

binary search tree

1. tests passed :

The screenshot displays the PyCharm IDE interface. The top pane shows the source code of `test_bst.py`, featuring a `TestBST` class with methods `test_insertion`, `test_algorithms_comparison`, and `test_visualizer`. The bottom pane shows the test results for `TestBST`, indicating that all 7 tests passed successfully. The test results are as follows:

```
Run: Unittests for test_bst.TestBST
Tests passed: 7 of 7 tests - 600ms

Test Results 600ms
/Users/mac/Applications/python3/bin/python3 /Applications/PyCharm.app/Contents/plugins/python/helpers/pycharm/_jb_unittest_runner.py
Testing started at 6:03 PM ...
Launching unittests with arguments python -m unittest test_bst.TestBST in /Users/mac/Documents/GitHub/INF06205/Python/test/binary_search_tree

Process finished with exit code 0
hibbard 19 10.771484375
arbitrary-principle 18 10.525390625

Ran 7 tests in 0.605s
/Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/subprocess.py:942: ResourceWarning: subprocess 13049 is still running
warn("subprocess %s is still running" % self.pid,
ResourceWarning: Enable tracemalloc to get the object allocation traceback

def test_algorithms_comparison(self) -> None:
    instructions = make_random_bst_instructions(1024, 512)
    root_1 = execute_instructions(instructions, 'hibbard')
    root_2 = execute_instructions(instructions, 'arbitrary-principle')
    print('hibbard', depth(root_1), average_depth(root_1))
    print('arbitrary-principle', depth(root_2), average_depth(root_2))
    self.assertTrue(depth(root_2) <= depth(root_1))
    self.assertTrue(average_depth(root_2) <= average_depth(root_1))

def test_visualizer(self) -> None:
    instructions = make_random_bst_instructions(1024, 512)
    root = execute_instructions(instructions, 'hibbard')
    visualize_bt(root, converter)

TestBST > test_delete_key()

Test Results 896ms
/Users/mac/Applications/python3/bin/python3 /Applications/PyCharm.app/Contents/plugins/python/helpers/pycharm/_jb_unittest_runner.py
Testing started at 6:07 PM ...
Launching unittests with arguments python -m unittest test_bst.TestBST in /Users/mac/Documents/GitHub/INF06205/Python/test/binary_search_tree

Process finished with exit code 0
hibbard 18 9.75
arbitrary-principle 16 9.52734375

Ran 7 tests in 0.902s
/Library/Frameworks/Python.framework/Versions/3.8/lib/python3.8/subprocess.py:942: ResourceWarning: subprocess 13144 is still running
warn("subprocess %s is still running" % self.pid,
ResourceWarning: Enable tracemalloc to get the object allocation traceback
```

2. conclusion: I made a visualizer for our tests to figure out how a binary search tree after insertion and deletion looks like. The output graph is 'bst_test.gv.pdf' in test file. From the output data and the visualizer, we can figure out that Hibbard deletion is not always faster than arbitrary-principle deletion. The key point is that it depends on the order of insertion.