

Test Log For MP5

Student: Yulin Xiao

CUID: C16278133

Student UserName: Yulinx

1. Do the unit driver with custom test 0 through 6 (4-6 is my added binary search tree tests) to verify that my code is able to do the basic binary search tree operation (insert, remove):

```
lab5 -u 0
```

```
//test to remove leaves, 12 and 20, then internal nodes
```

```
// 8, 24, 40 with one child, then 16, 48 with two children
```

```
ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -u 0
Seed: 1476379020

===== Unit Driver =====
Inserting 14 items into tree
Created tree for testing removes
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  24 28 20 12  8
 /  \  /  \ /  \
4   8 12 20 28 42 46

Removing 7 items from tree
-- Test (0) about to remove key 12
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  24 28 20  8  4
 /  \  /  \ /  \
4   8 20 28 42 46

-- Test (1) about to remove key 20
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  24 28  8  4
 /  \  /  \ /  \
4   8 20 28 42 46
```

```
-- Test (1) about to remove key 20
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  24 28  8  4
 /  \  /  \ /  \
4   8 20 28 42 46

-- Test (2) about to remove key 8
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  24 28  4
 /  \  /  \ /  \
4   8 20 28 42 46

-- Test (3) about to remove key 24
      56
     /  \
    48   46
   /  \  /  \
  32  40 44  42
 /  \  /  \ /  \
16  28  4
 /  \  /  \ /  \
4   8 20 28 42 46
```

```
-- Test (4) about to remove key 40
      48
      56
      46
      42
32      28
      16
      4

      /-32-----\
      /-16-\      /-44-\      /-48-\
      4      28      42      46      56

-- Test (5) about to remove key 16
      48
      56
      46
      42
32      28
      16
      4

      /-32-----\
      /-28-\      /-44-\      /-48-\
      4      42      46      56

-- Test (6) about to remove key 48
      56
      46
      42
32      28
      16
      4

      /-32-----\
      /-28-\      /-44-\      /-56-\
      4      42      46
```

lab5 -u 1

```
// tests: (48) is missing its right-left child and
```

```
//      (16) is missing its left-right child
```

```

===== Unit Driver =====
Inserting 13 items into tree
Created tree for testing removes
      60
     /  \
    56    4
   /  \  /  \
  48   40 44   36
 /  \  /  \ /  \
32   24 28  20 16
 /  \  /  \ /  \
16   8   4
 /  \ /  \ /  \ /  \
/------32-----\
/-16----\ /-----48-\
/-8-20-24-\ /-40-\ 56-\
4      28 36 44 60
Removing 3 items from tree
-- Test (0) about to remove key 16
      60
     /  \
    56    4
   /  \  /  \
  48   40 44   36
 /  \  /  \ /  \
32   24 28  20 16
 /  \  /  \ /  \
16   8   4
 /  \ /  \ /  \ /  \
/------32-----\
/-20-\ /-----48-\
/-8-24-\ /-40-\ 56-\
4      28 36 44 60
-- Test (1) about to remove key 48
      60
     /  \
    56    4
   /  \  /  \
  48   40 44   36
 /  \  /  \ /  \
32   24 28  20 16
 /  \  /  \ /  \
16   8   4
 /  \ /  \ /  \ /  \
/------32-----\
/-20-\ /-----48-\
/-8-24-\ /-40-\ 56-\
4      28 36 44 60

```

```

-- Test (1) about to remove key 48
      60
    56
  52
40
36
32
28
24
8
12
4

/-----\
/---24-\ /---52-\
/--8-\ /-40- \ 56-\
4 12 36 60

-- Test (2) about to remove key 32
      60
    56
  52
40
36
28
24
12
8
4

/-----\
/---24-\ /---36---\
/--8-\ /-52-\
4 12 40 56-\
      60

```

```
lab5 -u 2
// deletion with many children
```

```
ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -u 2
Seed: 1476379020

===== Unit Driver =====

Inserting 16 items into tree
Created tree for testing removes
200
      175
     /  \
    150  140
   /  \  \
  100 125 130
   /  \  \
  25  85  82
     /  \  \
    75  80  78
   /  \  \
  50  65
 /  \
25

      175
     /  \
    150  140
   /  \  \
  100 125 130
   /  \  \
  25  85  82
     /  \  \
    75  80  78
   /  \  \
  50  65
 /  \
25

      175
     /  \
    150  140
   /  \  \
  100 125 130
   /  \  \
  25  85  82
     /  \  \
    75  80  78
   /  \  \
  50  65
 /  \
25

Removing 3 items from tree
-- Test (0) about to remove key 100
200
      175
     /  \
    150  140
   /  \  \
  125 135 130
   /  \  \
  25  85  82
     /  \  \
    75  80  78
   /  \  \
  50  65
 /  \
25

      175
     /  \
    150  140
   /  \  \
  125 135 130
   /  \  \
  25  85  82
     /  \  \
    75  80  78
   /  \  \
  50  65
 /  \
25

-- Test (1) about to remove key 85
200
      175
     /  \
    150  140
   /  \  \
  125 135 130
   /  \  \
  25  82
     /  \
    75  78
   /  \
  50  65
 /  \
25

      175
     /  \
    150  140
   /  \  \
  125 135 130
   /  \  \
  25  82
     /  \
    75  78
   /  \
  50  65
 /  \
25

-- Test (2) about to remove key 125
200
      175
     /  \
    150  140
   /  \  \
  130 135 130
   /  \  \
  25  82
     /  \
    75  78
   /  \
  50  65
 /  \
25

      175
     /  \
    150  140
   /  \  \
  130 135 130
   /  \  \
  25  82
     /  \
    75  78
   /  \
  50  65
 /  \
25
```

```
lab5 -u 3
// check replace for duplicate key
```

```
Seed: 1476379020

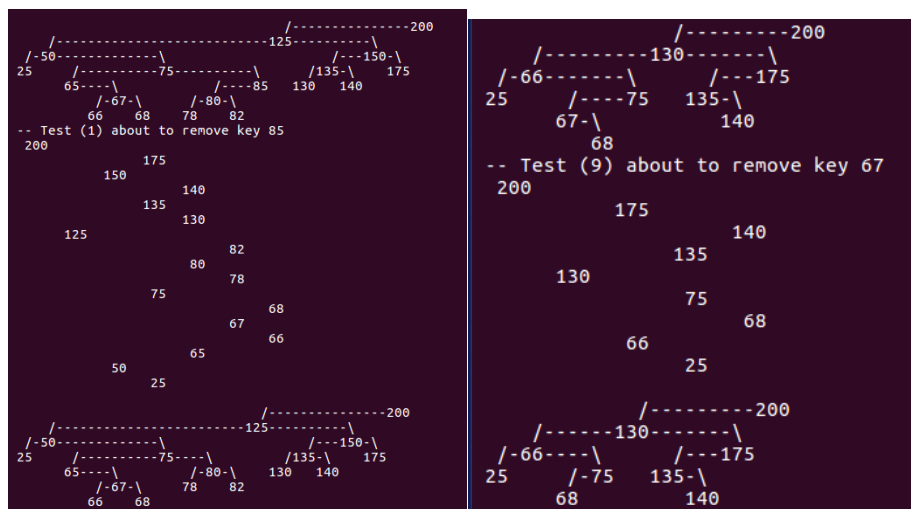
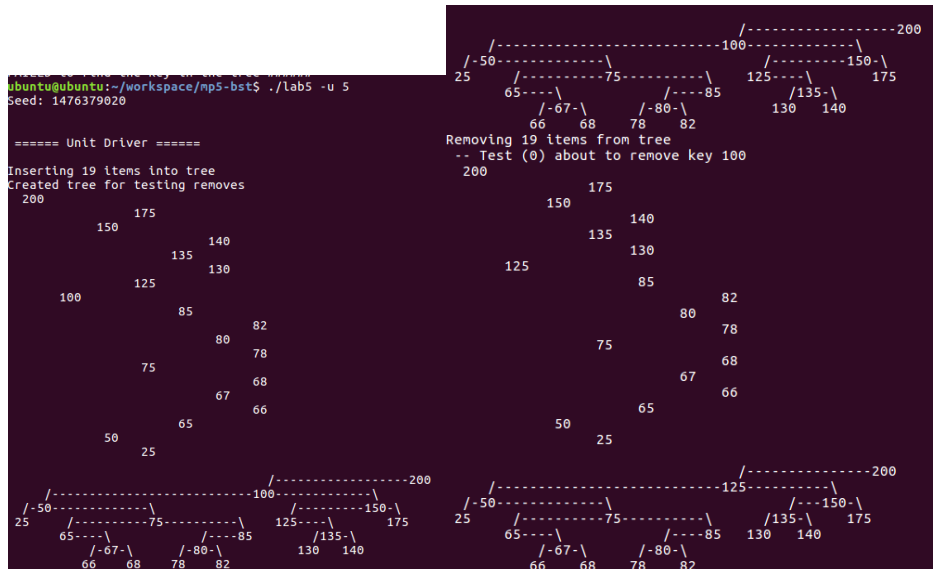
===== Unit Driver =====

Inserting 2 items into tree
Created tree for testing removes
10
10
Removing 1 items from tree
-- Test (0) about to remove key 10
```

```
lab5 -u 4
// check replace and double deletion for duplicate key
```

