Problem 1b. Enter the Time and compute the Ratio of Times to two decimal places (x.xx)

Graph Size	Time for Computing Spanning Tree	Ratio of Time: Size 2N/Size N
1,000	0.084	No ratio for first graph size
2,000	0.214	2.55
4,000	0.472	2.21
8,000	0.921	1.95
16,000	2.136	2.32
32,000	5.112	2.39
64,000	10.423	2.04
128,000	21.377	2.05

Approximate the complexity class for the spanning_tree function based on the data above.

Answer: O(NLogN)

Problem 2b. Answer each of the following question based on the profiles produced when running spanning_tree : the 5,000 node random graph sorted by ncalls for parts 2 and 3; the 10,000 node random graph sorted by tottime for parts 1 and 4.

1) What function/method takes the most tottime to execute?

Answer: sorted

2) What non-built in function/method is called the most times?

Answer: __getitem__

3) What method defined in graph.py is called the most times?

Answer: __getitem__

4) What percent of the entire execution time is spent in the 5 functions with the most tottime?

Answer: 76%