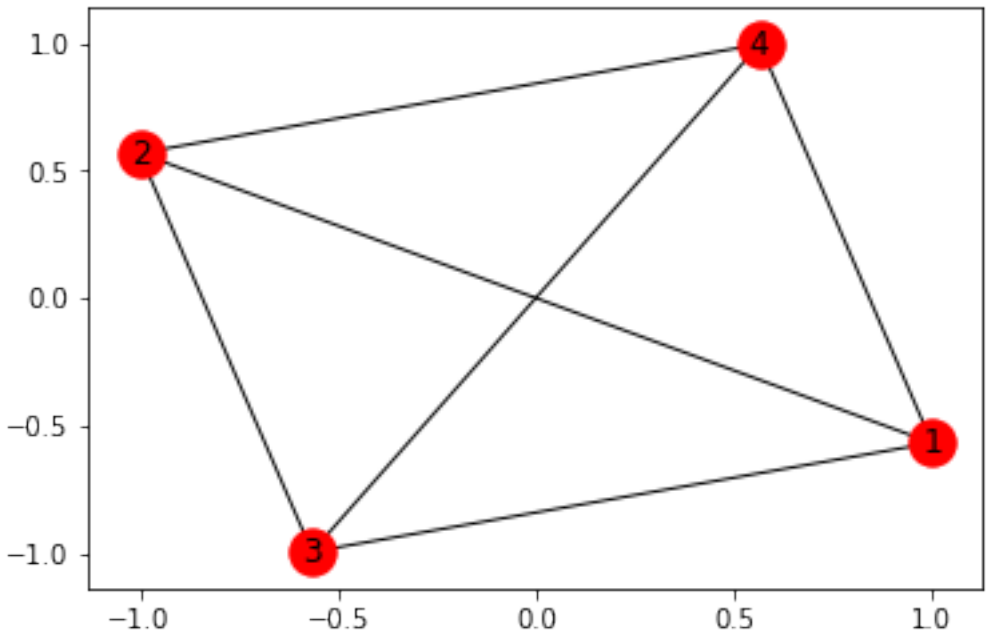
graphs

April 7, 2019

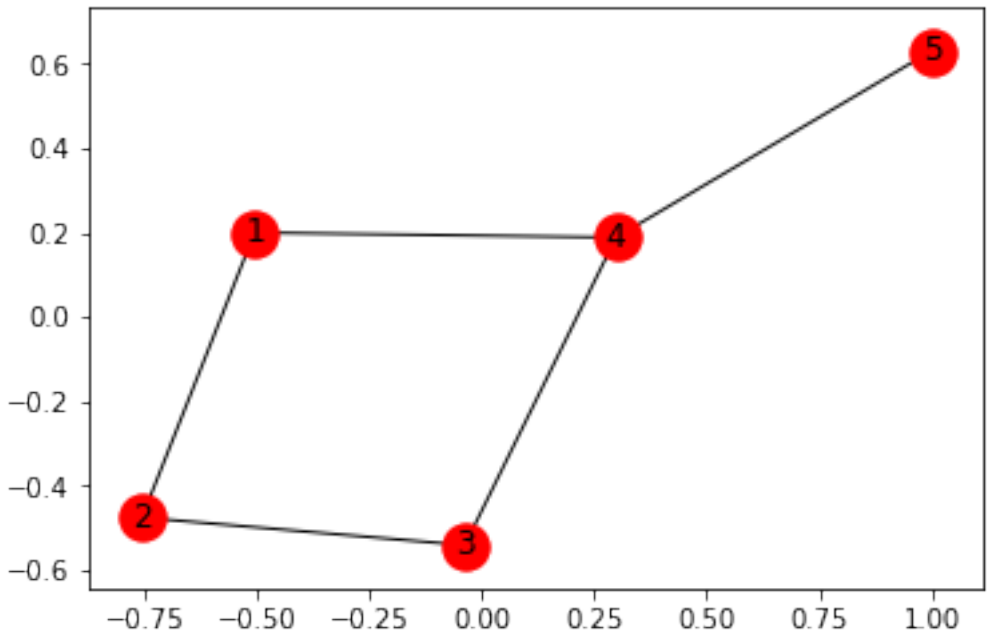
In [2]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import networkx as nx
```

In [2]:
```
G = nx.Graph()
G.add_edge(1,2)
G.add_edge(2,3)
G.add_edge(3,1)
G.add_edge(3,4)
G.add_edge(4,1)
G.add_edge(4,2)
nx.draw_networkx(G)
plt.show()
```

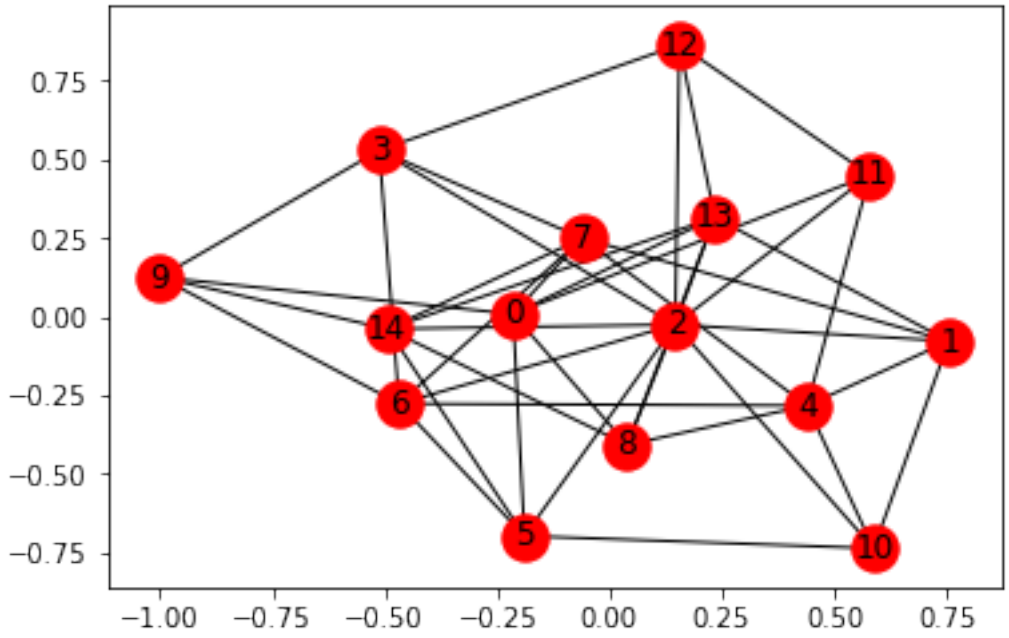
```
d:\python\lib\site-packages\networkx\drawing\nx_pylab.py:611: MatplotlibDeprecationWarning: isi
  if cb.is_numlike(alpha):
```



In [3]:
```
G = nx.Graph()
G.add_edges_from([(1,2), (2,3),(3,4), (4,1), (5,4)])
nx.draw_networkx(G)
plt.show()
```

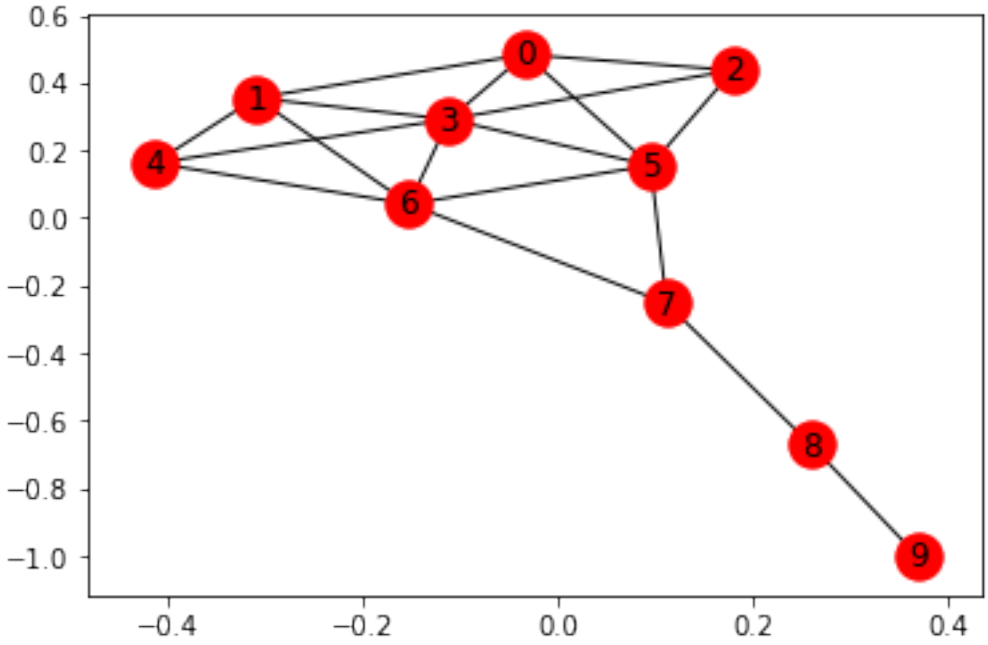


In [6]:
```
k = nx.fast_gnp_random_graph(15,0.40)
nx.draw_networkx(k)
plt.show()
```



In [8]:
```
G = nx.krackhardt_kite_graph()
nx.draw_networkx(G)
plt.show()
```

```
d:\python\lib\site-packages\networkx\drawing\nx_pylab.py:611: MatplotlibDeprecationWarning: is
  if cb.is_numlike(alpha):
```



```
In [10]: print(nx.has_path(G, source=1, target=9))
         print(nx.shortest_path(G, source=1, target=9))
         print(nx.shortest_path_length(G, source=1, target=9))
```

```
True
[1, 6, 7, 8, 9]
4
```

```
In [14]: all_paths = list(nx.shortest_simple_paths(G, source=1, target=9))
         for path in all_paths:
             print(path)
```

```
[1, 6, 7, 8, 9]
