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 = 0.079   min = 0.076  max = 0.084  span = 9.7%

   Time Ranges
7.59e-02<>7.66e-02[ 40.0%]|**************************************************
7.66e-02<>7.74e-02[  0.0%]|
7.74e-02<>7.82e-02[  0.0%]|
7.82e-02<>7.89e-02[  0.0%]|
7.89e-02<>7.97e-02[ 20.0%]|*************************A
7.97e-02<>8.05e-02[  0.0%]|
8.05e-02<>8.12e-02[ 20.0%]|*************************
8.12e-02<>8.20e-02[  0.0%]|
8.20e-02<>8.28e-02[  0.0%]|
8.28e-02<>8.35e-02[  0.0%]|
8.35e-02<>8.43e-02[ 20.0%]|*************************

Spanning Tree of size 2000
Analysis of 5 timings
avg = 0.166   min = 0.161  max = 0.172  span = 6.1%

   Time Ranges
1.61e-01<>1.62e-01[ 20.0%]|*************************
1.62e-01<>1.63e-01[  0.0%]|
1.63e-01<>1.64e-01[  0.0%]|
1.64e-01<>1.65e-01[ 40.0%]|**************************************************
1.65e-01<>1.66e-01[  0.0%]|A
1.66e-01<>1.68e-01[  0.0%]|
1.68e-01<>1.69e-01[ 20.0%]|*************************
1.69e-01<>1.70e-01[  0.0%]|
1.70e-01<>1.71e-01[  0.0%]|
1.71e-01<>1.72e-01[  0.0%]|
1.72e-01<>1.73e-01[ 20.0%]|*************************

Spanning Tree of size 4000
Analysis of 5 timings
avg = 0.351   min = 0.342  max = 0.359  span = 4.9%

   Time Ranges
3.42e-01<>3.44e-01[ 20.0%]|**************************************************
3.44e-01<>3.45e-01[  0.0%]|
3.45e-01<>3.47e-01[  0.0%]|
3.47e-01<>3.49e-01[ 20.0%]|**************************************************
3.49e-01<>3.50e-01[ 20.0%]|**************************************************
3.50e-01<>3.52e-01[  0.0%]|A
3.52e-01<>3.54e-01[  0.0%]|
3.54e-01<>3.55e-01[ 20.0%]|**************************************************
3.55e-01<>3.57e-01[  0.0%]|
3.57e-01<>3.59e-01[  0.0%]|
3.59e-01<>3.61e-01[ 20.0%]|**************************************************

Spanning Tree of size 8000
Analysis of 5 timings
avg = 0.733   min = 0.723  max = 0.749  span = 3.5%

   Time Ranges
7.23e-01<>7.26e-01[ 40.0%]|**************************************************
```

```
7.26e-01<>7.28e-01[  0.0%]|
7.28e-01<>7.31e-01[ 20.0%]|*************************
7.31e-01<>7.34e-01[  0.0%]|A
7.34e-01<>7.36e-01[  0.0%]|
7.36e-01<>7.39e-01[  0.0%]|
7.39e-01<>7.41e-01[ 20.0%]|*************************
7.41e-01<>7.44e-01[  0.0%]|
7.44e-01<>7.46e-01[  0.0%]|
7.46e-01<>7.49e-01[  0.0%]|
7.49e-01<>7.51e-01[ 20.0%]|*************************

Spanning Tree of size 16000
Analysis of 5 timings
avg = 1.582   min = 1.544  max = 1.642  span = 6.2%

   Time Ranges
1.54e+00<>1.55e+00[ 20.0%]|*************************
1.55e+00<>1.56e+00[  0.0%]|
1.56e+00<>1.57e+00[ 40.0%]|**************************************************
1.57e+00<>1.58e+00[  0.0%]|A
1.58e+00<>1.59e+00[ 20.0%]|*************************
1.59e+00<>1.60e+00[  0.0%]|
1.60e+00<>1.61e+00[  0.0%]|
1.61e+00<>1.62e+00[  0.0%]|
1.62e+00<>1.63e+00[  0.0%]|
1.63e+00<>1.64e+00[  0.0%]|
1.64e+00<>1.65e+00[ 20.0%]|*************************

Spanning Tree of size 32000
Analysis of 5 timings
avg = 3.432   min = 3.377  max = 3.514  span = 4.0%

   Time Ranges
3.38e+00<>3.39e+00[ 20.0%]|**************************************************
3.39e+00<>3.40e+00[ 20.0%]|**************************************************
3.40e+00<>3.42e+00[ 20.0%]|**************************************************
3.42e+00<>3.43e+00[  0.0%]|A
3.43e+00<>3.45e+00[  0.0%]|
3.45e+00<>3.46e+00[  0.0%]|
3.46e+00<>3.47e+00[ 20.0%]|**************************************************
3.47e+00<>3.49e+00[  0.0%]|
3.49e+00<>3.50e+00[  0.0%]|
3.50e+00<>3.51e+00[  0.0%]|
3.51e+00<>3.53e+00[ 20.0%]|**************************************************

