第2次作業-作業-HW2

> 學號：112111209
> 姓名：張香裕
> 作業撰寫時間：150 (mins，包含程式撰寫時間)
> 最後撰寫文件日期：2024/10/26

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

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. 問題如下圖所述，並回答下面問題。

Ans:

```
a.
def getResult():
    # 定義字符
    alphabet1 = [
        ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0'],
        ['Q', 'W', 'E', 'R', 'T', 'Y', 'U', 'I', 'O', 'P'],
        ['A', 'S', 'D', 'F', 'G', 'H', 'J', 'K', 'L', ';'],
        ['Z', 'X', 'C', 'V', 'B', 'N', 'M', ',', '.', '/']
    ]

b.
```

---

2. 給定一個包含 n 個不同數字的數組，這些數字的範圍是從 0 到 n。找出數組中缺失的那一個數字。

Ans:

```
def missingNumber(nums):
    # 計算數組的長度 n
    n = len(nums)

    # 計算 0 到 n 的總和
    c_sum = n * (n + 1) // 2

    # 計算數組中的總和
    now_sum = sum(nums)

    # 缺失的數字就是總和實際總和
    missing_num = c_sum - now_sum

    # 返回缺失的數字
    return missing_num


nums1 = [3, 0, 1]
print(missingNumber(nums1))

nums2 = [9, 6, 4, 2, 3, 5, 7, 0, 1]
print(missingNumber(nums2))
```

---

3. 請回答下面問題：

Ans:

a.
$2^{N+1}=2x2^N$
討論時間複雜度時,會忽略常數係數,所以2可以忽略不計。
所以答案是: 會
b.
2^{2N}和2^N對比起來,2^{2N}比2^N快了許多,所以時間複雜度不可能為$O(2^N)$。
所以答案是: 不會

---

4. 請問以下各函式，在進行呼叫後，請計算(1)執行次數T(n)，並(2)透過執行次數判斷時間複雜度為何(請用Big-Oh進行表示)？

Ans:

a.

```
def calculateTimes (number: int) -> None:
    while number >= 1:              #n+1
        counter:int = number        #n
        while counter >= 1:         #(n+1+2)*n/2
            print(number, counter)  #(n+1)*n/2
            counter = counter - 1   #(n+1)*n/2
        number = number - 1         #n
```

$$ (1) T(n)= \frac{3}{2}n^2+\frac{11}{2}n+1 \qquad $$ $$ (2) T(n)=O(n^2) $$

b.

```
def calculateTimes (number: int) -> None:
    while number >= 1:              #floor(log_{2}n)+2
        print(number)               #floor(log_{2}n)+1
        number = number // 2        #floor(log_{2}n)+1
```

$$ (1) T(n)=3[(log_2n)]+4 \qquad $$ $$ (2) T(n)=O(log_2n) $$

c.

```
def calculateTimes (number: int, size: int) -> None:
    while number >= 1:              #floor(log_{2}n)+2
        while size >= 1:            #(m+1)(floor(log_{2}n)+1)
            print(number, size)     #m(floor(log_{2}n)+1)
```

```
            size = size - 1         #m(floor(log_{2}n)+1)
    number = number // 2            #floor(log_{2}n)
```

$$ (1) T(n,m)=(3m+3)[(log_2n)]+3m+4\qquad $$ $$ (2) T(n,m)=O(mlog_2n) $$

d.

```
def calculateTimes (number: int, size: int) -> None:
    while number >= 1:              #floor(log_{2}n)+2
        while size >= 1:            #(n+1)(floor(log_{2}n)+1)
            print(number, size)     #n(floor(log_{2}n)+1)
            size = size - 1         #n(floor(log_{2}n)+1)
    number = number // 2            #floor(log_{2}n)+1
```

```
def calculateTimes (number: int, size: int) -> None:
    while number >= 1:              #floor(log_{2}n)+2
        while size >= 1:            #(n/2+1)(floor(log_{2}n)+1)
            print(number, size)     #n/2(floor(log_{2}n)+1)
            size = size - 1         #n/2(floor(log_{2}n)+1)
    number = number // 2            #floor(log_{2n}n)+1
```

$$ (1) (3n+3)[(log_2n)]+3n+4\ge T(n) \ge (\frac{3n}{2}+3)[(log_2n)]+\frac{3n}{2}+4\qquad $$ $$ (2) T(n)=O(nlog_2n) $$

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

這次的作業需要具備:
1.基本陣列的相關觀念
2.並且對於時間複雜度的概念了解
3.了解程式的完整運行過程以便計算時間複雜度