[1, 0, 5, 7, 8, 9]
[1, 6, 5, 7, 8, 9]
[1, 3, 5, 7, 8, 9]
[1, 4, 6, 7, 8, 9]
[1, 3, 6, 7, 8, 9]
[1, 0, 2, 5, 7, 8, 9]
[1, 0, 5, 6, 7, 8, 9]
[1, 6, 3, 5, 7, 8, 9]
[1, 3, 5, 6, 7, 8, 9]
[1, 4, 3, 5, 7, 8, 9]
[1, 4, 6, 5, 7, 8, 9]
[1, 3, 0, 5, 7, 8, 9]
[1, 3, 6, 5, 7, 8, 9]
[1, 0, 3, 5, 7, 8, 9]
[1, 4, 3, 6, 7, 8, 9]
[1, 3, 2, 5, 7, 8, 9]
[1, 0, 3, 6, 7, 8, 9]
[1, 3, 4, 6, 7, 8, 9]
[1, 0, 2, 3, 5, 7, 8, 9]
[1, 0, 2, 5, 6, 7, 8, 9]
[1, 0, 5, 3, 6, 7, 8, 9]
[1, 6, 4, 3, 5, 7, 8, 9]
[1, 6, 3, 0, 5, 7, 8, 9]
[1, 4, 3, 5, 6, 7, 8, 9]
[1, 4, 6, 3, 5, 7, 8, 9]
[1, 3, 0, 2, 5, 7, 8, 9]
[1, 3, 0, 5, 6, 7, 8, 9]
[1, 0, 3, 5, 6, 7, 8, 9]
[1, 4, 3, 0, 5, 7, 8, 9]
[1, 4, 3, 6, 5, 7, 8, 9]
[1, 3, 2, 0, 5, 7, 8, 9]
[1, 3, 2, 5, 6, 7, 8, 9]
```

```
[1, 0, 3, 2, 5, 7, 8, 9]
[1, 0, 3, 6, 5, 7, 8, 9]
[1, 3, 4, 6, 5, 7, 8, 9]
[1, 0, 2, 3, 6, 7, 8, 9]
[1, 6, 3, 2, 5, 7, 8, 9]
[1, 4, 3, 2, 5, 7, 8, 9]
[1, 0, 3, 4, 6, 7, 8, 9]
[1, 0, 2, 3, 5, 6, 7, 8, 9]
[1, 0, 2, 5, 3, 6, 7, 8, 9]
[1, 0, 5, 2, 3, 6, 7, 8, 9]
[1, 0, 5, 3, 4, 6, 7, 8, 9]
[1, 6, 4, 3, 0, 5, 7, 8, 9]
[1, 6, 3, 0, 2, 5, 7, 8, 9]
[1, 4, 6, 3, 0, 5, 7, 8, 9]
[1, 3, 0, 2, 5, 6, 7, 8, 9]
[1, 4, 3, 0, 2, 5, 7, 8, 9]
[1, 4, 3, 0, 5, 6, 7, 8, 9]
[1, 3, 2, 0, 5, 6, 7, 8, 9]
[1, 0, 3, 2, 5, 6, 7, 8, 9]
[1, 0, 2, 3, 4, 6, 7, 8, 9]
[1, 0, 2, 3, 6, 5, 7, 8, 9]
[1, 6, 3, 2, 0, 5, 7, 8, 9]
[1, 4, 3, 2, 0, 5, 7, 8, 9]
