Sample output from my solution to Problem #1:
(yours should match the format: the times depend on your machine’s speed).

```
Spanning Tree of size 1000
Analysis of 5 timings
avg = 0.08043   min = 0.07824  max = 0.08163  span = 4.2%

   Time Ranges
7.82e-02<>7.86e-02[ 20.0%]|**************************************************
7.86e-02<>7.89e-02[  0.0%]|
7.89e-02<>7.93e-02[  0.0%]|
7.93e-02<>7.96e-02[  0.0%]|
7.96e-02<>7.99e-02[ 20.0%]|**************************************************
7.99e-02<>8.03e-02[  0.0%]|
8.03e-02<>8.06e-02[  0.0%]|A
8.06e-02<>8.10e-02[  0.0%]|
8.10e-02<>8.13e-02[ 20.0%]|**************************************************
8.13e-02<>8.16e-02[ 20.0%]|**************************************************
8.16e-02<>8.20e-02[ 20.0%]|**************************************************

Spanning Tree of size 2000
Analysis of 5 timings
avg = 0.17235   min = 0.16886  max = 0.17738  span = 4.9%

   Time Ranges
1.69e-01<>1.70e-01[ 40.0%]|**************************************************
1.70e-01<>1.71e-01[  0.0%]|
1.71e-01<>1.71e-01[  0.0%]|
1.71e-01<>1.72e-01[ 20.0%]|*************************
1.72e-01<>1.73e-01[  0.0%]|A
1.73e-01<>1.74e-01[ 20.0%]|*************************
1.74e-01<>1.75e-01[  0.0%]|
1.75e-01<>1.76e-01[  0.0%]|
1.76e-01<>1.77e-01[  0.0%]|
1.77e-01<>1.77e-01[  0.0%]|
1.77e-01<>1.78e-01[ 20.0%]|*************************

Spanning Tree of size 4000
Analysis of 5 timings
avg = 0.35948   min = 0.35017  max = 0.37024  span = 5.6%

   Time Ranges
3.50e-01<>3.52e-01[ 20.0%]|**************************************************
3.52e-01<>3.54e-01[ 20.0%]|**************************************************
3.54e-01<>3.56e-01[  0.0%]|
3.56e-01<>3.58e-01[  0.0%]|
3.58e-01<>3.60e-01[  0.0%]|A
3.60e-01<>3.62e-01[ 20.0%]|**************************************************
3.62e-01<>3.64e-01[ 20.0%]|**************************************************
3.64e-01<>3.66e-01[  0.0%]|
3.66e-01<>3.68e-01[  0.0%]|
3.68e-01<>3.70e-01[  0.0%]|
3.70e-01<>3.72e-01[ 20.0%]|**************************************************

Spanning Tree of size 8000
Analysis of 5 timings
avg = 0.77867   min = 0.74521  max = 0.81487  span = 8.9%

   Time Ranges
7.45e-01<>7.52e-01[ 20.0%]|**************************************************
```

```
7.52e-01<>7.59e-01[  0.0%]|
7.59e-01<>7.66e-01[ 20.0%]|**************************************************
7.66e-01<>7.73e-01[ 20.0%]|**************************************************
7.73e-01<>7.80e-01[  0.0%]|A
7.80e-01<>7.87e-01[  0.0%]|
7.87e-01<>7.94e-01[  0.0%]|
7.94e-01<>8.01e-01[ 20.0%]|**************************************************
8.01e-01<>8.08e-01[  0.0%]|
8.08e-01<>8.15e-01[  0.0%]|
8.15e-01<>8.22e-01[ 20.0%]|**************************************************

Spanning Tree of size 16000
Analysis of 5 timings
avg = 1.66178   min = 1.61219  max = 1.72383  span = 6.7%

   Time Ranges
1.61e+00<>1.62e+00[ 20.0%]|*************************
1.62e+00<>1.63e+00[  0.0%]|
1.63e+00<>1.65e+00[ 20.0%]|*************************
1.65e+00<>1.66e+00[  0.0%]|
1.66e+00<>1.67e+00[ 40.0%]|**************************************************A
1.67e+00<>1.68e+00[  0.0%]|
1.68e+00<>1.69e+00[  0.0%]|
1.69e+00<>1.70e+00[  0.0%]|
1.70e+00<>1.71e+00[  0.0%]|
1.71e+00<>1.72e+00[  0.0%]|
1.72e+00<>1.73e+00[ 20.0%]|*************************

Spanning Tree of size 32000
Analysis of 5 timings
avg = 3.75091   min = 3.57036  max = 3.99897  span = 11.4%

   Time Ranges
3.57e+00<>3.61e+00[ 20.0%]|*************************
3.61e+00<>3.66e+00[  0.0%]|
3.66e+00<>3.70e+00[  0.0%]|
3.70e+00<>3.74e+00[ 40.0%]|**************************************************
3.74e+00<>3.78e+00[ 20.0%]|*************************A
3.78e+00<>3.83e+00[  0.0%]|
3.83e+00<>3.87e+00[  0.0%]|
3.87e+00<>3.91e+00[  0.0%]|
3.91e+00<>3.96e+00[  0.0%]|
3.96e+00<>4.00e+00[  0.0%]|
4.00e+00<>4.04e+00[ 20.0%]|*************************

Spanning Tree of size 64000
Analysis of 5 timings
avg = 7.85534   min = 7.75109  max = 8.00142  span = 3.2%

   Time Ranges
7.75e+00<>7.78e+00[ 20.0%]|**************************************************
7.78e+00<>7.80e+00[ 20.0%]|**************************************************
7.80e+00<>7.83e+00[  0.0%]|