lab5 -u 5

// complete deletion, first for parent(100,85(L),65(R),200(Root)) and then
for child(67,68,66)



lab5 -u 6 // complete deletion, remove root



2) avl:

```
ubuntu@ubuntu:~/workspace/mps-bst$ ./lab5 -f avl -o -w 5 -t 0 -v
Seed: 1476379020

----- Access driver -----
Access trials: 0
Levels for tree: 5
Build optimal tree with size=31
      62-0
     /  \
    60-1 58-0
   /  \  /  \
  56-2 52-1 54-0 50-0
 /  \  /  \ /  \ /  \
48-3 44-1 46-0 42-0 38-0 34-0 30-0 26-0
 /  \  /  \ /  \ /  \ /  \
40-2 36-1 32-4 28-1 24-2 20-1 18-0 14-0 10-0 6-0 2-0
 /  \  /  \ /  \ /  \ /  \ /  \
32-4 28-1 24-2 20-1 16-3 12-1 8-2 4-1 2-0
 /  \  /  \ /  \ /  \ /  \ /  \ /  \
2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62

----- End of access driver -----
```

```
ubuntu@ubuntu:~/workspace/mps-bst$ ./lab5 -f avl -r -w 5 -t 0 -v -s 1
Seed: 1

----- Access driver -----
Access trials: 0
Levels for tree: 5
Build random tree with size=31
      62-1
     /  \
    58-2 60-0
   /  \  /  \
  52-3 50-1 54-1 56-0
 /  \  /  \ /  \ /  \
46-4 42-1 48-0 44-0 38-0 36-1 32-0 28-0 24-0 20-1 18-0 14-0 10-0 6-0 2-0
 /  \  /  \ /  \ /  \ /  \ /  \ /  \ /  \
40-3 34-2 30-5 26-1 22-3 16-2 12-4 8-1 4-2 2-0
 /  \  /  \ /  \ /  \ /  \ /  \ /  \ /  \
2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62

----- End of access driver -----
```

```
ubuntu@ubuntu:~/workspace/mps-bst$ ./lab5 -f avl -p -w 5 -t 0 -v -s 2
Seed: 2

----- Access driver -----
Access trials: 0
Levels for tree: 5
Build poor tree with size=31
      62-0
     /  \
    58-2 60-1
   /  \  /  \
  50-3 54-1 52-0 56-0
 /  \  /  \ /  \ /  \
44-2 48-1 46-0 42-0 38-0 34-0 30-0 26-1 24-1 22-0 18-1 16-0 12-0 10-1 8-0 4-1 2-0
 /  \  /  \ /  \ /  \ /  \ /  \ /  \ /  \ /  \ /  \
36-4 32-2 28-1 24-1 20-2 14-3 6-2 4-1 2-0
 /  \  /  \ /  \ /  \ /  \ /  \ /  \ /  \ /  \ /  \
2 6 10 12 16 20 24 28 30 34 38 42 46 50 54 58 62

----- End of access driver -----
```

4. Do the command line arguments of:

```
lab5 -o -w 20 -t 1000000
// tests inserts and accesses
lab5 -r -w 20 -t 1000000
// same with random tree
lab5 -p -w 20 -t 1000000
// same with poor insertion order
```

To verify that the expected number of searches predicted by the theory matches the measured performance from my program to three significant digits when run with 1,000,000 trials. 1) bst:

```
ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f bst -o -w 20 -t 1000000
Seed: 1476379020

----- Access driver -----
Access trials: 1000000
Levels for tree: 20
Build optimal tree with size=1048575
After access exercise, time=977.116, tree size=1048575
Expect successful search=37, measured=19, trials=499682
Expect unsuccessful search=40, measured=19, trials=500318
----- End of access driver -----

ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f bst -r -w 20 -t 1000000
Seed: 1476379020

----- Access driver -----
Access trials: 1000000
Levels for tree: 20
Build random tree with size=1048575
After access exercise, time=1384.18, tree size=1048575
Expect successful search=49.9952, measured=28, trials=499871
Expect unsuccessful search=52.9952, measured=28, trials=500129
----- End of access driver -----

ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f bst -p -w 20 -t 1000000
Seed: 1476379020

----- Access driver -----
Access trials: 1000000
Levels for tree: 20
Build poor tree with size=1048575
After access exercise, time=5889.4, tree size=1048575
Expect successful search=1042, measured=1032, trials=499703
Expect unsuccessful search=1045, measured=1032, trials=500297
----- End of access driver -----
```

2) avl:

```
ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f avl -p -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build poor tree with size=1023
After access exercise, time=0.133, tree size=1023
Expect successful search=17.4633, measured=11, trials=467
Expect unsuccessful search=20.4434, measured=11, trials=533
----- End of access driver -----

ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f avl -r -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build random tree with size=1023
After access exercise, time=0.131, tree size=1023
Expect successful search=17.4282, measured=10, trials=477
Expect unsuccessful search=20.4082, measured=10, trials=523
----- End of access driver -----

ubuntu@ubuntu:~/workspace/mp5-bst$ ./lab5 -f avl -o -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build optimal tree with size=1023
After access exercise, time=0.185, tree size=1023
Expect successful search=17.0196, measured=9, trials=471
Expect unsuccessful search=20, measured=9, trials=529
----- End of access driver -----
```

bst: best: 37, 40
average: 50, 53
worst: 1042, 1045
avl: best: 17, 20
average: 17, 20

worst: 17, 20

5. I find out that in standish's textbook:

Case	C_n	C_n'
Best	$2\log_2 n - 3$	$\log_2 n$
Average	$2.77\log_2 n - 4.7$	$2.77\log_2 n - 1.7$
Worst	n	$n + 2$

