Dsnp HW5 Report

* 報告撰寫人：R05943092曾育為
* 實驗設計
  + 設計Array、Dlist、Bst三者資料結構，進行比較，下表為三者實作方法

|  |  |  |  |
| --- | --- | --- | --- |
|  | Array | Double link list | Binary search tree |
| Insert | 分成三種情況：   1. 當\_capacity為空，建立一個\_capacity為1的array 2. 當\_capacity=\_size時，將\_capacity放大二倍，將原先的值放入 3. 當\_capacity<\_size   ，直接放入 | 分為兩種情況：   1. 當empty()時，直接取代插入\_head 2. 當不是empty()時，插入至\_head 前面的前面 | 會先原先\_root的左子樹建一個真實的\_root，而原先的＿root 就等於iterator end()結束的地方  分成二種情況：  與真實的＿root比較 1. 小者往左走   1. 大者往右走   以此遞迴，找到正確位置，插入此點 |
| erase | 用iterator指向預刪除的點，用array最後一個值補上，並將\_size-1 | 分成二種情況：   1. 刪除\_head，刪除後並將\_head指標指向下一個 2. 不是刪除\_head，刪除後將前後指標補上即可 | 我會先判斷預刪除的點是屬於其父點的左還是右子點  再分四種情況討論：  刪除的點   1. 沒有左右子點 2. 只有有左子點 3. 只有有右子點 4. 有左子點和右子點 |
| pop\_front | 將array最後一個值取代第一個值，並將\_size-1 | 將\_head刪除，並將\_head指標指向下一個 | 刪除\_root最左邊的點 |
| pop\_back | 直接將\_size-1 | 將\_head前一個刪除 | 刪除\_root最右邊的點 |
| clear | 直接將\_size=0、\_data=0 | 將\_head的前面及後面接等於\_head | 將\_root的左子數等於零即可 |
| sort | 無 | 用insertion sort的方法完成 | 無 |

* 實驗預期
  + 跟據Time Complexity進行判斷

|  |  |  |  |
| --- | --- | --- | --- |
|  | Array | Double link list | Binary search tree |
| Insert | 0(1) | 0(1) | O(logn) |
| erase | 0(1) | 0(1)/ 0(n)  [不知iterator情況下] | O(logn) |
| pop\_front | 0(1) | 0(1) | O(logn) |
| pop\_back | 0(1) | 0(1) | O(logn) |
| clear | 0(1) | 0(1) | O(1) |
| sort | 0(n\*n) | 0(n\*n) | 無 |

* 結果比較與討論
  + 輸入以下指令，記錄結果(單位：second)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Array | Double link list | Binary search tree |
| Insert  100000筆資料 | 0.03 | 0.02 | 0.19 |
| 刪除時，皆有100000筆資料情況下 | | | |
| 隨機delete  100000筆資料 | 0.01 | 20.93 | 229.9 |
| pop\_front  100000筆資料 | 0.01 | 0.02 | 0.24 |
| pop\_back  100000筆資料 | 0.02 | 0.02 | 0.23 |
| clear  100000筆資料 | 0.33 | 0.01 | 0.19 |
| sort  100000筆資料 | 0.07 | 120.7 | 無 |

* + 用作業所提供的測資do2進行比較(單位：second)

|  |  |  |  |
| --- | --- | --- | --- |
|  | Array | Double link list | Binary search tree |
| do2 | 0.54 | 20.31 | 21.64 |

* + 討論
    - Array為連續記憶體，除了在clear外，其餘皆為最快
    - Binary search tree本身建樹時，已經將資料排序好，但如果隨機刪除資料，必須每筆搜尋相當慢
    - Double link list如果改用merge sort 或是 quicksort，O(log n)的演算法，結果會更好
    - 結論，就實驗數據而言，不考慮其他用途，Array為最好的方式