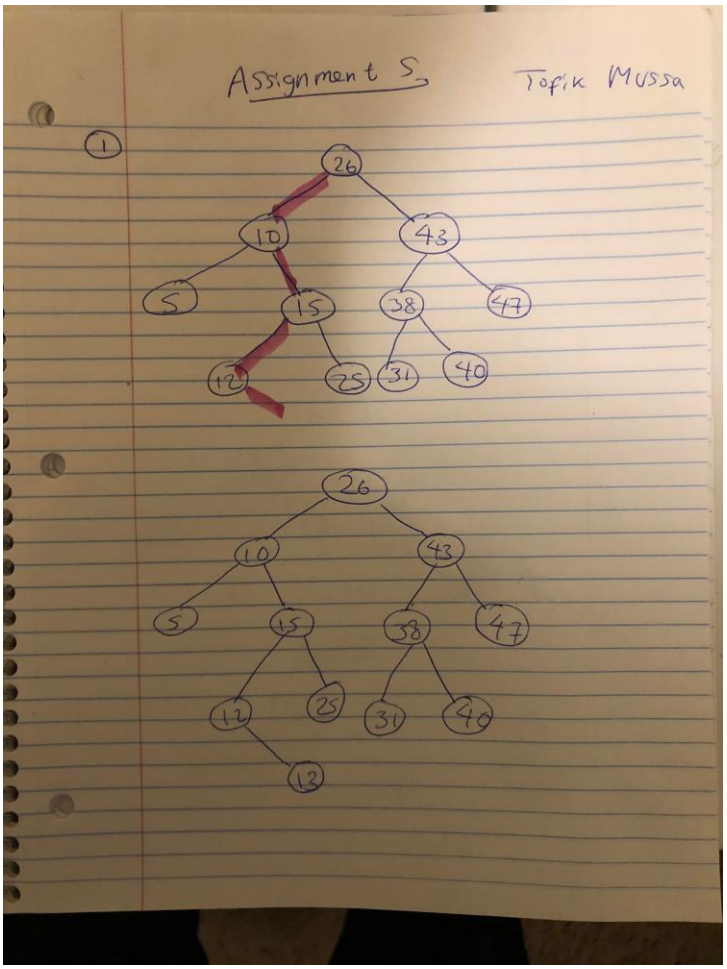
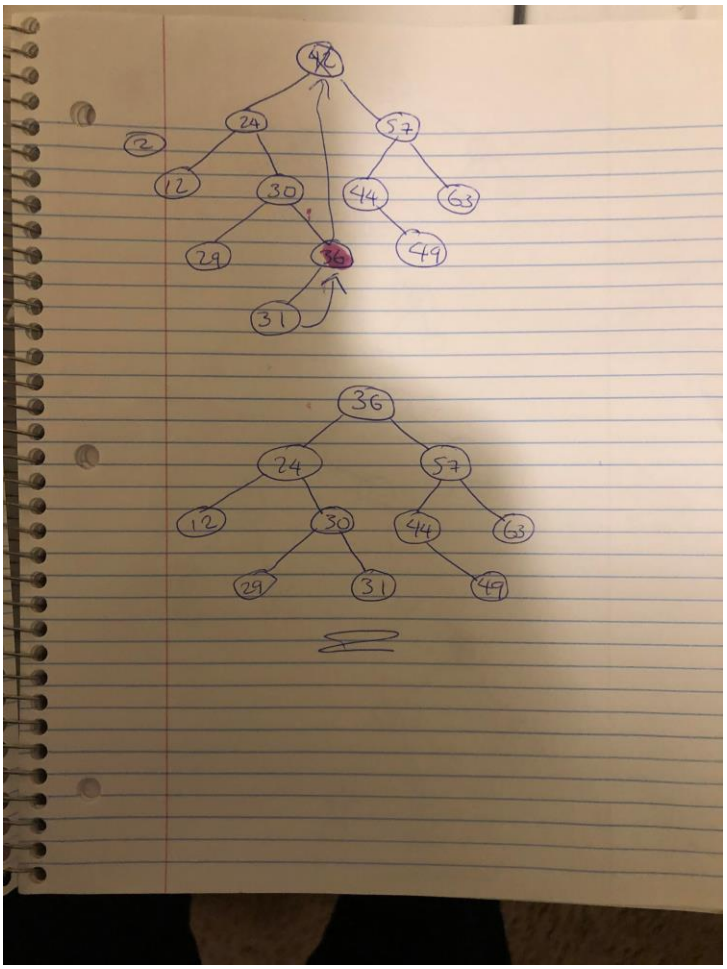
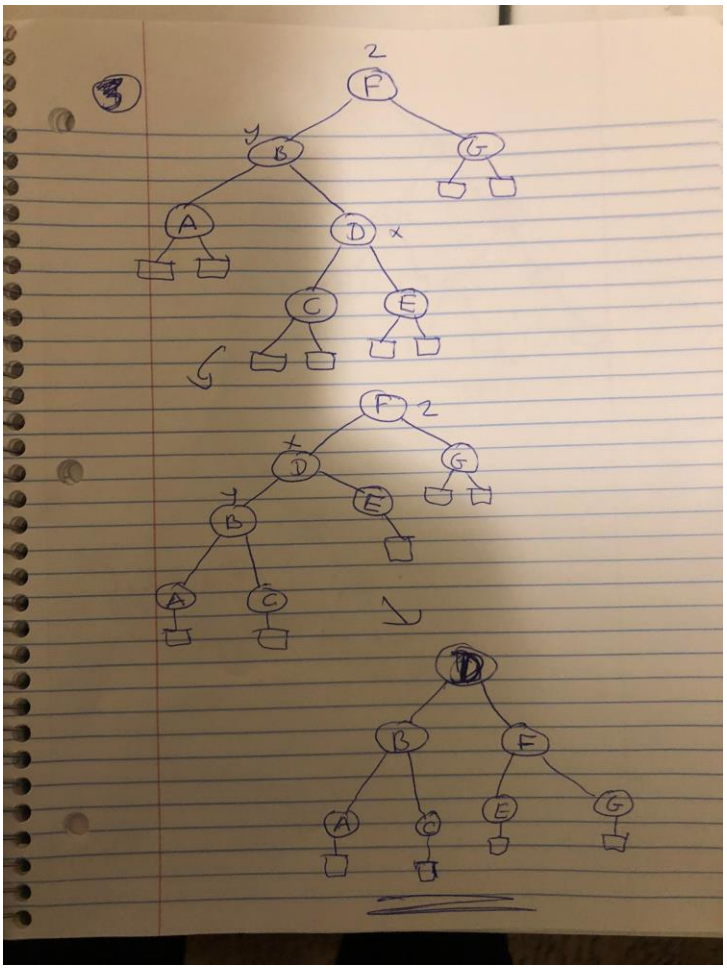