And what I found: (bst->20 levels)

Case	$C_{2^{20}}$ (Mine)	$C_{2^{20}}'$ (Mine)
Best	$2\log_2 2^{20} - 3 = 37 (37)$	$2\log_2 2^{20} = 40 (40)$
Average	$2.77\log_2 2^{20} - 4.7 = 50.7 (50)$	$2.77\log_2 2^{20} - 1.7 = 53.7 (53)$
Worst	$2^{20} (1000000) (1042)$	$2^{20} + 2 (1000002) (1045)$

And what I found: (avl->10 levels)

Case	$C_{2^{10}}$ (Mine)	$C_{2^{10}}'$ (Mine)
Best	$2\log_2 2^{10} - 3 = 17 (17)$	$2\log_2 2^{10} = 20 (20)$
Average	$2.77\log_2 2^{10} - 4.7 = 23 (17)$	$2.77\log_2 2^{10} - 1.7 = 26 (20)$
Worst	$2^{10} (1000) (17)$	$2^{10} + 2 (1002) (20)$

- 1) So by comparison of my number of successful and unsuccessful searches, it just perfectly match the Approximate Average Number of Comparisons for successful and unsuccessful search , so that my binary search tree works well in three cases.
- 2) As for worst case, I guess its time of complexity is $O(\sqrt{n})$, for it generates trees that only have two subtrees that looks like linked list. For the time complexity of remove, insert and access of a certain node in linked list is $O(n)$, and the poor tree looks like ^, it cuts more than 2 times in every moves downward.
- 3) As for the avl tree because its balanced characteristic, so, no matter in what policy the node is inserted, the avl tree matches the best cases.

EXTENDED FOR REPLACEMENT MP!

1. Test through unit test driver as same as what I do for bst testing for No.1 test plan, but rather than do: ./lab5 -u 0-6, do: ./lab5 -f -avl -u 0-7

(1) ./lab5 -f -avl -u 0 (first go up to down and then go right)

```
Removing 7 items from tree
-- Test (0) about to remove key 12
      48-1
    44-2 46-0
      42-1
    32-3 40-0
      24-1
    16-2 28-0
      8-1
    4-0

      /-----32-----\
    /-16-\ /-24-\ /-42-\ /-44-\ /-48-\
    4 20 28 40 46 56
-- Test (1) about to remove key 20
      48-1
    44-2 46-0
      42-1
    32-3 40-0
      24-1
    16-2 28-0
      8-1
    4-0

      /-----32-----\
    /-16-\ /-24-\ /-42-\ /-44-\ /-48-\
    4 20 28 40 46 56
-- Test (2) about to remove key 8
      48-1
    44-2 46-0
      42-1
    32-3 40-0
      24-1
    16-2 28-0
      4-0

      /-----32-----\
    /-16-\ /-24-\ /-42-\ /-44-\ /-48-\
    4 20 28 40 46 56
-- Test (3) about to remove key 24
      48-1
    44-2 46-0
      42-1
    32-3 40-0
      16-1
    4-0

      /-----32-----\
    /-16-\ /-24-\ /-42-\ /-44-\ /-48-\
    4 20 28 40 46 56
-- Test (4) about to remove key 40
      48-1
    44-2 46-0
      42-0
    32-3 28-0
      16-1
    4-0

      /-----32-----\
    /-16-\ /-24-\ /-42-\ /-44-\ /-48-\
    4 20 28 40 46 56
-- Test (5) about to remove key 16
      48-1
    44-2 46-0
      42-0
    32-3 28-1
      4-0

      /-32-----\
    /-28-\ /-44-\ /-48-\
    4 40 46 56
-- Test (6) about to remove key 48
      4
    /-28-\ /-44-\ /-48-\
    4 42 46 56
      46
```

(2) ./lab5 -f -avl -u 1

```
Removing 3 items from tree
-- Test (0) about to remove key 16
      56-0
    48-2 52-0
      40-1
    32-3 36-0
      24-2
    28-0
      8-1
    12-0
      4-0

      /-----32-----\
    /-24-\ /-48-\
    /-8-\ /-40-\ /-56-\
    4 12 36 52 60
-- Test (1) about to remove key 48
      56-1
    52-2 40-1
      36-0
    32-3 28-0
      24-2
    12-0
      8-1
    4-0

      /-----32-----\
    /-24-\ /-48-\
    /-8-\ /-40-\ /-52-\ /-56-\
    4 12 36 52 60
-- Test (2) about to remove key 32
      56-1
    52-2 40-0
      36-3
    32-3 28-0
      24-2
    12-0
      8-1
    4-0

      /-----36-----\
    /-24-\ /-52-\
    /-8-\ /-40-\ /-56-\
    4 12 40 56 60
```

(3)./lab5 -f avl -u 2

```

      /-----100-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /130-\   /200
   25  65  78  82  125  135  140  175
Removing 3 items from tree
-- Test (0) about to remove key 100

      /-----125-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /135-\   /200
   25  65  78  82  130  140  175
-- Test (1) about to remove key 85

      /-----130-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /135-\   /200
   25  65  78  82  140  175

```

(4)./lab5 -f avl -u 3 and 4(just as same as bst, because only one node, do not involve with rotation)

(5)./lab5 -f avl -u 5 (only the original and the last five steps are shown)

```

      /130-----\
     /-75-\   /-175-\
    25  135  200
-- Test (12) about to remove key 175
      140-1  200-0
      130-2  135-0
      75-1  25-0

      /130-----\
     /-75-\   /140-\
    25  135  200
-- Test (13) about to remove key 130
      140-1  200-0
      135-2  75-1  25-0

      /135-\
     /-75-\   /140-\
    25  140  200
-- Test (14) about to remove key 135
      200-0
      140-2  75-1  25-0

      /140-\
     /-75-\   200
    25
-- Test (15) about to remove key 75
      200-0
      140-1  25-0

      /140-\
     25  200
-- Test (16) about to remove key 140
      200-1  25-0

      /200
     25
-- Test (17) about to remove key 200
      25-0

      /200
     25
-- Test (18) about to remove key 25

