DATA STRUCTURES

ASSIGNMENT 3

NAME - SUGANDHI SAGGU

TCD ID – 21355223

Task 1:

Student STDOUT.txt

```
Raw Data:
FLOCCINAUCINIHILIPILIFICATION

Sorted Data:
AACCCCFFHIIIIIIIIILLLNNNOOPTU

A found
B is not in the dataset
C found
D is not in the dataset
E is not in the dataset
F found
G is not in the dataset
H found
I found
J is not in the dataset
K is not in the dataset
L found
M is not in the dataset
N found
O found
P found
Q is not in the dataset
R is not in the dataset
S is not in the dataset
T found
U found
V is not in the dataset
W is not in the dataset
X is not in the dataset
Y is not in the dataset
Z is not in the dataset
```

Expected STDOUT.txt

```
Raw Data:
FLOCCINAUCINIHILIPILIFICATION

Sorted Data:
AACCCCFFHIIIIIIIIILLLNNNOOPTU

A found
B is not in the dataset
C found
D is not in the dataset
E is not in the dataset
F found
G is not in the dataset
H found
I found
J is not in the dataset
K is not in the dataset
L found
M is not in the dataset
N found
O found
P found
Q is not in the dataset
R is not in the dataset
S is not in the dataset
T found
U found
V is not in the dataset
W is not in the dataset
X is not in the dataset
Y is not in the dataset
Z is not in the dataset
```

1 / 1 Using Valgrind to check for memory leaks

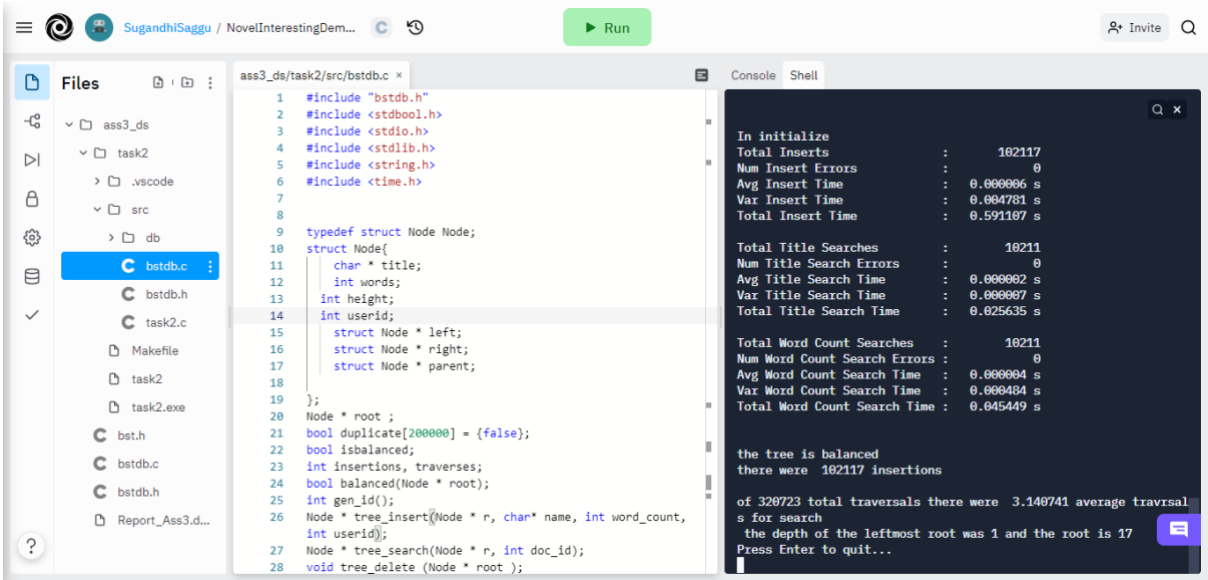
Hide Details

Visualize whitespace characters

Student Standard Error (STDERR)

```
==1253251== Memcheck, a memory error detector
==1253251== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==1253251== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info
==1253251== Command: ./p1.out
==1253251==
==1253251==
==1253251== HEAP SUMMARY:
==1253251==     in use at exit: 0 bytes in 0 blocks
==1253251==   total heap usage: 30 allocs, 30 frees, 4,792 bytes allocated
==1253251==
==1253251== All heap blocks were freed -- no leaks are possible
==1253251==
==1253251== For lists of detected and suppressed errors, rerun with: -s
==1253251== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Task 2:



In this data structures, I have used AVL Tree. It is a self balancing binary tree. In an AVL tree, the height of the two child subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property. The height of an AVL tree is always OLog(n), where n is the number of nodes.