[1, 4, 3, 2, 5, 6, 7, 8, 9]
[1, 0, 3, 4, 6, 5, 7, 8, 9]
[1, 6, 4, 3, 2, 5, 7, 8, 9]
[1, 4, 6, 3, 2, 5, 7, 8, 9]
[1, 0, 2, 5, 3, 4, 6, 7, 8, 9]
[1, 0, 5, 2, 3, 4, 6, 7, 8, 9]
[1, 6, 4, 3, 0, 2, 5, 7, 8, 9]
[1, 4, 6, 3, 0, 2, 5, 7, 8, 9]
[1, 4, 3, 0, 2, 5, 6, 7, 8, 9]
[1, 0, 2, 3, 4, 6, 5, 7, 8, 9]
[1, 4, 3, 2, 0, 5, 6, 7, 8, 9]
[1, 6, 4, 3, 2, 0, 5, 7, 8, 9]
[1, 4, 6, 3, 2, 0, 5, 7, 8, 9]
```

```
In [11]: paths = nx.all_pairs_shortest_path(G)
         for path in paths:
             print(path)
```

```
(0, {0: [0], 1: [0, 1], 5: [0, 5], 10: [0, 10], 46: [0, 46], 50: [0, 50], 61: [0, 61], 75: [0,
(1, {1: [1], 0: [1, 0], 2: [1, 2], 3: [1, 3], 4: [1, 4], 8: [1, 8], 9: [1, 9], 11: [1, 11], 16
(2, {2: [2], 1: [2, 1], 0: [2, 1, 0], 3: [2, 1, 3], 4: [2, 1, 4], 8: [2, 1, 8], 9: [2, 1, 9], 
(3, {3: [3], 1: [3, 1], 13: [3, 13], 31: [3, 31], 0: [3, 1, 0], 2: [3, 1, 2], 4: [3, 1, 4], 8:
(4, {4: [4], 1: [4, 1], 7: [4, 7], 34: [4, 34], 66: [4, 66], 85: [4, 85], 0: [4, 1, 0], 2: [4,
(5, {5: [5], 0: [5, 0], 6: [5, 6], 14: [5, 14], 19: [5, 19], 32: [5, 32], 38: [5, 38], 52: [5,
```

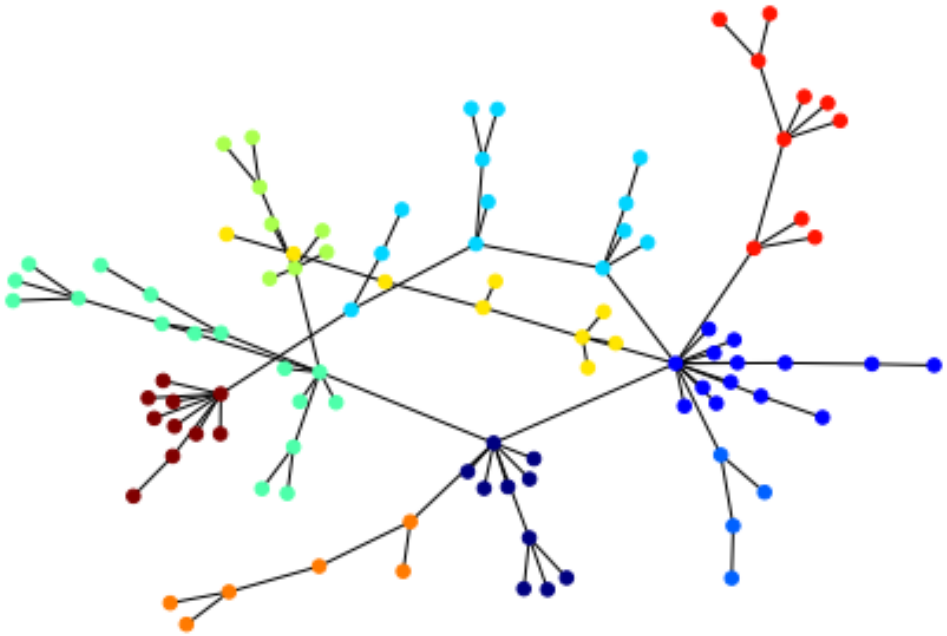
```
(6, {6: [6], 5: [6, 5], 18: [6, 18], 24: [6, 24], 54: [6, 54], 58: [6, 58], 97: [6, 97], 0: [6
(7, {7: [7], 4: [7, 4], 15: [7, 15], 39: [7, 39], 44: [7, 44], 1: [7, 4, 1], 34: [7, 4, 34], 66