Spanning Tree of size 64000
Analysis of 5 timings
avg = 7.470   min = 7.358  max = 7.680  span = 4.3%

   Time Ranges
7.36e+00<>7.39e+00[ 40.0%]|**************************************************
7.39e+00<>7.42e+00[  0.0%]|
7.42e+00<>7.45e+00[  0.0%]|
7.45e+00<>7.49e+00[ 40.0%]|**************************************************A
7.49e+00<>7.52e+00[  0.0%]|
7.52e+00<>7.55e+00[  0.0%]|
7.55e+00<>7.58e+00[  0.0%]|
7.58e+00<>7.62e+00[  0.0%]|
7.62e+00<>7.65e+00[  0.0%]|
7.65e+00<>7.68e+00[  0.0%]|
7.68e+00<>7.71e+00[ 20.0%]|*************************
```

```
Spanning Tree of size 128000
Analysis of 5 timings
avg = 16.294   min = 16.071  max = 16.665  span = 3.6%

   Time Ranges
1.61e+01<>1.61e+01[ 20.0%]|**************************************************
1.61e+01<>1.62e+01[ 20.0%]|**************************************************
1.62e+01<>1.62e+01[ 20.0%]|**************************************************
1.62e+01<>1.63e+01[  0.0%]|A
1.63e+01<>1.64e+01[  0.0%]|
1.64e+01<>1.64e+01[ 20.0%]|**************************************************
1.64e+01<>1.65e+01[  0.0%]|
1.65e+01<>1.65e+01[  0.0%]|
1.65e+01<>1.66e+01[  0.0%]|
1.66e+01<>1.67e+01[  0.0%]|
1.67e+01<>1.67e+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).

```
Fri May 29 20:16:10 2015    profile50K

         8007108 function calls (7957107 primitive calls) in 6.638 seconds

   Ordered by: call count

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
  1999509    0.084    0.000    0.084    0.000 {built-in method len}
  1205714    0.831    0.000    0.831    0.000 equivalence.py:28(_compress_to_root)
  1049754    0.885    0.000    1.586    0.000 graph.py:23(__getitem__)
   999755    0.495    0.000    2.969    0.000 graph_goody.py:27(<genexpr>)
   999755    0.639    0.000    2.432    0.000 graph.py:125(__iter__)
   999754    0.659    0.000    0.701    0.000 graph.py:12(legal_tuple)
   552858    0.251    0.000    1.024    0.000 equivalence.py:60(in_same_class)
  50002/1    2.199    0.000    4.962    4.962 {built-in method sorted}
    50000    0.018    0.000    0.018    0.000 equivalence.py:19(add_singleton)
    49999    0.008    0.000    0.008    0.000 {method 'add' of 'set' objects}
    49999    0.056    0.000    0.114    0.000 equivalence.py:68(merge_classes_containing)
        2    0.007    0.003    0.007    0.003 graph.py:73(all_nodes)
        2    0.000    0.000    0.000    0.000 {method 'keys' of 'dict' objects}
        1    0.325    0.325    6.468    6.468 graph_goody.py:25(spanning_tree)
        1    0.000    0.000    6.638    6.638 {built-in method exec}
        1    0.011    0.011    0.029    0.029 equivalence.py:8(__init__)
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
        1    0.170    0.170    6.638    6.638 <string>:1(<module>)


Fri May 29 20:16:34 2015    profile100K

         16718291 function calls (16618290 primitive calls) in 14.579 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
 100002/1    5.196    0.000   10.749   10.749 {built-in method sorted}
  2879692    1.975    0.000    1.975    0.000 equivalence.py:28(_compress_to_root)
  2099790    1.787    0.000    3.195    0.000 graph.py:23(__getitem__)
  1999790    1.321    0.000    1.408    0.000 graph.py:12(legal_tuple)
  1999791    1.278    0.000    4.894    0.000 graph.py:125(__iter__)
  1999791    0.996    0.000    5.975    0.000 graph_goody.py:27(<genexpr>)
        1    0.758    0.758   14.300   14.300 graph_goody.py:25(spanning_tree)
  1339847    0.598    0.000    2.457    0.000 equivalence.py:60(in_same_class)
        1    0.279    0.279   14.579   14.579 <string>:1(<module>)
  3999581    0.172    0.000    0.172    0.000 {built-in method len}
    99999    0.116    0.000    0.232    0.000 equivalence.py:68(merge_classes_containing)
   100000    0.039    0.000    0.039    0.000 equivalence.py:19(add_singleton)
        1    0.026    0.026    0.065    0.065 equivalence.py:8(__init__)
        2    0.021    0.011    0.021    0.011 graph.py:73(all_nodes)
    99999    0.018    0.000    0.018    0.000 {method 'add' of 'set' objects}
        1    0.000    0.000   14.579   14.579 {built-in method exec}
        2    0.000    0.000    0.000    0.000 {method 'keys' of 'dict' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
```