Mandatory exercise #1

Štěpán Martínek

Project is using CMake for easy solution/makefile generation. Test code is then present in main files.

Binary search tree

Elements inserted into tree: 5, 3 ,4, 7, 6, 8
Tree:
```
     5
   /   \
  3     7
   \   / \
    4 6   8
```

Complexity of 4-8: All of these methods are written recursively going through every node so time complexity is O(N). We could change it to O(1) by caching each of those values, for a price of little bit of memory.

Run:
```
PREORDER
5
3
 4
7
 6
 8

--------------

INORDER
3
 4
5
 6
7
 8

--------------

POSTORDER
 4
3
 6
 8
7
5

--------------

LEVELORDER
5
```

```
3 7
4 6 8

--------------
Nr of nodes: 6
Nr of full nodes: 2
Nr of leafs: 3
Internal path length: 8
```

Bins & balls

Answers:

1. In experiments (test nr1. without power of two) the highest number of balls in one bin is 10, while average number of balls in bin climbing up to 7 with increased test sample.
2. O(log(n) / log(log(n))
3. In experiments (test nr2. without power of two) the highest number of balls in one bin is 10. So according to experiment bins should have space for at least 10 balls. Theoretical maximum number of balls in one bin is N, but chance of that happening is 1/N^N.
4.
   a. 1] 4 (avg ~3)
   b. 2] ln(ln(n)) / ln(2))
   c. 3] At least 4 balls into bin, but again to be sure answer should be N
   d. ln(ln(32768)) ~= 2.342, my experiments exceeded this number (i suppose logarithm is in base of e instead of 10 since log(log(32768)) ~= 0.6 which is nowhere near) so answer is NO

Experiments:

```
Test nr 1. Using power of two: false
Test sample: 1000
Avg max number: 6.705
Max number: 10
==============================
Test nr 1. Using power of two: true
Test sample: 1000
Avg max number: 3.056
Max number: 4
==============================
Test nr 2. Using power of two: false
Test sample: 1000
Avg max number: 7.265
Max number: 10
==============================
Test nr 2. Using power of two: true
Test sample: 1000
Avg max number: 3.182
Max number: 4
==============================
Test nr 1. Using power of two: false
Test sample: 100
```

```
Avg max number: 6.59
Max number: 9
===============================
Test nr 1. Using power of two: true
Test sample: 100
Avg max number: 3.05
Max number: 4
===============================
Test nr 2. Using power of two: false
Test sample: 100
Avg max number: 7.32
Max number: 10
===============================
Test nr 2. Using power of two: true
Test sample: 100
Avg max number: 3.22
Max number: 4
===============================
```