(8, {8: [8], 1: [8, 1], 17: [8, 17], 62: [8, 62], 70: [8, 70], 84: [8, 84], 0: [8, 1, 0], 2: [8
(9, {9: [9], 1: [9, 1], 0: [9, 1, 0], 2: [9, 1, 2], 3: [9, 1, 3], 4: [9, 1, 4], 8: [9, 1, 8], 1
(10, {10: [10], 0: [10, 0], 27: [10, 27], 49: [10, 49], 1: [10, 0, 1], 5: [10, 0, 5], 46: [10,
(11, {11: [11], 1: [11, 1], 12: [11, 12], 23: [11, 23], 99: [11, 99], 0: [11, 1, 0], 2: [11, 1
(12, {12: [12], 11: [12, 11], 1: [12, 11, 1], 23: [12, 11, 23], 99: [12, 11, 99], 0: [12, 11, 1
(13, {13: [13], 3: [13, 3], 33: [13, 33], 1: [13, 3, 1], 31: [13, 3, 31], 0: [13, 3, 1, 0], 2:
(14, {14: [14], 5: [14, 5], 21: [14, 21], 0: [14, 5, 0], 6: [14, 5, 6], 19: [14, 5, 19], 32: [1
(15, {15: [15], 7: [15, 7], 25: [15, 25], 90: [15, 90], 4: [15, 7, 4], 39: [15, 7, 39], 44: [15
(16, {16: [16], 1: [16, 1], 0: [16, 1, 0], 2: [16, 1, 2], 3: [16, 1, 3], 4: [16, 1, 4], 8: [16
(17, {17: [17], 8: [17, 8], 22: [17, 22], 41: [17, 41], 1: [17, 8, 1], 62: [17, 8, 62], 70: [17
(18, {18: [18], 6: [18, 6], 5: [18, 6, 5], 24: [18, 6, 24], 54: [18, 6, 54], 58: [18, 6, 58], 9
(19, {19: [19], 5: [19, 5], 37: [19, 37], 71: [19, 71], 0: [19, 5, 0], 6: [19, 5, 6], 14: [19,
(20, {20: [20], 1: [20, 1], 29: [20, 29], 0: [20, 1, 0], 2: [20, 1, 2], 3: [20, 1, 3], 4: [20,
(21, {21: [21], 14: [21, 14], 40: [21, 40], 76: [21, 76], 79: [21, 79], 5: [21, 14, 5], 0: [21
(22, {22: [22], 17: [22, 17], 55: [22, 55], 8: [22, 17, 8], 41: [22, 17, 41], 67: [22, 55, 67]