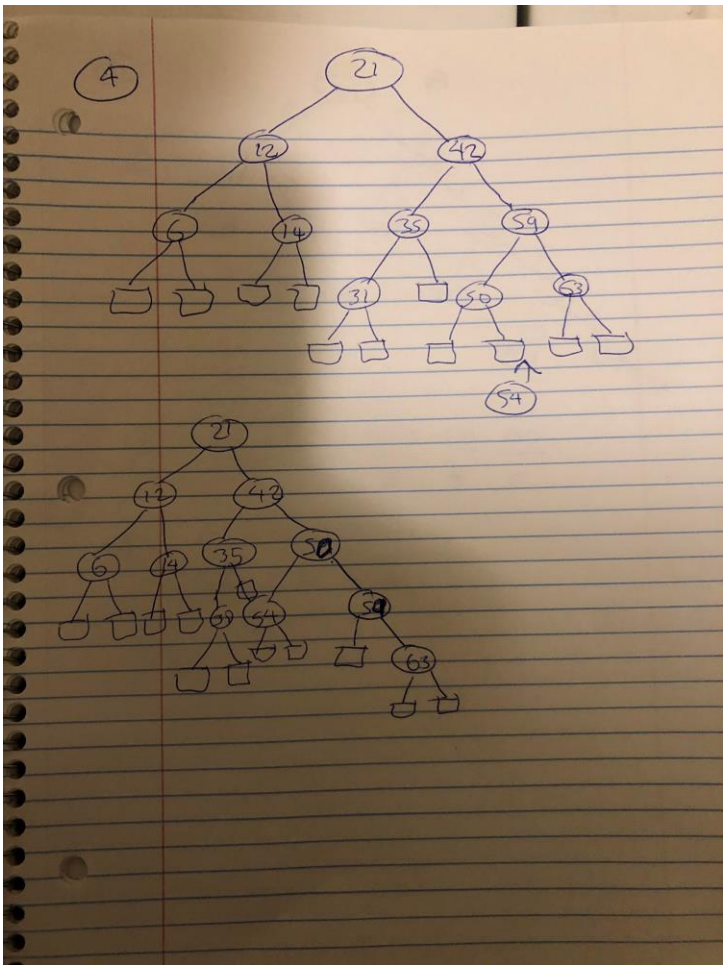
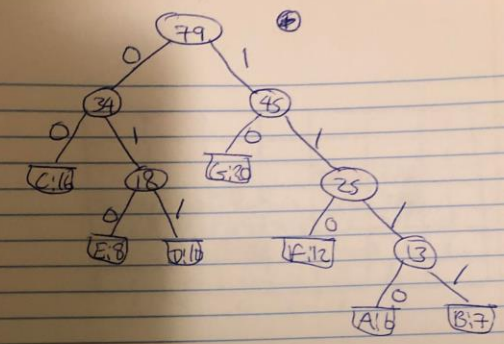

