Distribution of distances in our graph: {1: 176468, 2: 2716134, 3: 3981852, 5: 2565170, 6: 677214, 4: 5861560, 7: 315464, 8: 15620}

Mode is 4, Median is 4, Diameter is 8. Running time to compute diameter is ~ 170 seconds, or around 3 minutes  
  
# Does the data exhibit the small world phenomenom?

Yes, there is a maximum of 8 degrees of separation for any node within the graph, with a median of 4 degrees of separation. A network with the small-world property is characterized by small path lengths.  
  
8) Each run of BFS takes theta(n+m) time where n is nodes and m is edges. Running BFS for all nodes in the graph takes theta(n(n+m)).

T(n) in theta(n^2)  
  
Runtime for canvas for 4039 nodes and 88234 edges:

4039(4039 + 88234) = 372690647 size of input, which takes 170 seconds  
  
Runtime to calculate diameter of a Facebook graph with 2 million nodes and 44 million edges:  
2mil(2mil+44mil) = 92,000,000,000,000 size of input (12 zeros).  
  
f(n) = an^2 + bnm

The 2nd term, bnm is irrelevant with large values of n so we will drop it   
  
170 seconds = a \* 372690647

So my computer constant is: a = 4.5614 x10^-7  
  
Therefore f(n) = 4.5614 x10^-7 \* n^2  
  
x seconds = 4.5614 x10^-7 \* 92000000000000 = 4.19 x10^7 seconds  
Divide by 60 to get minutes = 699418 minutes  
Divide by 60 to get hours = 11657 hours  
Divide by 24 to get days = 485.7 days  
  
Or it would take approximately 486 days to run BFS on the Facebook graph example.  
  
#What would you do to compute the diameter of this large graph in less than 4 hours of computation time?  
  
I would run a cluster on AWS and parallelize the problem. BFS can be run in parallel on each node. Therefore, I would assign a core from the AWS cluster to each node to compute BFS (2 million cores for 2 million nodes), get a list of all the distances like I do in distanceDistribution, then find the max path to get the diameter like I do in the diameter function.   
  
For 1 node and checking 44 million edges,  
x seconds = 4.5614 x10^-7 \* 44000001 = 20 seconds