In [8]:

```
import matplotlib.pyplot as plt
import networkx as nx
```

1. Load a graph database of your choosing from a text file or other source. If you take a large network dataset from the web (such as from Stanford Large Network Dataset Collection), please feel free at this point to load just a small subset of the nodes and edges.

Directed graph (each unordered pair of nodes is saved once): Amazon0302.txt Amazon product co-purchaisng network from March 02 2003 Nodes: 262111 Edges: 1234877

The original file had 262111 nodes and 1234877 edges, I tried to run nx and even after running for 2 hours it could not develop the graph due to size. I had to eliminate a majority of the data and left approximately 1500 nodes.

In [9]:

```
FileName="Amazon0302.txt"
Graphtype=nx.DiGraph()

G = nx.read_edgelist(FileName, create_using=Graphtype, nodetype=
int, data=(('weight',float),))
nx.info(G)
```

Out[9]:

```
'Name: \nType: DiGraph\nNumber of nodes: 1612\nNumbe
r of edges: 6120\nAverage in degree:   3.7965\nAvera
ge out degree:   3.7965'
```

In [10]:

```
nx.diameter(G)
```

```
---------------------------------------------------------
----------------------
NetworkXError                                     Traceback
(most recent call last)
<ipython-input-10-6332baa7b858> in <module>
----> 1 nx.diameter(G)

~/opt/anaconda3/lib/python3.7/site-packages/networkx
/algorithms/distance_measures.py in diameter(G, e, u
sebounds)
    274         return extrema_bounding(G, compute="
diameter")
    275     if e is None:
--> 276         e = eccentricity(G)
    277     return max(e.values())
    278

~/opt/anaconda3/lib/python3.7/site-packages/networkx
/algorithms/distance_measures.py in eccentricity(G,
v, sp)
    239                 msg = ('Found infinite path
length because the graph is not'
    240                        ' connected')
--> 241             raise networkx.NetworkXError(msg
)
    242
    243         e[n] = max(length.values())

NetworkXError: Found infinite path length because th
e digraph is not strongly connected
```

1. Create basic analysis on the graph, including the graph's diameter, and at least one other metric of your choosing. You may either code the functions by hand (to build your intuition and insight), or use functions in an existing package.

In [11]:

```
nx.number_of_nodes(G)
```

Out[11]:

```
1612
```

In [12]:

```
nx.number_of_edges(G)
```

Out[12]:

```
6120
```

In [19]:

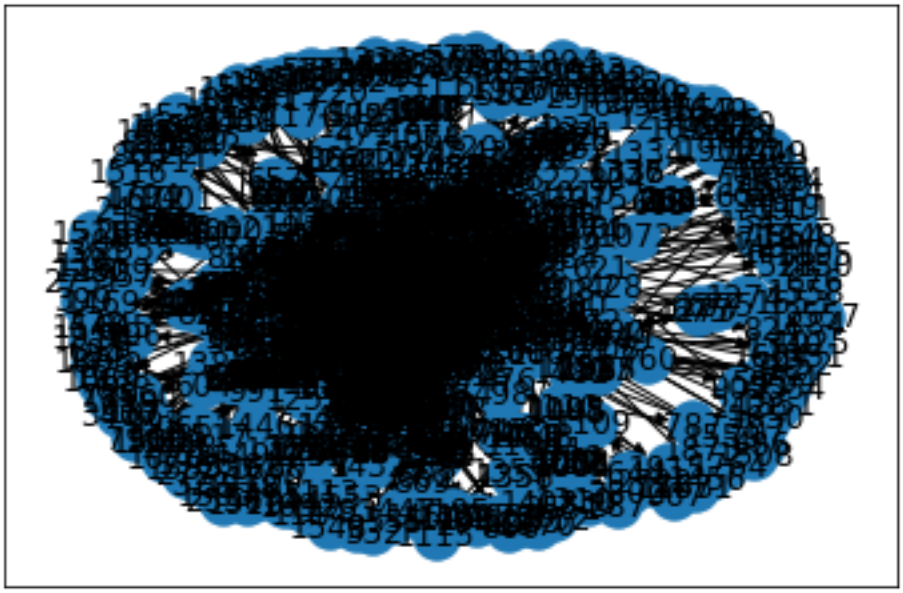
```
N, K = G.order(), G.size()
avg_deg = float(K) / N

print("Nodes: ", N)
print("Edges: ", K)
print("Average degree: ", avg_deg)
print("SCC: ", nx.number_strongly_connected_components(G))
print("WCC: ", nx.number_weakly_connected_components(G))
```

```
Nodes:  1612
Edges:  6120
Average degree:  3.796526054590571
SCC:  271
WCC:  1
```

It's a blur, the better method of viewing this is through a visualization tool. I decided to export into a .gexf format.

In [13]:

```
nx.draw_networkx(G)
```



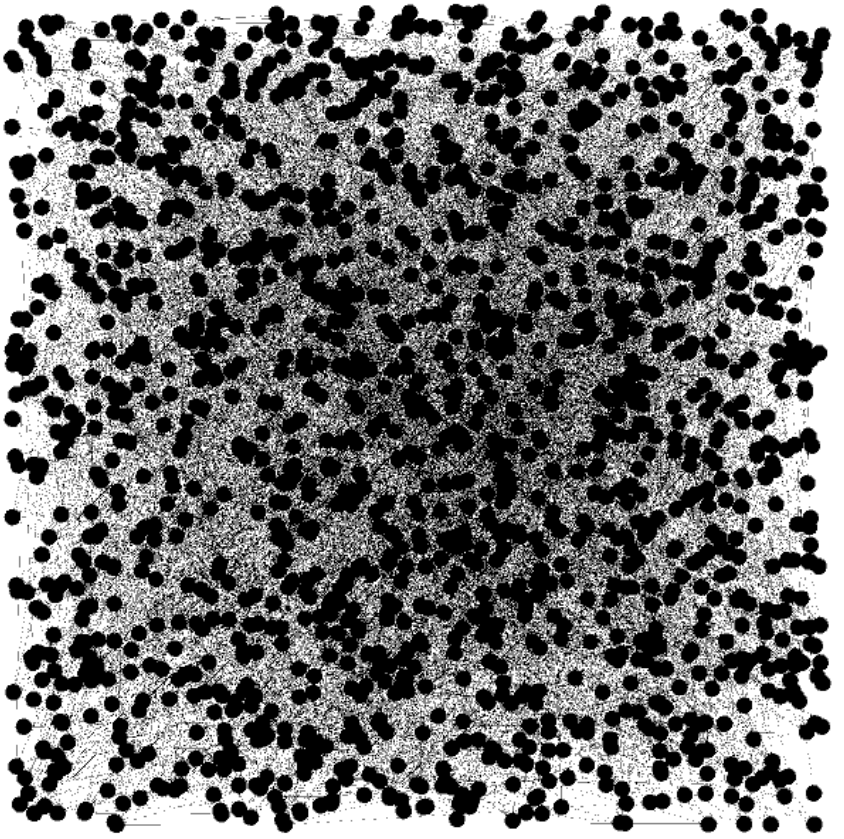
In [14]:

```
plt.show()
```

1. Use a visualization tool of your choice (Neo4j, Gephi, etc.) to display information.

In [21]:

```
nx.write_gexf(G, "Assignment3.gexf")
```



1. Please record a short video (~ 5 minutes), and submit a link to the video in advance of our meet-up.

https://www.youtube.com/watch?v=QP-V4F6pVgY&feature=youtu.be (https://www.youtube.com/watch?v=QP-V4F6pVgY&feature=youtu.be)