(23, {23: [23], 11: [23, 11], 60: [23, 60], 68: [23, 68], 80: [23, 80], 81: [23, 81], 1: [23, 1
(24, {24: [24], 6: [24, 6], 43: [24, 43], 48: [24, 48], 5: [24, 6, 5], 18: [24, 6, 18], 54: [24
(25, {25: [25], 15: [25, 15], 26: [25, 26], 30: [25, 30], 36: [25, 36], 42: [25, 42], 47: [25,
(26, {26: [26], 25: [26, 25], 15: [26, 25, 15], 30: [26, 25, 30], 36: [26, 25, 36], 42: [26, 25
(27, {27: [27], 10: [27, 10], 45: [27, 45], 0: [27, 10, 0], 49: [27, 10, 49], 59: [27, 45, 59]
(28, {28: [28], 1: [28, 1], 0: [28, 1, 0], 2: [28, 1, 2], 3: [28, 1, 3], 4: [28, 1, 4], 8: [28
(29, {29: [29], 20: [29, 20], 1: [29, 20, 1], 0: [29, 20, 1, 0], 2: [29, 20, 1, 2], 3: [29, 20
(30, {30: [30], 25: [30, 25], 15: [30, 25, 15], 26: [30, 25, 26], 36: [30, 25, 36], 42: [30, 25
(31, {31: [31], 3: [31, 3], 1: [31, 3, 1], 13: [31, 3, 13], 0: [31, 3, 1, 0], 2: [31, 3, 1, 2]
(32, {32: [32], 5: [32, 5], 0: [32, 5, 0], 6: [32, 5, 6], 14: [32, 5, 14], 19: [32, 5, 19], 38
(33, {33: [33], 13: [33, 13], 3: [33, 13, 3], 1: [33, 13, 3, 1], 31: [33, 13, 3, 31], 0: [33, 1
(34, {34: [34], 4: [34, 4], 53: [34, 53], 1: [34, 4, 1], 7: [34, 4, 7], 66: [34, 4, 66], 85: [3
(35, {35: [35], 1: [35, 1], 0: [35, 1, 0], 2: [35, 1, 2], 3: [35, 1, 3], 4: [35, 1, 4], 8: [35
(36, {36: [36], 25: [36, 25], 63: [36, 63], 15: [36, 25, 15], 26: [36, 25, 26], 30: [36, 25, 30
