0819823陳子祈 演算法導論 HW3

1. 試驗成果

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

根據基本要求1、2，每個input array 所有元素都是隨機產生的，並使用for迴圈設定了不同大小的input array，且大小從100逐次成長10倍到100000，根據實測結果如下:

1. Insertion sort

|  |  |
| --- | --- |
| Input size | Runtime(ms) |
| 100 | 0 |
| 1000 | 1 |
| 10000 | 147 |
| 100000 | 14223 |

就Insertion sort而言，理論上10倍的input size，執行時間約成長100倍。不過資料量不足，所以我來測測看輸入大小從1000~10000，每次輸入大小為前一次輸入大小加10000，程式碼改為如下:

一張含有 文字 的圖片

自動產生的描述

重製表格如下:

一張含有 文字, 電腦, 檔案 的圖片

自動產生的描述

由上表可知，執行時間與n^2有接近正比的關係，若n為input size，f(n)為此演算法執行的時間，符合Θ(f(n)) = n^2的趨勢。

1. Merge sort

|  |  |
| --- | --- |
| Input size | Runtime(ms) |
| 100 | 0 |
| 1000 | 0 |
| 10000 | 2 |
| 100000 | 23 |

就Merge sort而言，輸入大小小於10000的執行時間太短，資料量不足，所以我來測測看輸入大小從1000~10000，每次輸入大小為前一次輸入大小加10000，程式碼改為如下:

一張含有 文字 的圖片

自動產生的描述

重製表格如下:

一張含有 文字, 檔案, 電腦, 鍵盤 的圖片

自動產生的描述

由上表可知，執行時間與nlgn有接近正比的關係，若n為input size，f(n)為此演算法執行的時間，大致符合Θ(f(n)) = n\*lg(n)的趨勢。

1. Heap sort

|  |  |
| --- | --- |
| Input size | Runtime(ms) |
| 100 | 0 |
| 1000 | 0 |
| 10000 | 3 |
| 100000 | 40 |

就Heap sort而言，輸入大小小於10000的執行時間太短，資料量不足，所以我來測測看輸入大小從1000~10000，每次輸入大小為前一次輸入大小加10000，程式碼改為如下:

一張含有 文字 的圖片

自動產生的描述

重製表格如下:

一張含有 文字, 檔案, 電腦 的圖片

自動產生的描述

由上表可知，執行時間與nlgn有接近正比的關係，若n為input size，f(n)為此演算法執行的時間，大致也符合Θ(f(n)) = n\*lg(n)的趨勢。

1. 測試演算法正確性

1. 將第65行註解消除掉，可印出Input array

一張含有 文字 的圖片

自動產生的描述

2. 調整第233、245、257行成固定size，即可準備測試

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

3. 將第89、144、198行的PrintA()註解拿掉

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

一張含有 文字 的圖片

自動產生的描述

4. 編譯並執行程式

一張含有 文字 的圖片

自動產生的描述

1. 心得

此作業與第一次作業非常類似，只是要再加Heap sort進去第一次作業的程式碼而已。不過我覺得我第一次作業寫的方向有點不好，應該不要使用Python視覺化執行時間，而是印出來不同執行時間與Θ(f(n))的比值，這樣更能驗證執行時間的複雜度。根據我印出來的結果，不同執行時間與Θ(f(n))的比值變動幅度非常小，因此可以推測執行時間與Θ(f(n))成正比，也就是說執行時間與Θ(f(n))只差一個倍數，那執行時間複雜度就是Θ(f(n))了。

最近期中考將近，很多科都要開始準備了，我要對每個演算法更熟悉，且對各種時間複雜度也要很熟。希望期中順利!