Assignment 5
Tofik Mussa
1
26
10
43
5
15
38
47
12
25
31
40
26
10
43
5
15
38
47
12
25
31
40
13


42
24
57
2
12
30
44
63
29
36
49
31
36
24
57
12
30
44
63
29
31
49


3
2
F
y
B
G
A
D
x
C
E
F
2
x
D
G
y
B
E
A
C
D
B
F
A
C
E
G


4
21
12
42
6
14
35
59
31
50
63
54
21
12
42
6
14
35
50
64
69
63

(5)

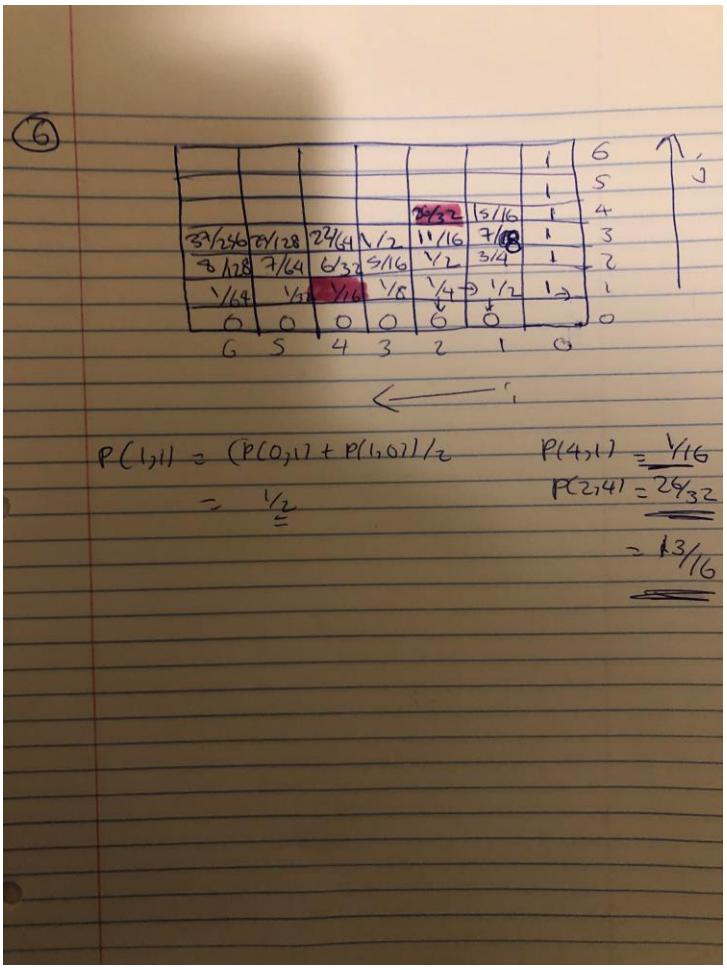


(1) 1111011010 10 00

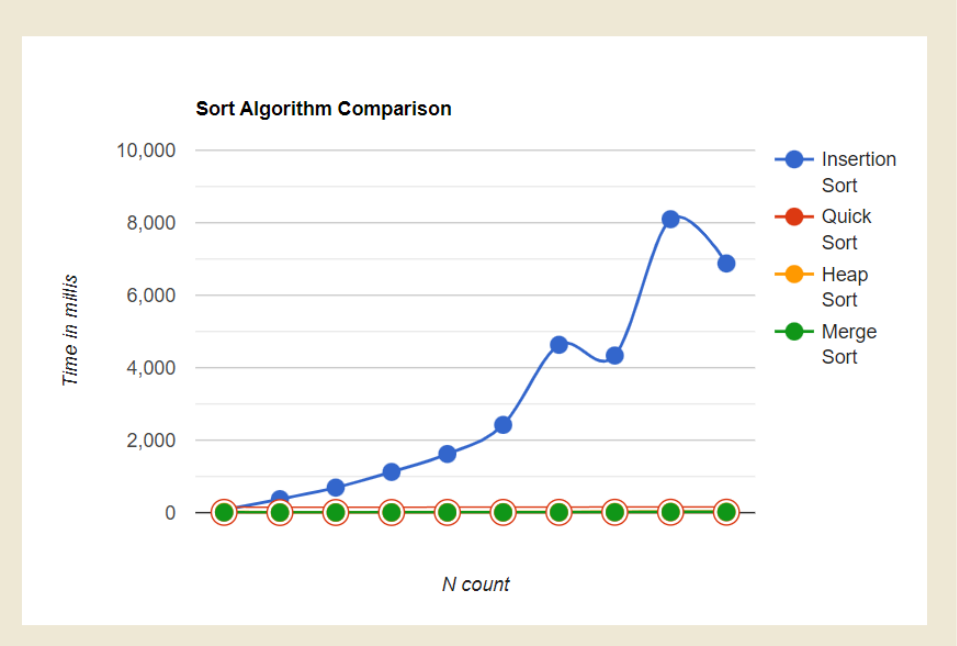
B D E G C

(2) 0101111010011

E A G D



Problem 7



The output

Running insertion sort on 10000 numbers

Running insertion sort on 10000 numbers took 74

Running merge sort on 10000 numbers

Running merge sort on 10000 numbers took 7

Running quick sort on 10000 numbers

Running quick sort on 10000 numbers took 8

Running heap sort on 10000 numbers

Running heap sort on 10000 numbers took 21

Running insertion sort on 20000 numbers

Running insertion sort on 20000 numbers took 372

Running merge sort on 20000 numbers

Running merge sort on 20000 numbers took 3

Running quick sort on 20000 numbers

Running quick sort on 20000 numbers took 2

Running heap sort on 20000 numbers

Running heap sort on 20000 numbers took 5

Running insertion sort on 30000 numbers

Running insertion sort on 30000 numbers took 691

Running merge sort on 30000 numbers

Running merge sort on 30000 numbers took 4

Running quick sort on 30000 numbers

Running quick sort on 30000 numbers took 4

Running heap sort on 30000 numbers

Running heap sort on 30000 numbers took 8

Running insertion sort on 40000 numbers

Running insertion sort on 40000 numbers took 1123

Running merge sort on 40000 numbers

Running merge sort on 40000 numbers took 6

Running quick sort on 40000 numbers

Running quick sort on 40000 numbers took 4

Running heap sort on 40000 numbers

Running heap sort on 40000 numbers took 9

Running insertion sort on 50000 numbers

Running insertion sort on 50000 numbers took 1618

Running merge sort on 50000 numbers

Running merge sort on 50000 numbers took 7

Running quick sort on 50000 numbers

