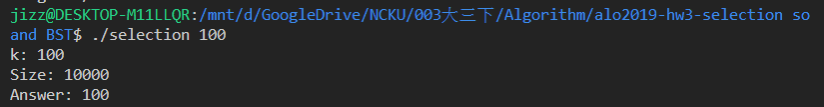
Algorithm HW3

C44051037 地科109 謝達永

Compile Command: make

第一部分: Selection algorithm

1. 執行command(尋找第k小的element): ./selection k
2. input file格式: 與之前相同，第一行為data數，之後開始才是data
3. 執行結果: 如下圖，程式會印出k、input data size與執行結果Answer，下圖剛好第100小的element是100。
4. 程式說明:

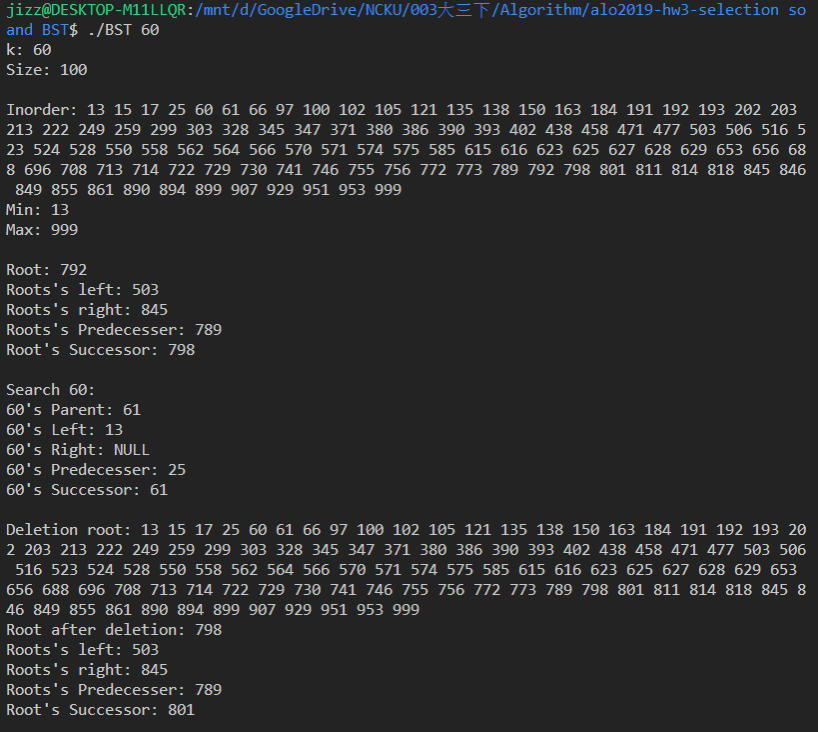
partition與insertion sort的部分基礎上直接採用上次的作業，只有partition有加上指定pivot的功能。

主要的selection部分，基礎沿用講義的algorithm一步一步照著寫，再參考網路上的資源，修改了一些細節。

首先是假如向下遞迴到資料量不成一組5個時，直接做insertion sort後return median。

再來是在對每一組做insertion sort後取median的部分，技術性的把每個median放到當前list中的最前面(從p開始放)，之後直接對最前面那一段遞迴取中位數，因為這樣可以直接再遞迴回來時，直接知道下一步要拿來當作pivot之median的index值，方便partition使用。但是這種操作會導致這裡的selection與下面的selection在傳輸遞四個參數k上會有些不同，造成在第一個if片段中的return時會有Segmentation fault問題(因為p+k-1超過list大小導致Access memory which not belong to you)，不過在最開始的insertion sort後return median的部分加個條件判斷，在超過range時return 0即可，因為只有第一次的selection遞迴會產生這個問題，但是那裏來說return值不重要。

第二部分: Binary Search Tree

1. 執行command(k為要拿來測試search的數值): ./BST k
2. input file格式: 與之前相同，第一行為data數，之後開始才是data
3. 執行結果: 如下圖，有稍微寫了一個測試用的mian() program。首先會印出預計要search的key值與總資料數，後印出排完之BST的inorder、MIN value、MAX value。再來分別印出root的相關資料與serach結果之node的相關資料，最後再把root刪掉。
4. 程式說明:

Tree的主體部分採用class建置，每個node即為一個class物件，每個物件中會有四項基礎資料: key、parent、left與right，其中附帶一建構子。另外準備一個Global variable pointer拿來指向root的位置，把它當作整棵樹，所以之後再建置樹時，直接insertion root即可。

Tree operation主要依照PPT上的虛擬碼實作，基礎上沒遇到太多問題。比較有遇到的問題是在實作PPT上的deletion時遇到的問題，在虛擬碼的第16行寫道”copy y’s satellite data into z”，我本來以為所謂”satellite data”是指pointer部分像right那樣的structure用變數，但幾經查證後發現完全相反，那紙的反而是實際上的資料像key那樣的data變數。