```

(6)./lab5 -f avl -u 6

```

ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -u 6
Seed: 1476379020

===== Unit Driver =====
Inserting 16 items into tree
Created tree for testing removes
      200-1
     150-3  175-0
           140-0
          135-1
         130-2  125-0
        100-4  85-1  82-0
              80-2  78-0
             75-3  65-0
                50-1  25-0

      /-----100-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /130-\   /200
   25  65  78  82  125  135  140  175
Removing 16 items from tree
-- Test (0) about to remove key 100
      200-1
     150-2  175-0
           140-0
          135-1
         130-0
        125-4  85-1  82-0
              80-2  78-0
             75-3  65-0
                50-1  25-0

      /-----125-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /135-\   /200
   25  65  78  82  130  140  175
-- Test (1) about to remove key 125

      200-1  175-0
     150-2  140-0
    135-4  85-1  82-0
          80-2  78-0
         75-3  65-0
            50-1  25-0

      /-----135-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /140-\   /200
   25  65  78  82  175
-- Test (2) about to remove key 130
      200-1  175-0
     150-2  140-0
    135-4  85-1  82-0
          80-2  78-0
         75-3  65-0
            50-1  25-0

      /-----135-----\
     /-----75-----\   /-----150-----\
    /-50-\   /-80-\   /140-\   /200
   25  65  78  82  175
-- Test (3) about to remove key 135

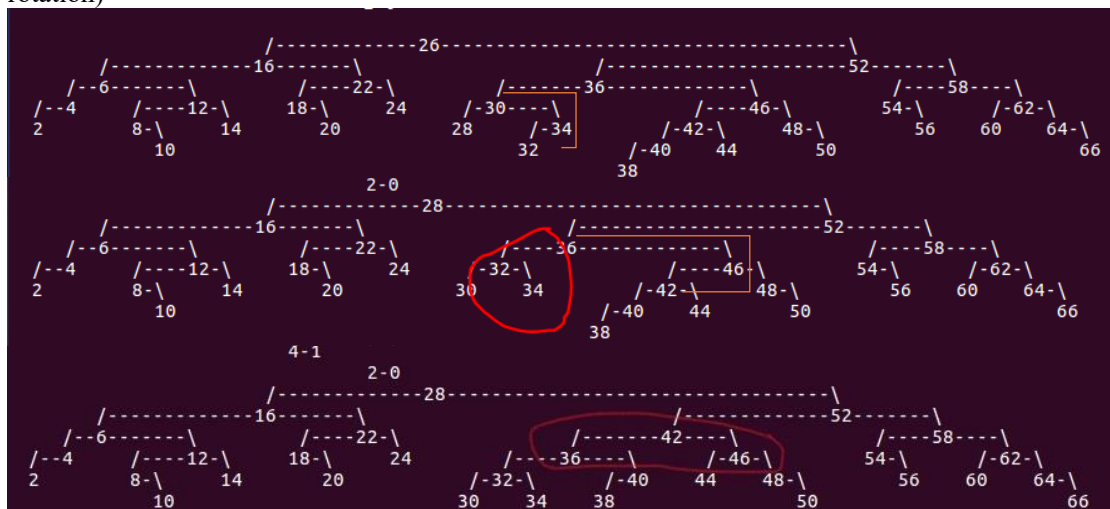
```

```

      200-0
    175-1 150-0
  140-2 85-1 82-0
80-3 78-0
  75-2 65-0
    50-1 25-0
      /---80-----\
    /---75-\ /---140---\
  /-50-\ 78 /-85 150 /175-\
25 65 82 200
-- Test (4) about to remove key 80
      200-0
    175-1 150-0
  140-2 85-0
82-3 78-0
  75-2 65-0
    50-1 25-0
      /---82-----\
    /---75-\ /---140---\
  /-50-\ 78 /-85 150 /175-\
25 65 82 200
-- Test (5) about to remove key 82
      200-0
    175-2 150-0
  140-1 85-3
75-2 78-0
  50-1 65-0
    25-0
      /---85-----\
    /---75-\ /---175-\
  /-50-\ 78 /-140-\ 200
25 65 150
-- Test (6) about to remove key 85
      200-0
    175-1 150-0
  140-3 78-0
75-2 65-0
  50-1 25-0
      /---140---\
    /---75-\ /---175-\
  /-50-\ 78 /-150-\ 200
25 65 150
-- Test (7) about to remove key 140
      200-0
    175-1 150-0
  150-3 78-0
75-2 65-0
  50-1 25-0
      /---150---\
    /---75-\ /---175-\
  /-50-\ 78 /-200-\
25 65 150
-- Test (8) about to remove key 150
      200-0
    175-1 78-0
75-2 65-0
  50-1 25-0
      /---75---\
    /---50-\ /---175-\
  /-25-\ 65 /-200-\
25 65 150
-- Test (9) about to remove key 75
      200-0
    175-1 65-0
78-2 50-1 25-0
      /---78-\
    /---50-\ /---175-\
  /-25-\ 65 /-200-\
25 65 150
-- Test (10) about to remove key 78
      200-0
    175-2 65-0
  50-1 25-0
      /---175-\
    /---50-\ /---200-\
  /-25-\ 65 /-200-\
25 65 150
-- Test (11) about to remove key 175
      200-0
    65-2 50-1 25-0
      /---65-\
    /---50-\ /---200-\
  /-25-\ 65 /-200-\
25 65 150
-- Test (12) about to remove key 65
      200-0
    50-1 25-0
      /---50-\
    /---25-\ /---200-\
  /-25-\ 65 /-200-\
25 65 150
-- Test (13) about to remove key 50
      200-1
    25-0
      /200
25
-- Test (14) about to remove key 200
      25-0
25
-- Test (15) about to remove key 25
ubuntu@ubuntu:~/Desktop/mp5-avl-remove$

```

(7) ./lab5 -f avl -u 7(Original, After 1st rotation, After 2rd rotations, every rotation is a double rotation)



- For all the combination of options, since I have already done the BST deletion in test plan #4 and #5, so that I only do the command line of :

“./lab5 -f avl -p -w 10 -t 1000”// performance evaluation(Like redo of #5)

“./lab5 -f avl -r -w 10 -t 1000”

“./lab5 -f avl -b -w 10 -t 1000”

and example test:

“./lab5 -f avl -e -w 16 -t 100000”// exercise tree (Like redo of #4)

and

“./lab5 -f avl -e -w 5 -t 10 -v -s 2”// tests random tree operations(Like redo of #3)

(1) `./lab5 -f avl -p -w 10 -t 1000`// performance evaluation(Like redo of #5)

`./lab5 -f avl -r -w 10 -t 1000`

`./lab5 -f avl -b -w 10 -t 1000`

They are the same as what I have tested during performance evaluations.

```
ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -o -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build optimal tree with size=1023
After access exercise, time=0.129, tree size=1023
Expect successful search=17.0196, measured=9, trials=471
Expect unsuccessful search=20, measured=9, trials=529
----- End of access driver -----

ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -r -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build random tree with size=1023
After access exercise, time=0.128, tree size=1023
Expect successful search=17.4282, measured=10, trials=477
Expect unsuccessful search=20.4082, measured=10, trials=523
----- End of access driver -----

ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -p -w 10 -t 1000
Seed: 1476379020

----- Access driver -----
Access trials: 1000
Levels for tree: 10
Build poor tree with size=1023
After access exercise, time=0.126, tree size=1023
Expect successful search=17.4633, measured=11, trials=467
Expect unsuccessful search=20.4434, measured=11, trials=533
----- End of access driver -----

ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -e -w 10 -t 1000
Seed: 1476379020

----- Equilibrium test driver -----
Trials in equilibrium: 1000
Levels in initial tree: 10
Initial random tree size=1023
Expect successful search for initial tree=17.4282
Expect unsuccessful search for initial tree=20.4082
After exercise, time=13.847, new tree size=1023
successful searches during exercise=9.1996, trials=506
unsuccessful searches during exercise=9.80769, trials=494
Validating tree...passed
After access experiment, time=0.133, tree size=1023
Expect successful search=17.4418, measured=9, trials=501
Expect unsuccessful search=20.4219, measured=9, trials=499
----- End of equilibrium test -----
```

(2) `./lab5 -f avl -e -w 16 -t 100000`// exercise tree (Like redo of #4)