7.83e+00<>7.85e+00[ 20.0%]|**************************************************
7.85e+00<>7.88e+00[  0.0%]|A
7.88e+00<>7.90e+00[ 20.0%]|**************************************************
7.90e+00<>7.93e+00[  0.0%]|
7.93e+00<>7.95e+00[  0.0%]|
7.95e+00<>7.98e+00[  0.0%]|
7.98e+00<>8.00e+00[  0.0%]|
8.00e+00<>8.03e+00[ 20.0%]|**************************************************
```

```
Spanning Tree of size 128000
Analysis of 5 timings
avg = 17.39035   min = 16.93979  max = 17.89083  span = 5.5%

   Time Ranges
1.69e+01<>1.70e+01[ 20.0%]|*************************
1.70e+01<>1.71e+01[  0.0%]|
1.71e+01<>1.72e+01[  0.0%]|
1.72e+01<>1.73e+01[ 40.0%]|**************************************************
1.73e+01<>1.74e+01[  0.0%]|A
1.74e+01<>1.75e+01[  0.0%]|
1.75e+01<>1.76e+01[  0.0%]|
1.76e+01<>1.77e+01[ 20.0%]|*************************
1.77e+01<>1.78e+01[  0.0%]|
1.78e+01<>1.79e+01[  0.0%]|
1.79e+01<>1.80e+01[ 20.0%]|*************************
```

Sample output from my solution to Problem #2:
(yours should match the format: the times/counts depend on your machine’s speed and the random graph created).

```
TFri Mar  9 09:14:51 2018    profile5K

         767850 function calls (762849 primitive calls) in 0.559 seconds

   Ordered by: call count

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
   199557    0.011    0.000    0.011    0.000 {built-in method builtins.len}
   104778    0.090    0.000    0.167    0.000 graph.py:23(__getitem__)
    99779    0.051    0.000    0.303    0.000 graph_goody.py:26(<genexpr>)
    99779    0.060    0.000    0.247    0.000 graph.py:125(__iter__)
    99778    0.072    0.000    0.077    0.000 graph.py:12(legal_tuple)
    99446    0.068    0.000    0.068    0.000 equivalence.py:28(_compress_to_root)
    44724    0.021    0.000    0.082    0.000 equivalence.py:60(in_same_class)
   5002/1    0.148    0.000    0.431    0.431 {built-in method builtins.sorted}
     5000    0.002    0.000    0.002    0.000 equivalence.py:19(add_singleton)
     4999    0.006    0.000    0.012    0.000 equivalence.py:68(merge_classes_containing)
     4999    0.001    0.000    0.001    0.000 {method 'add' of 'set' objects}
        2    0.000    0.000    0.000    0.000 graph.py:73(all_nodes)
        2    0.000    0.000    0.000    0.000 {method 'keys' of 'dict' objects}
        1    0.023    0.023    0.551    0.551 graph_goody.py:24(spanning_tree)
        1    0.001    0.001    0.003    0.003 equivalence.py:8(__init__)
        1    0.007    0.007    0.559    0.559 <string>:1(<module>)
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
        1    0.000    0.000    0.559    0.559 {built-in method builtins.exec}


Fri Mar  9 09:14:54 2018    profile10K

         1613285 function calls (1603284 primitive calls) in 1.213 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
  10002/1    0.323    0.000    0.889    0.889 {built-in method builtins.sorted}
   209780    0.184    0.000    0.338    0.000 graph.py:23(__getitem__)
   249728    0.171    0.000    0.171    0.000 equivalence.py:28(_compress_to_root)
   199780    0.143    0.000    0.154    0.000 graph.py:12(legal_tuple)
   199781    0.116    0.000    0.493    0.000 graph.py:125(__iter__)
   199781    0.102    0.000    0.605    0.000 graph_goody.py:26(<genexpr>)
        1    0.060    0.060    1.192    1.192 graph_goody.py:24(spanning_tree)
   114865    0.053    0.000    0.212    0.000 equivalence.py:60(in_same_class)
   399561    0.021    0.000    0.021    0.000 {built-in method builtins.len}
        1    0.020    0.020    1.212    1.212 <string>:1(<module>)
     9999    0.011    0.000    0.023    0.000 equivalence.py:68(merge_classes_containing)
    10000    0.004    0.000    0.004    0.000 equivalence.py:19(add_singleton)
        1    0.002    0.002    0.006    0.006 equivalence.py:8(__init__)
     9999    0.001    0.000    0.001    0.000 {method 'add' of 'set' objects}
        2    0.001    0.000    0.001    0.000 graph.py:73(all_nodes)
        1    0.000    0.000    1.213    1.213 {built-in method builtins.exec}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
        2    0.000    0.000    0.000    0.000 {method 'keys' of 'dict' objects}
```