第2次隨堂-隨堂-QZ2

> 學號：1234567
> 姓名：陳恩偉
> 作業撰寫時間：180 (mins，包含程式撰寫時間)
> 最後撰寫文件日期：2024/12/18

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

1. 請執行下面Git操作 (請參照題目pdf)

Ans:

2. 請整理課堂上所提的Git指令，顯示其語法語給予明顯的例子 GitHub Desktop Ans:

3. 請先寫出同學認為資料結構一門課每周需要花多少時間？並提出理由。並將「請問要學習大學的資料結構每一週需要對於一般大學生需要多少時間」貼入chatgpt，並將其所建議的敘述貼出後，比較同學所認為若是每周低於chatgpt所評估的時間，兩者主要差異為何？應該怎麼安排自己時間了解資料結構？

Ans: 每周10小時，大量概念及技術皆須手動實驗、驗證。 CHATGPT: 學習大學的資料結構課程，所需的時間因人而異，但可以提供一般的參考範圍：

每週學習時間分配：

1. **上課時間**
   通常資料結構是核心課程，每週大約有 **2-3小時** 的授課時間，包含理論講解和例題分析。

2. **自習時間**

   - 理解課堂內容：每週約 **2-3小時**，複習筆記、教科書或講義，消化新概念（如鏈結串列、堆疊、樹、圖等）。
   - 實作練習：每週約 **3-5小時**，用程式（如C++、Java、Python）實現課程中的資料結構和演算法，這部分非常重要。
   - 預習和準備：每週約 **1-2小時**，提前閱讀下一次課程的內容。

3. **額外學習（視需求而定）**

   - 如果有額外的作業、測驗準備或專題，可能需要再投入 **1-3小時**。
   - 如果不熟悉程式語言或演算法，可能需要更多時間補足基礎。

總時間估算：

一般情況下，建議每週花 **8-12小時**，包含課堂時間、自習和實作練習。如果課程進度特別緊湊，或需要準備考試，可能需要更高的投入。

影響因素：

1. **基礎程度**
    - 如果熟悉程式設計和演算法，學習速度會快些。
    - 如果是初學者，可能需要更多時間去理解基本概念。
2. **學習方式**
    - 使用視覺化工具（如VisuAlgo）或教學影片來輔助學習，能幫助快速理解。
    - 加入小組討論或學習社群，可以減少卡關時間。
3. **課程設計**
    - 有些課程設計偏理論（需要花更多時間理解概念），有些偏實作（需要更多時間寫程式練習）。

建議學習策略：

1. **重點掌握概念**：
    - 確保理解基本結構（陣列、鏈結串列、堆疊、佇列、樹、圖等）和它們的操作（插入、刪除、搜尋、遍歷）。
    - 了解每個結構適用的情境與效率分析（時間與空間複雜度）。
2. **動手實作**：
    - 理解是第一步，實作是核心，建議用一種主流語言（如Python、Java）重點實作學過的資料結構。
3. **持續練習與測試**：
    - 嘗試做一些演算法題目（如LeetCode、HackerRank），這些平台能強化你的應用能力。

時間差不多，做的事情也差不多但偏向閱讀文獻居多。

個人認為完成作業須具備觀念

本次的核心在於掌握基礎的資料結構設計與應用能力。需理解不同結構的特性與操作，包括但不限於陣列的索引與排序、連結串列的節點操作、堆疊與佇列的先進先出與後進先出特性，以及樹和圖的遍歷方式。演算法的時間與空間複雜度分析亦為不可忽略的部分，因為其直接影響程式的效能。動手實作能強化對概念的理解，同時提升問題解決能力與程式設計技巧。