That is the ultimate limit short running time of my avl tree, I cannot wait further (eg. I really don't know how long will it take beyond that). If you run 20 levels with 1,000,000 trials, you might wait for hours.

```
ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -e -w 16 -t 100000
Seed: 1476379020

----- Equilibrium test driver -----
Trials in equilibrium: 100000
Levels in initial tree: 16
Initial random tree size=65535
Expect successful search for initial tree=29.7222
Expect unsuccessful search for initial tree=32.7217
After exercise, time=248699, new tree size=65699
successful searches during exercise=15.356, trials=50078
unsuccessful searches during exercise=15.9348, trials=49922
Validating tree...passed
After access experiment, time=34.02, tree size=65699
Expect successful search=29.6901, measured=17, trials=50285
Expect unsuccessful search=32.6896, measured=17, trials=49715
----- End of equilibrium test -----
```

(3) *“./lab5 -f avl -e -w 5 -t 10 -v -s 2”// tests random tree operations(Like redo of #3)*

```

ubuntu@ubuntu:~/Desktop/mp5-avl-remove$ ./lab5 -f avl -e -w 5 -t 10 -v -s 2
Seed: 2

----- Equilibrium test driver -----
Trials in equilibrium: 10
Levels in initial tree: 5
Initial random tree size=31
Expect successful search for initial tree=7.58065
Expect unsuccessful search for initial tree=10.25

      62-1
        60-0
          58-2
            56-0
              54-1
                52-3
                  50-0
                    48-1
                      46-2
                        44-0
                          42-1
                            40-0
                              38-5
                                36-1
                                  34-0
                                    32-2
                                      30-0
                                        28-3
                                          26-1
                                            24-0
                                              22-4
                                                20-0
                                                  18-1
                                                    16-2
                                                      14-0
                                                        12-3
                                                          10-1
                                                            8-0
                                                              6-2
                                                                4-0
                                                                  2-1

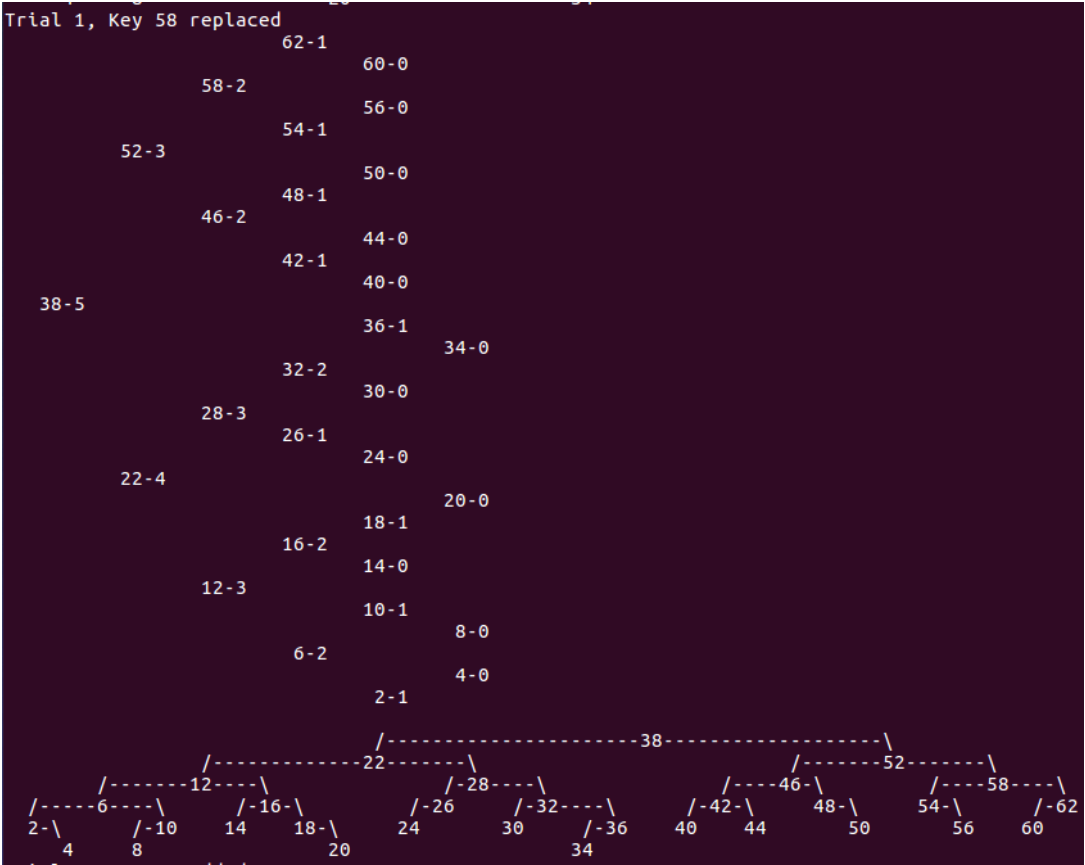
      /-----38-----\
     /-----22-----\
    /-----12-----\
   /-----6-----\
  /-----2-----\
 /-----4-----\
/-----8-----\
2- 4 8 10 14 18 20 24 26 28 30 32 34 36 40 42 44 46 48 50 52 54 56 58 60 62

```

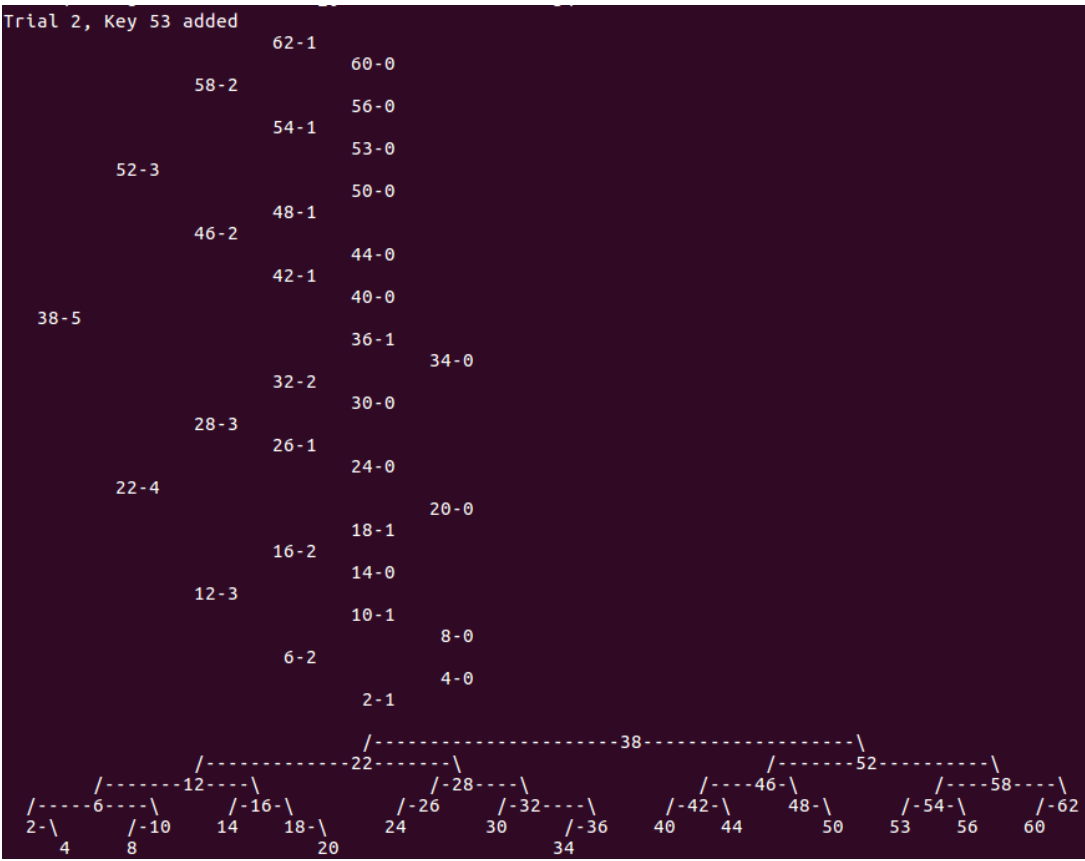
Trial 0, key 2 replaced

[illegible]

