第2次練習-練習-PC2

> 學號：1234567
> 姓名：李佩琪
> 作業撰寫時間：90 (mins，包含程式撰寫時間)
> 最後撰寫文件日期：2024/1/

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. 請參照Topic2投影片p. 39，...(見題目.pdf)

Ans:

```
第一題
# 定義多項式的係數 (按次數從高到低排列)
coefficients = [6, 0, 2, 0, 3]  # 6x^4 + 0x^3 + 2x^2 + 0x + 3

def evaluate_polynomial(coeffs, x):
    """計算多項式的值"""
    return sum(c * (x ** i) for i, c in enumerate(reversed(coeffs)))

# 計算 x = 91 時的值
x = 91
result = evaluate_polynomial(coefficients, x)

# 顯示結果
print(f"f({x}) = {result}")

第二題
class Polynomial:
    """多項式 (如 f(x) = 6x^4 + 2x^2 + 3)"""
    def __init__(self, coefficients):
        """
        初始化多項式
        :param coefficients: 按次數從高到低排列的係數列表
        """
        self.coefficients = coefficients

    def evaluate(self, x):
        """計算多項式的值"""
        return sum(c * (x ** i) for i, c in
enumerate(reversed(self.coefficients)))

    def __str__(self):
        """返回多項式的字串表示形式"""
        terms = []
        degree = len(self.coefficients) - 1
        for i, c in enumerate(self.coefficients):
            if c == 0:
                continue
            power = degree - i
            if power == 0:
                terms.append(f"{c}")
            elif power == 1:
                terms.append(f"{c}x")
            else:
                terms.append(f"{c}x^{power}")
        return " + ".join(terms)

# 建立多項式 f(x) = 6x^4 + 2x^2 + 3
f = Polynomial([6, 0, 2, 0, 3])
```

```
# 顯示多項式
print(f"Polynomial: {f}")

# 計算 x = 91 時的值
x = 91
result = f.evaluate(x)

# 顯示結果
print(f"f({x}) = {result}")
```

2. 承1，請使用物件導向方式實作上題，也就是每個單位的x次方做成一個類別後，完成上述儲存功能，並算其結果

Ans:

```
# 初始化矩陣
def create_matrix(rows, cols, initial_value=0):
    return [[initial_value for _ in range(cols)] for _ in range(rows)]

# 定義函式，將值存入矩陣的指定位置
def gray(array, i, j, value):
    if 0 <= i < len(array) and 0 <= j < len(array[0]):  # 檢查索引是否在範圍內
        array[i][j] = value
    else:
        print(f"索引 ({i}, {j}) 超出範圍！")

# 定義顯示函式
def print_matrix(matrix):
    for row in matrix:
        print(" ".join(f"{val:3}" for val in row))

# 測試
rows, cols = 5, 5
image = create_matrix(rows, cols)

gray(image, 0, 1, 50)
gray(image, 1, 3, 120)
gray(image, 2, 4, 180)
gray(image, 3, 2, 255)

# 顯示結果
print("稀疏矩陣存儲結果：")
print_matrix(image)
```

3. 在數位圖像處理中，大多數的高分辨率灰度圖像中，很多像素的值為 0，這使得 圖像可以用稀疏矩陣來有效地表示與壓縮。假設你有一個 5x5 的灰度圖像，其中 大多數像素值為 0，只有少數像素有非零值。

Ans:

```python
def count_inversions(arr):
    """計算逆序對數量，使用歸併排序法"""
    def merge_sort_and_count(array, temp_array, left, right):
        inv_count = 0
        if left < right:
            mid = (left + right) // 2

            inv_count += merge_sort_and_count(array, temp_array, left, mid)
            inv_count += merge_sort_and_count(array, temp_array, mid + 1, right)
            inv_count += merge_and_count(array, temp_array, left, mid, right)

        return inv_count

    def merge_and_count(array, temp_array, left, mid, right):
        i = left       # 左子陣列起點
        j = mid + 1    # 右子陣列起點
        k = left       # 暫存陣列起點
        inv_count = 0  # 逆序對計數

        while i <= mid and j <= right:
            if array[i] <= array[j]:
                temp_array[k] = array[i]
                i += 1
            else:
                temp_array[k] = array[j]
                inv_count += (mid - i + 1)  # 計算逆序對
                j += 1
            k += 1

        # 複製剩餘元素
        while i <= mid:
            temp_array[k] = array[i]
            i += 1
            k += 1

        while j <= right:
            temp_array[k] = array[j]
            j += 1
            k += 1

        # 將排序後的子陣列複製回原陣列
        for i in range(left, right + 1):
            array[i] = temp_array[i]

        return inv_count

    n = len(arr)
    temp_array = [0] * n
    return merge_sort_and_count(arr, temp_array, 0, n - 1)


# 讀取輸入
n = int(input("請輸入陣列元素個數："))
```

```
array = list(map(int, input("請輸入陣列元素：").split()))

# 計算逆序對數量
result = count_inversions(array)

# 輸出結果
print(f"Output：{result}")
```

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 本次練習的核心在於學習陣列中逆序對的計算。透過題目的解決過程，應掌握暴力解法與歸併排序兩種方法的差異與應用。需理解基本程式設計技巧，例如雙層迴圈的遍歷方式，以及遞迴與分治法在優化計算中的重要性。此外，還要熟悉如何將數學問題轉化為程式碼，並有效率地解決問題。