軟體工程實務的一封信

親愛的修課同學

歡迎來到中央大學也歡迎來到軟體工程實務這門課。有鑑於各位來自不同的學校，在各式各樣的學校文化風氣陶養下，難免對某些事情有不一樣的認知。過去也許在你過去的學校裡，拿同學的程式稍作修改一下然後做為你自己的作業繳交是沒有問題的。當然，也許你隱約地認知到這是不對的，因為你出社會之後每個人的工作通常不會重複的，所以你自己的工作通常沒有的抄。

是的，程式作業抄襲的認定，其實通常比你們想的更為直覺。你們可能說服自己，現在 open source 的世界不是程式互相抄來抄去? 又沒有什麼問題? 其實那只是你涉世未深，還不曉得真實世界的運作所營造出來的過於單純的想法。另外，你以後出社會的要面對的工作，其實沒有人寫程式給你抄的。你所要完成的工作，也幾乎是沒有 open source 可以抄的。最後，我們是學校，就算真實社會上有這一類的抄襲的叢林法則，學校也不能教你們這些事情。

所以，如果你過去學習方式是幾乎很少自行完成作業與程式習作，**建議你退選這一門課。一切都還來的及。**

所以，你應該怎麼做?

A. 你可以跟同學討論問題，可能的解決方法，但是不能看同學的程式或報告
B. 程式與報告你必須自己獨力完成。

理論上你遵守上述兩點，你所繳交的程式或是報告，應該永遠不會有超過 10%的內容跟別人一樣。

剽竊認定標準:

剽竊的認定沒有標準。就是老師與助教願意花多大的力氣來跟你們週旋。如同我說的，如果你不是真正來學習的，我們勸你退選，那我們就輕鬆一點。

過去的案例

幾份作業或程式一模一樣，辯稱幾個人一起做的。抱歉，我們不接受這種解釋，你們幾位的作業都會是 0 分。另外我們可能懷疑你的誠實問題，會回頭檢驗你所有的作業。

所以，如果你是原始版本，你是個好心想要幫助同學的人，你可以口頭上暗示或告訴他們你的做法，你也甚至於可以幫他們 debug，這樣子的話，你的原始版本才不會流出。請記得保障你自己的權益。

HomeWork1: Debug an AVL implementation

說明:

在這個練習中，一個 AVLtree-incorrect.java 的程式提供給你。這個程式是一個 AVL tree 的實作。 AVL tree 是一個非常基本的資料結構。這個程式如果能正確的執行起來，可以與使用者進行下列如圖 1 的互動。不過很不幸地，這個程式是錯的。錯誤發生在一些地方，有兩個 bugs。

你的任務:

1. 如果你已經忘得一乾二淨。先上網 google AVL tree 是什麼資料結構。
2. 先將錯誤程式能夠在你的環境執行起來。如果你能找些關鍵的測試發現錯誤，please do。但是正常的方法是你先找幾個 test runs，尤其是讓程式出錯的，產生不正確的輸出。但是你使用人腦推導，知道正確的輸出應該為何。
3. 你應該用除錯器設定中斷點來加速你的除錯與發現問題。當然，沒有人能阻止你看 source code 來除錯。誠實告訴你，我將 bug 隱藏得很好，你如果堅持這樣子除錯，儘早抓出 bug 可能不容易。
4. 當你修復好 bugs 之後，請按照助教指示，上傳你的檔案。
5. 你的成績將由助教根據
   甲、是否成功修正 bug

提示:

1. 能夠 trace 別人的 source code 並修正別人的錯誤，是這個 lab 的主要目的
2. 表一有提示部分的測試行為，你可以參考表一的測試行為來幫助你理解程式原本正確的行為

表一: 程式的正確執行結果（**粗體字**表示輸入，非粗體字為程式輸出）

```
AVLTree Tree Test


AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
4
Empty status = true

Post order :
Pre order :
In order :
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
10

Post order : 10
Pre order : 10
In order : 10
Do you want to continue (Type y or n)

y
```

```
AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
9

Post order : 9 10
Pre order : 10 9
In order : 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
8

Post order : 8 10 9
Pre order : 9 8 10
In order : 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations
```

```
1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
7

Post order : 7 8 10 9
Pre order : 9 8 7 10
In order : 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
6

Post order : 6 8 7 10 9
Pre order : 9 7 6 8 10
In order : 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
```

```
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
5

Post order : 5 6 8 10 9 7
Pre order : 7 6 5 9 8 10
In order : 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
4

Post order : 4 6 5 8 10 9 7
Pre order : 7 5 4 6 9 8 10
In order : 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
```

```
4. check empty
5. clear tree
1
Enter integer element to insert
3

Post order : 3 4 6 5 8 10 9 7
Pre order : 7 5 4 3 6 9 8 10
In order : 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
2

Post order : 2 4 3 6 5 8 10 9 7
Pre order : 7 5 3 2 4 6 9 8 10
In order : 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
```

```
1
Enter integer element to insert
1

Post order : 1 2 4 6 5 3 8 10 9 7
Pre order : 7 3 2 1 5 4 6 9 8 10
In order : 1 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
1
Enter integer element to insert
0

Post order : 0 2 1 4 6 5 3 8 10 9 7
Pre order : 7 3 1 0 2 5 4 6 9 8 10
In order : 0 1 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
3
Nodes = 11
```

```
Post order : 0 2 1 4 6 5 3 8 10 9 7
Pre order : 7 3 1 0 2 5 4 6 9 8 10
In order : 0 1 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
2
Enter integer element to search
12
Search result : false

Post order : 0 2 1 4 6 5 3 8 10 9 7
Pre order : 7 3 1 0 2 5 4 6 9 8 10
In order : 0 1 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
2
Enter integer element to search
4
Search result : true
```

```
Post order : 0 2 1 4 6 5 3 8 10 9 7
Pre order : 7 3 1 0 2 5 4 6 9 8 10
In order : 0 1 2 3 4 5 6 7 8 9 10
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
5

Tree Cleared

Post order :
Pre order :
In order :
Do you want to continue (Type y or n)

y

AVLTree Operations

1. insert
2. search
3. count nodes
4. check empty
5. clear tree
4
Empty status = true

Post order :
Pre order :
```

```
In order :
Do you want to continue (Type y or n)

n
```