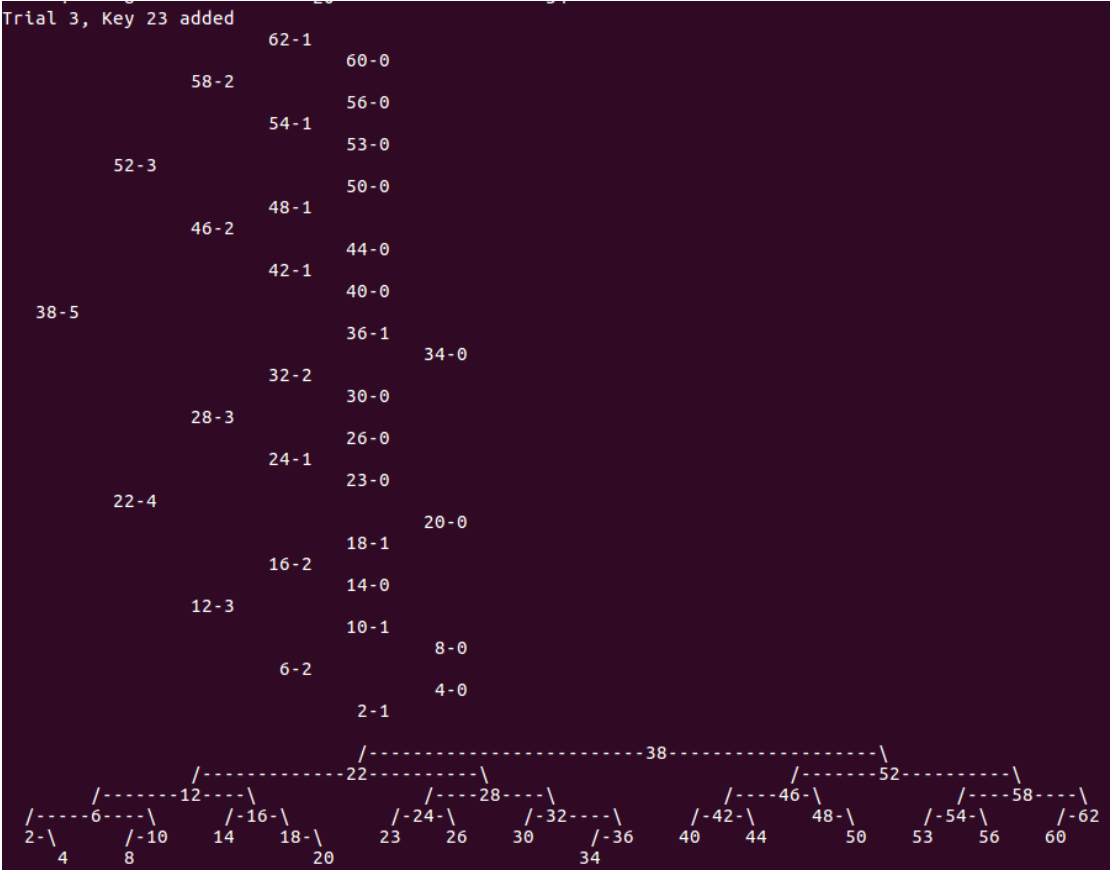
Trial 1, key 58 replaced



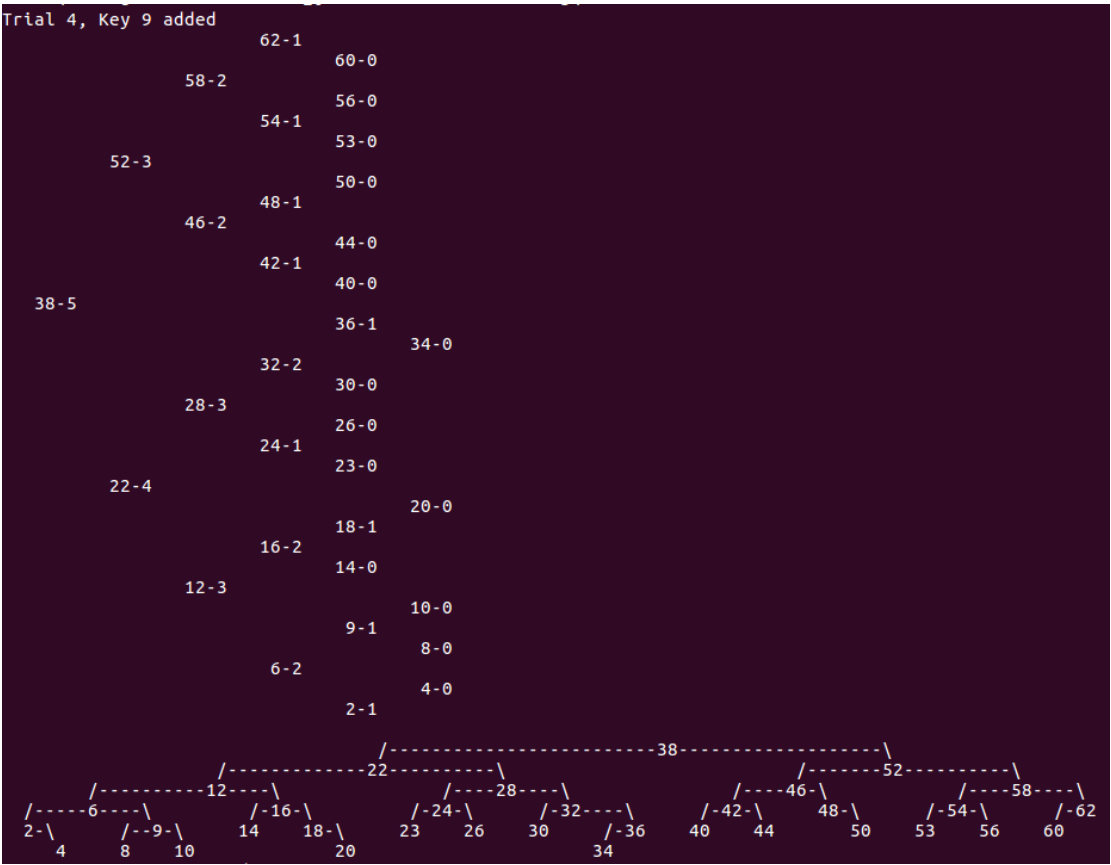
Trial 2, key 53 added



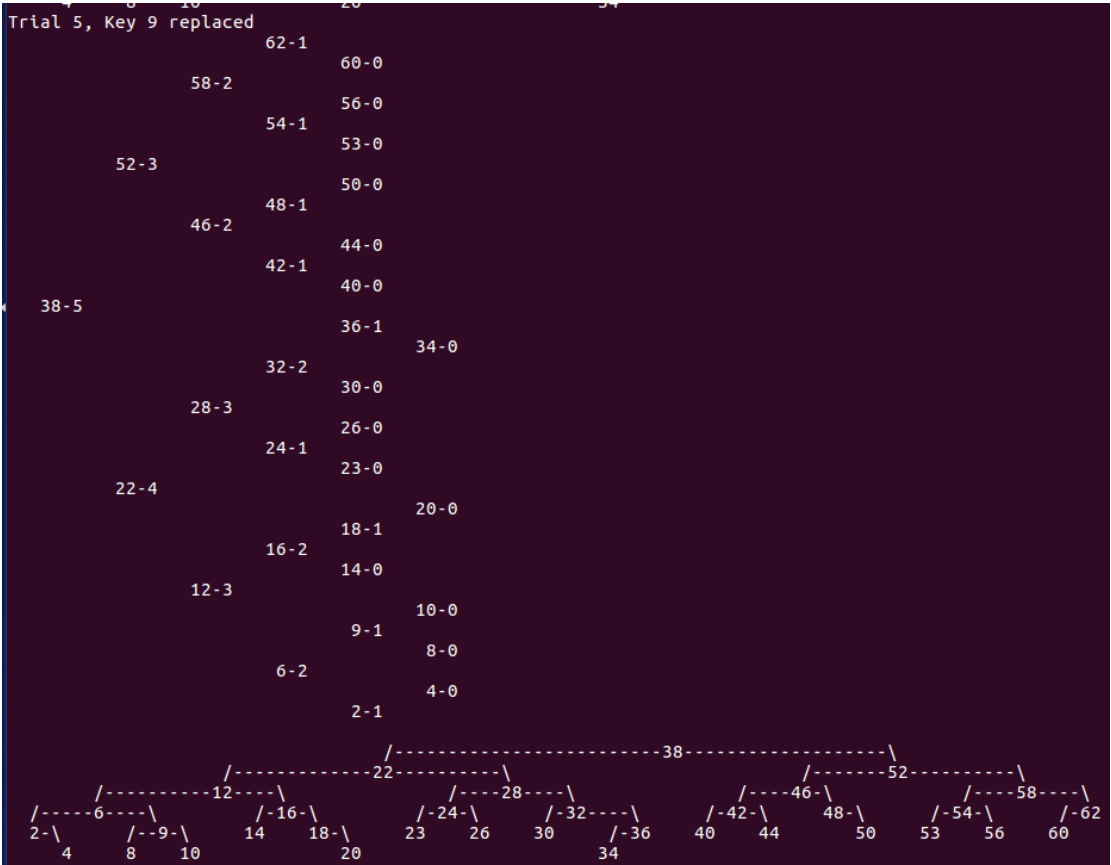
Trial 3, key 23 added



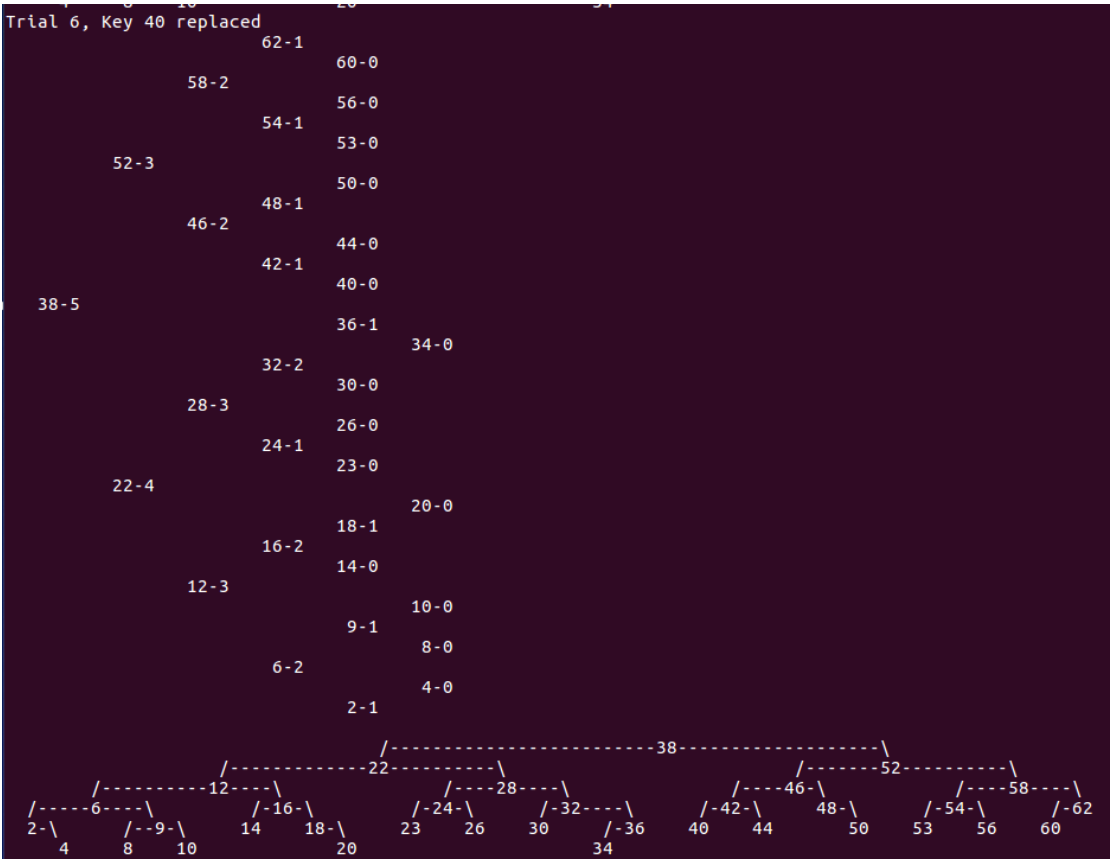
Trial 4, key 9 added



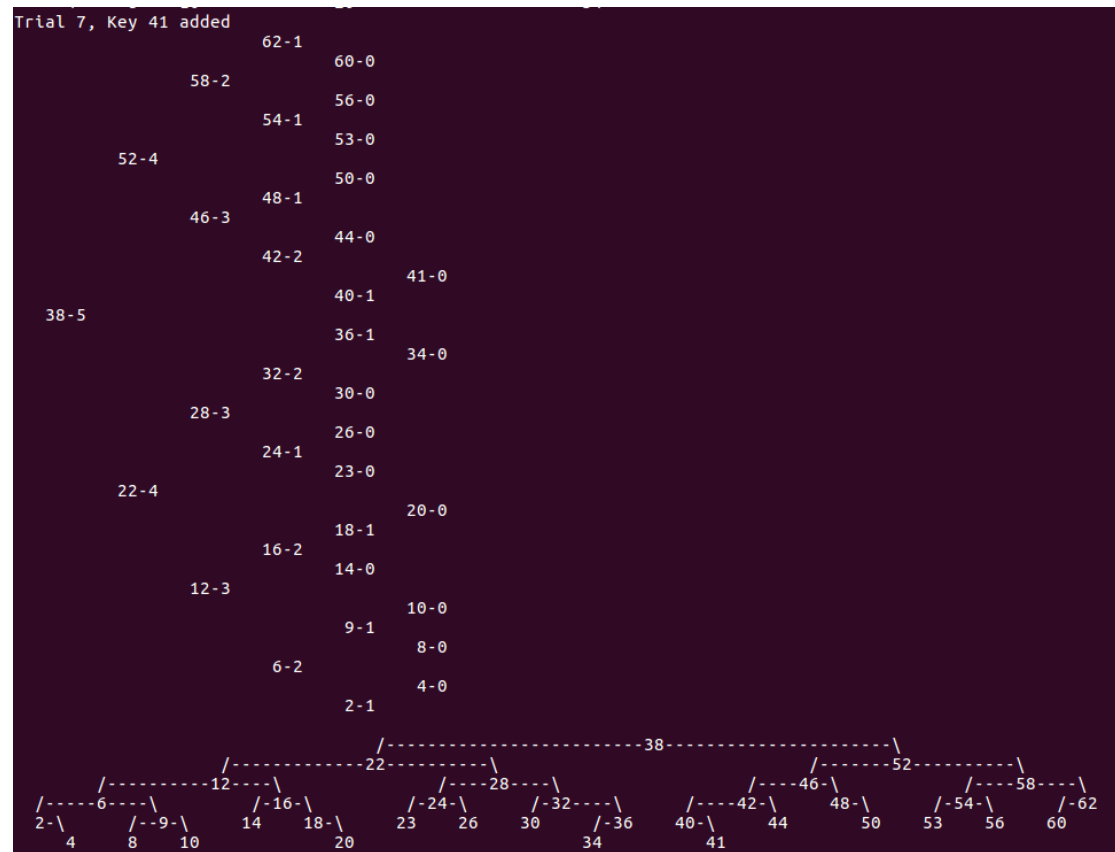
Trial 5, key 9 replaced



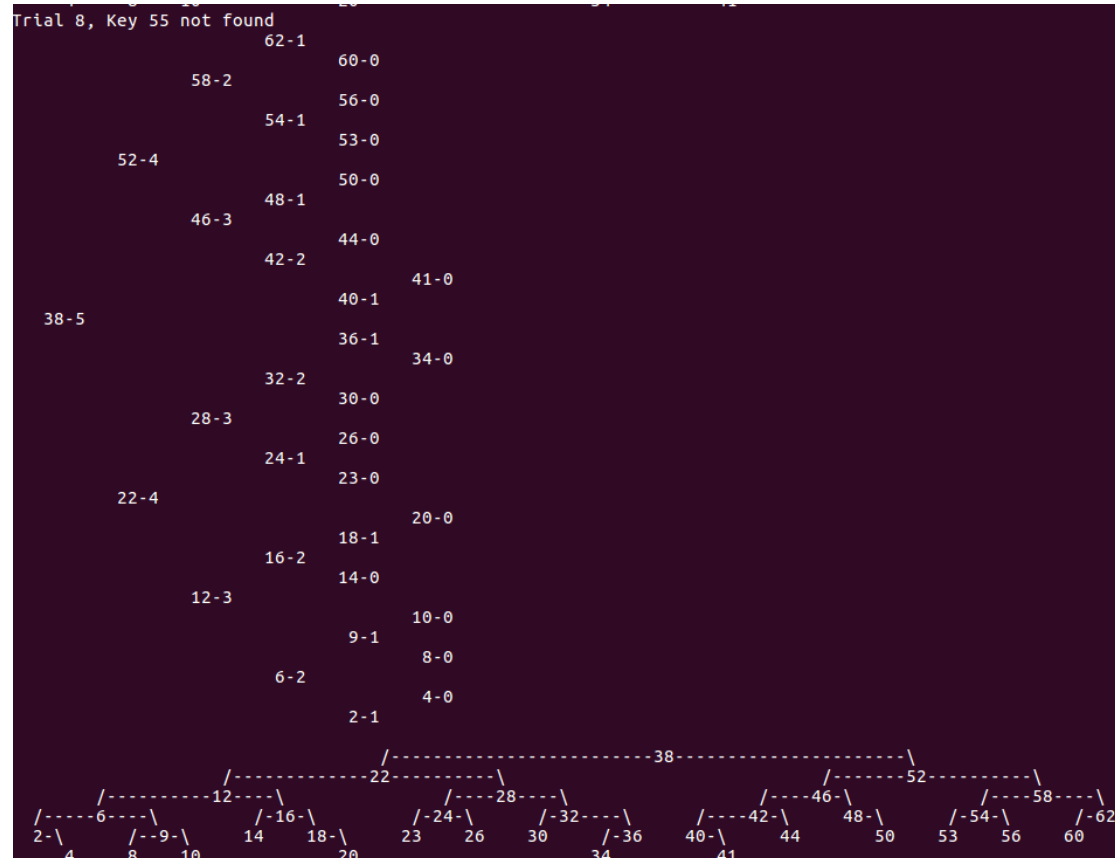
Trial 6, key 40 replaced



Trial 7, key 41 added



Trial 8, key 55 not found



Trial 9, Key 48 removed

```
Trial 9, Key 48 removed
      62-1
      60-0
      58-2
      56-0
      54-1
      53-0
      52-3
      50-0
      46-1
      44-0
      42-2
      41-0
      40-1
      38-5
      36-1
      34-0
      32-2
      30-0
      28-3
      26-0
      24-1
      23-0
      22-4
      20-0
      18-1
      16-2
      14-0
      12-3
