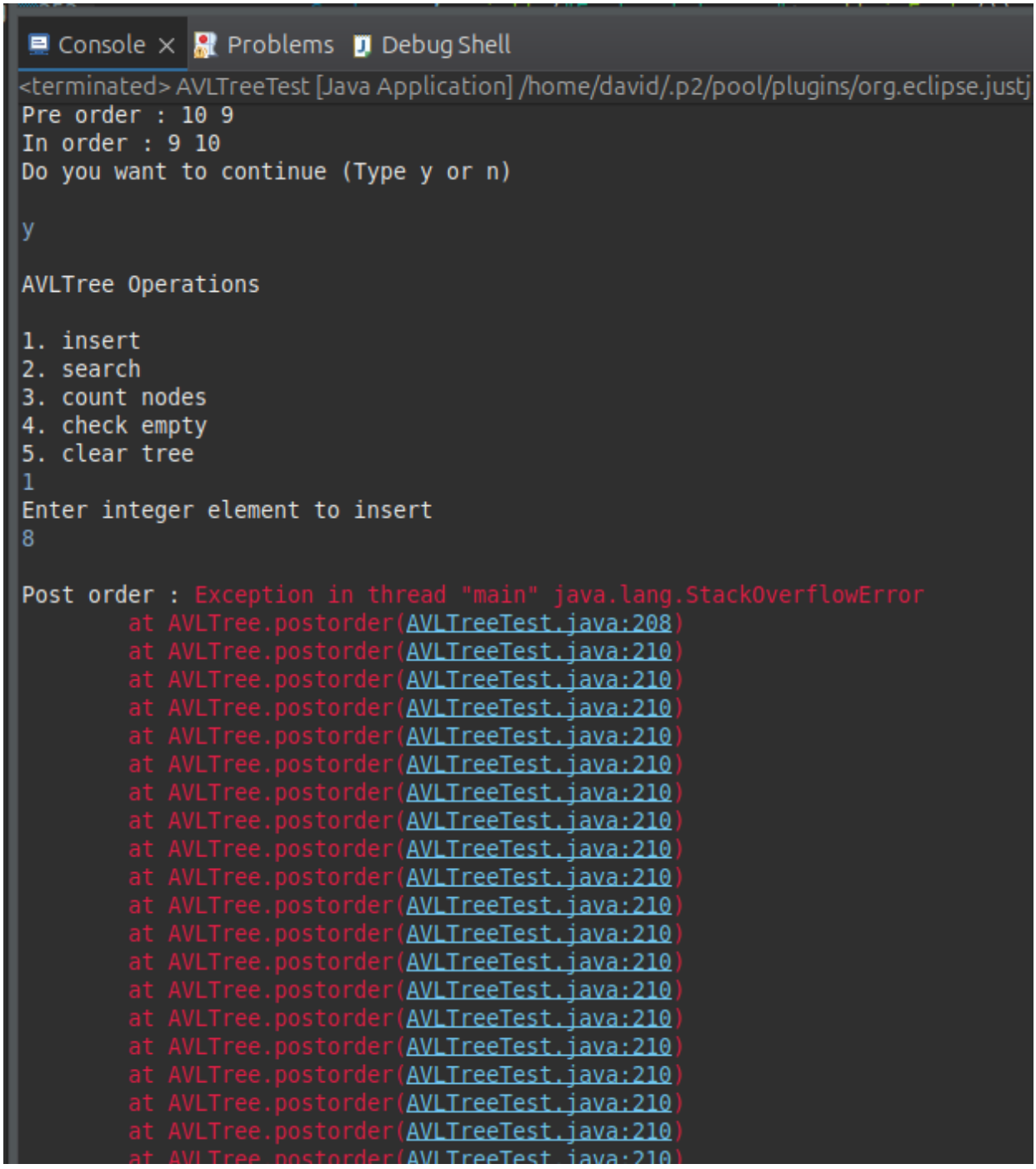
HW1: debug an AVL tree

> 軟工碩一 113525011 江大衛
> 題目要求

ANS:

bug1

- 原因：在輸入 8 的時候會發生錯誤
- 發生：不斷遞迴導致 stack overflow error.



PROF

修復

- bug1: `rotateWithLeftChild` 傳送的參數不正確

```
84              if( height( t.right ) - height( t.left ) == 2 )
85                  if( x > t.right.data)
86  -                   t = rotateWithRightChild( t.right );
87              else
88                  t = doubleWithRightChild( t );
89          }
```

```
84              if( height( t.right ) - height( t.left ) == 2 )
85                  if( x > t.right.data)
86  +                   t = rotateWithRightChild( t);
87              else
88                  t = doubleWithRightChild( t );
89          }
```

```
@@ -95,22 +95,31 @@ else if( x > t.data )
```

- bug2: `rotateWithLeftChild`, `rotateWithRightChild` rotate 錯誤
  - 解法
    - 新增一個 AVLNode 做為回傳值。
    - 將 function 傳近來的參數進行旋轉，並讓新增的 AVLNode 做為 root
    - 在旋轉的過程中，如果有資料會被排擠到，則將它重新做一次 insert.

```
95          /* Rotate binary tree node with left child */
96          private AVLNode rotateWithLeftChild(AVLNode k2)
97  -       {
98              AVLNode k1 = k2.left;
99  -           k2.right = k1.left;
100 -           k1.left = k2;
101
102             k2.height = max( height( k2.left ), height( k2.right ) ) + 1;
103             k1.height = max( height( k1.left ), k2.height ) + 1;
104
105             return k1;
106         }
107
108         /* Rotate binary tree node with right child */
109         private AVLNode rotateWithRightChild(AVLNode k1)
110         {
111 -           AVLNode k2 = k1.right;
112 -           k1.left = k2.right;
113 -           k2.right = k1;
114 -           k1.height = max( height( k1.left ), height( k1.right ) ) + 1;
115 -           k2.height = max( height( k2.right ), k1.height ) + 1;
116
117             return k2;
118         }
```

```
95          /* Rotate binary tree node with left child */
96          private AVLNode rotateWithLeftChild(AVLNode k2)
97  +       {
98              AVLNode k1 = k2.left;
99  +           AVLNode k3 = k1.right;
100 +           k2.left = null;
101 +           k1.right = k2;
102             k2.height = max( height( k2.left ), height( k2.right ) ) + 1;
103             k1.height = max( height( k1.left ), k2.height ) + 1;
104 +           if(k3 != null) {
105 +                   insert(k3.data);
106 +           }
107             return k1;
108         }
109
110         /* Rotate binary tree node with right child */
111         private AVLNode rotateWithRightChild(AVLNode k1)
112         {
113 +           AVLNode k2 = k1.right;
114 +           AVLNode k3 = k2.left;
115 +           k1.right = null;
116 +           k2.left = k1;
117 +           k1.height = max( height( k1.left ), height( k1.right ) ) + 1;
118 +           k2.height = max( height( k2.right ), k1.height ) + 1;
119 +           if(k3 != null) {
120 +                   insert(k3.data);
121 +           }
122             return k2;
123         }
```