那些觀念 (需寫成文章，需最少50字，並且文內不得有你、我、他三種文字)且必須提供完整與練習相關過程的notion筆記連結

> 完成此作業需要理解基本的程式設計邏輯，對於 `Git` 的基本指令需有清楚的認識，以便有效管理版本與協作。
> 在撰寫程式時，必須具備良好的習慣，以提升程式碼的可讀性和可維護性。了解單元測試的概念，能有效幫助及早發現錯誤，確保程式的穩定性與可靠性。