第2次練習-練習-PC2

學號：112111207
姓名：陳品霖
作業撰寫時間：60 (mins，包含程式撰寫時間)
最後撰寫文件日期：2024/12/26

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

1.

Ans:

```
##第1題
# 定義多項式 f(x) = 6x^4 + 2x^2 + 3
def f(x):
    return 6 * (x**4) + 2 * (x**2) + 3

# 儲存結果的字典格式
polynomial = {
    "coefficients": [4, 6, 0, 2, 3,0],  # 係數對應 [6x^4, 0x^3, 2x^2, 0x, 3]
    "degree": 4                         # 多項式的最高次數
}

# 計算 x = 91 時的值
x = 91
result = f(x)

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

程式碼說明：

$f(x)$

6 x 4 + 2 x 2 + 3 f(x)=6x 4 +2x 2 +3，用字典儲存其結構，並計算 f (91) f(91)，最後輸出結果。

2.

Ans:

```
    ##第2題
class Term:
    """多項式的一個單項 (如 6x^4)"""
    def __init__(self, coefficient, exponent):
        self.coefficient = coefficient  # 係數
        self.exponent = exponent        # 次方

    def evaluate(self, x):
        """計算單項的值"""
        return self.coefficient * (x ** self.exponent)


class Polynomial:
    """多項式 (如 f(x) = 6x^4 + 2x^2 + 3)"""
    def __init__(self):
        self.terms = []  # 儲存多項式中的所有單項

    def add_term(self, coefficient, exponent):
        """新增一個單項"""
        self.terms.append(Term(coefficient, exponent))

    def evaluate(self, x):
        """計算多項式的值"""
        return sum(term.evaluate(x) for term in self.terms)


# 建立多項式 f(x) = 6x^4 + 2x^2 + 3
f = Polynomial()
f.add_term(6, 4)  # 6x^4
f.add_term(2, 2)  # 2x^2
f.add_term(3, 0)  # 3 (常數項)

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

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

程式碼說明：

這段程式碼功能相同，使用類別結構來實現多項式運算，以下是簡要說明：

1. 新增了`Term`類別 表示多項式的一個單項（如 $6x^4$），包含係數和次方，並可計算單項的值。

2. `Polynomial` 類別：表示整個多項式，使用 `add_term` 方法新增單項，並用`evaluate` 方法計算多項式在 x 處的值。

3.

Ans:

```
# 初始化 5x5 矩陣
rows, cols = 5, 5
image = [[0 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！")

# 測試數據
gray(image, 0, 1, 50)
gray(image, 1, 3, 120)
gray(image, 2, 4, 180)
gray(image, 3, 2, 255)

# 輸出結果
print("稀疏矩陣存儲結果：")
for row in image:
    print(row)
```

程式碼說明：

此程式建立一個 5x5 的矩陣，用函式 `gray`將指定位置的值存入矩陣，並檢查索引範圍，最後輸出矩陣內容。

4.

Ans：

```
# 讀取輸入
n = int(input("請輸入陣列元素個數："))  # 第一行為陣列元素個數
array = list(map(int, input("請輸入陣列元素：").split()))  # 第二行為陣列元素

# 計算逆序對數量
count = 0
for i in range(n):
    for j in range(i + 1, n):
        if array[i] > array[j]:
```

```
                count += 1

    # 輸出結果
    print(f"Output：{count}")
```

程式碼說明：

此程式計算陣列中的逆序對數量即$i<j$i<j 且 $A[i]>A[j]$A[i]>A[j] 的對數），通過兩層迴圈遍歷陣列進行比較，最後輸出逆序對數量。

個人認為完成作業須具備觀念

1.條件判斷與邊界檢查：在操作矩陣或陣列時，應確認索引是否在合理範圍內，避免發生錯誤。

2.多重迴圈應用：熟練使用巢狀迴圈進行多層級的比較與處理，例如逆序對計算需要兩層迴圈來檢查陣列中元素之間的關係。

3.數據結構的設計與操作：包括使用二維陣列模擬矩陣以及適當儲存資料的方式，例如將多項式拆分成單項儲存。

4.函式的設計與重用：透過函式進行封裝，提高程式的可讀性與模組化設計。

5.基本演算法：理解和實現基礎的排序和比較邏輯，例如計算逆序對的暴力解法。