Running quick sort on 50000 numbers took 6

Running heap sort on 50000 numbers

Running heap sort on 50000 numbers took 11

Running insertion sort on 60000 numbers

Running insertion sort on 60000 numbers took 2421

Running merge sort on 60000 numbers

Running merge sort on 60000 numbers took 8

Running quick sort on 60000 numbers

Running quick sort on 60000 numbers took 7

Running heap sort on 60000 numbers

Running heap sort on 60000 numbers took 13

Running insertion sort on 70000 numbers

Running insertion sort on 70000 numbers took 4631

Running merge sort on 70000 numbers

Running merge sort on 70000 numbers took 9

Running quick sort on 70000 numbers

Running quick sort on 70000 numbers took 8

Running heap sort on 70000 numbers

Running heap sort on 70000 numbers took 15

Running insertion sort on 80000 numbers

Running insertion sort on 80000 numbers took 4335

Running merge sort on 80000 numbers

Running merge sort on 80000 numbers took 11

Running quick sort on 80000 numbers

Running quick sort on 80000 numbers took 10

Running heap sort on 80000 numbers

Running heap sort on 80000 numbers took 19

Running insertion sort on 90000 numbers

Running insertion sort on 90000 numbers took 8097

Running merge sort on 90000 numbers

Running merge sort on 90000 numbers took 17

Running quick sort on 90000 numbers

Running quick sort on 90000 numbers took 12

Running heap sort on 90000 numbers

Running heap sort on 90000 numbers took 22

Running insertion sort on 100000 numbers

Running insertion sort on 100000 numbers took 6876

Running merge sort on 100000 numbers

Running merge sort on 100000 numbers took 18

Running quick sort on 100000 numbers

Running quick sort on 100000 numbers took 12

Running heap sort on 100000 numbers

Running heap sort on 100000 numbers took 24

Conclusion

Asymptotically, insertion sort runs match slower than the other 3 sorting algorithms. This might because the numbers are randomly generated requiring too many swaps. Insertion sort runs faster if the array was almost sorted