(37, {37: [37], 19: [37, 19], 5: [37, 19, 5], 71: [37, 19, 71], 0: [37, 19, 5, 0], 6: [37, 19,
(38, {38: [38], 5: [38, 5], 0: [38, 5, 0], 6: [38, 5, 6], 14: [38, 5, 14], 19: [38, 5, 19], 32
(39, {39: [39], 7: [39, 7], 4: [39, 7, 4], 15: [39, 7, 15], 44: [39, 7, 44], 1: [39, 7, 4, 1],
(40, {40: [40], 21: [40, 21], 14: [40, 21, 14], 76: [40, 21, 76], 79: [40, 21, 79], 5: [40, 21
(41, {41: [41], 17: [41, 17], 8: [41, 17, 8], 22: [41, 17, 22], 1: [41, 17, 8, 1], 62: [41, 17
(42, {42: [42], 25: [42, 25], 15: [42, 25, 15], 26: [42, 25, 26], 30: [42, 25, 30], 36: [42, 25
(43, {43: [43], 24: [43, 24], 6: [43, 24, 6], 48: [43, 24, 48], 5: [43, 24, 6, 5], 18: [43, 24
(44, {44: [44], 7: [44, 7], 87: [44, 87], 91: [44, 91], 4: [44, 7, 4], 15: [44, 7, 15], 39: [44
(45, {45: [45], 27: [45, 27], 59: [45, 59], 95: [45, 95], 10: [45, 27, 10], 0: [45, 27, 10, 0]
(46, {46: [46], 0: [46, 0], 73: [46, 73], 74: [46, 74], 86: [46, 86], 1: [46, 0, 1], 5: [46, 0
(47, {47: [47], 25: [47, 25], 15: [47, 25, 15], 26: [47, 25, 26], 30: [47, 25, 30], 36: [47, 25
(48, {48: [48], 24: [48, 24], 6: [48, 24, 6], 43: [48, 24, 43], 5: [48, 24, 6, 5], 18: [48, 24
(49, {49: [49], 10: [49, 10], 0: [49, 10, 0], 27: [49, 10, 27], 1: [49, 10, 0, 1], 5: [49, 10,
(50, {50: [50], 0: [50, 0], 1: [50, 0, 1], 5: [50, 0, 5], 10: [50, 0, 10], 46: [50, 0, 46], 61
