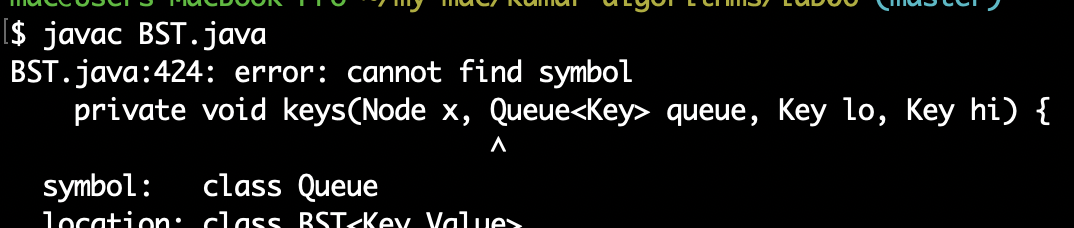
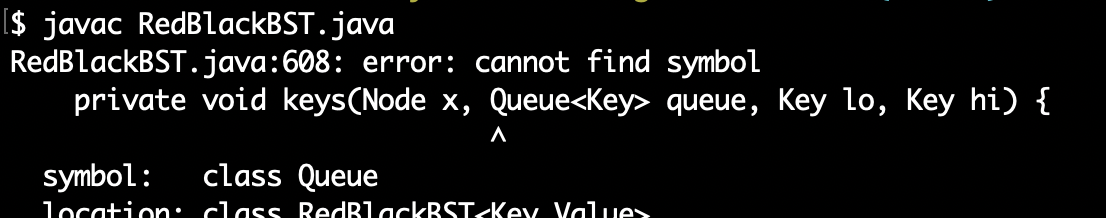
**Lab 6**

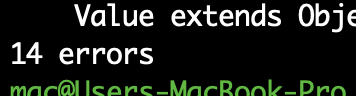
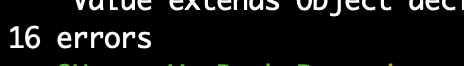
**Isa Dzhumabaev**

**1) Checkout the sample BST.java and RedBlackTree.java files**

So, I got a lot of error when I tried to compile these files.







And I decided to use C++ version from <https://www.geeksforgeeks.org/c-program-red-black-tree-insertion/> and modified main() function a little bit so it prints more information.

Here you can see some statistics on different inputs. It took a while to create them as the program runs for quite some time with 1 000 000 000 input.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 100 000 | 1 000 000 | 2 000 000 | 3 000 000 | 10 000 000 | 100 000 000 | 1 000 000 000 |
| Insertion | 0.032262 sec | 0.345542 sec | 0.709857 sec | 1.083931 sec | 3.780678 sec | 43.954579 sec | 7.8978819 min |
| Inorder Traversal | 0.001700 sec | 0.014995 sec | 0.029870 sec | 0.049232 sec | 0.169444 sec | 4.074239 sec | 58.810911 sec |
| Level Traversal | 0.011233 sec | 0.111430 sec | 0.225452 sec | 0.340764 sec | 1.126354 sec | 12.185512 sec | 12.298141 min |

**2) Perf report:**

After I did my benchmarks I decided not to profile this program with billion input as it would take a lot of time again.

You can find all the the reports in attached .data files.

**Files:**

million\_input.data

10\_million\_input.data

100\_million\_input.data

**3) Experiment with various input scenarios**

You can find results in attached “RedBlackBST Output.txt” file.

**4) Modified main() function used for time measurement:**

Remaining code is here:

<https://www.geeksforgeeks.org/c-program-red-black-tree-insertion/>

int main(int argc, char\*\* argv) {

int sz = -1;

printf("Enter size of array:**\n**");

scanf("%d", &sz);

int arr[sz];

srand(time(NULL));

**for** (int i = 0; i < sz; ++i)

{

arr[i] = rand() % (sz \* 20);

}

clock\_t begin = clock();

quickSort(arr, 0, sz - 1);

clock\_t end = clock();

printf("Time taken: %f seconds**\n**", ((double) (end - begin)) / CLOCKS\_PER\_SEC);

**return** 0;

}