      10-0
      9-1
      8-0
      6-2
      4-0
      2-1

      /-----38-----\
      /-----22-----\
      /-----12-----\
      /-----6-----\
      /-----9-----\
      /-----16-----\
      /-----24-----\
      /-----32-----\
      /-----40-----\
      /-----42-----\
      /-----46-----\
      /-----54-----\
      /-----58-----\
      /-----62-----\
      2-\ 4 8 10 14 18-\ 23 26 30 34 41 44 50 53 56 60
      2-\ 4 8 10 14 18-\ 23 26 30 34 41 44 50 53 56 60

After exercise, time=1.964, new tree size=34
successful searches during exercise=4.4, trials=5
unsuccessful searches during exercise=5, trials=5
Validating tree...passed
After access experiment, time=0.004, tree size=34
Expect successful search=7.76471, measured=4, trials=4
Expect unsuccessful search=10.4571, measured=4, trials=6
----- End of equilibrium test -----
```

Bonus: leak test!

```
After exercise, time=17.476, new tree size=34
successful searches during exercise=4.4, trials=5
unsuccessful searches during exercise=5, trials=5
Validating tree...passed
After access experiment, time=1.579, tree size=34
Expect successful search=7.76471, measured=4, trials=4
Expect unsuccessful search=10.4571, measured=4, trials=6
----- End of equilibrium test -----

==8598==
==8598== HEAP SUMMARY:
==8598==    in use at exit: 0 bytes in 0 blocks
==8598== total heap usage: 77 allocs, 77 frees, 2,448 bytes allocated
==8598== All heap blocks were freed -- no leaks are possible
==8598== For counts of detected and suppressed errors, rerun with: -v
==8598== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

/*<-----THE END OF TEST LOG FOR MP5 AND REPLACEMENT MP----->*/