(51, {51: [51], 25: [51, 25], 15: [51, 25, 15], 26: [51, 25, 26], 30: [51, 25, 30], 36: [51, 25
(52, {52: [52], 5: [52, 5], 64: [52, 64], 77: [52, 77], 0: [52, 5, 0], 6: [52, 5, 6], 14: [52,
(53, {53: [53], 34: [53, 34], 4: [53, 34, 4], 1: [53, 34, 4, 1], 7: [53, 34, 4, 7], 66: [53, 34
```

```
(54, {54: [54], 6: [54, 6], 5: [54, 6, 5], 18: [54, 6, 18], 24: [54, 6, 24], 58: [54, 6, 58], 9
(55, {55: [55], 22: [55, 22], 67: [55, 67], 17: [55, 22, 17], 8: [55, 22, 17, 8], 41: [55, 22,
(56, {56: [56], 25: [56, 25], 15: [56, 25, 15], 26: [56, 25, 26], 30: [56, 25, 30], 36: [56, 25
(57, {57: [57], 1: [57, 1], 0: [57, 1, 0], 2: [57, 1, 2], 3: [57, 1, 3], 4: [57, 1, 4], 8: [57
(58, {58: [58], 6: [58, 6], 5: [58, 6, 5], 18: [58, 6, 18], 24: [58, 6, 24], 54: [58, 6, 54], 9
(59, {59: [59], 45: [59, 45], 27: [59, 45, 27], 95: [59, 45, 95], 10: [59, 45, 27, 10], 0: [59
(60, {60: [60], 23: [60, 23], 11: [60, 23, 11], 68: [60, 23, 68], 80: [60, 23, 80], 81: [60, 23
(61, {61: [61], 0: [61, 0], 1: [61, 0, 1], 5: [61, 0, 5], 10: [61, 0, 10], 46: [61, 0, 46], 50
(62, {62: [62], 8: [62, 8], 1: [62, 8, 1], 17: [62, 8, 17], 70: [62, 8, 70], 84: [62, 8, 84], 0
(63, {63: [63], 36: [63, 36], 25: [63, 36, 25], 15: [63, 36, 25, 15], 26: [63, 36, 25, 26], 30
(64, {64: [64], 52: [64, 52], 5: [64, 52, 5], 77: [64, 52, 77], 0: [64, 52, 5, 0], 6: [64, 52,
(65, {65: [65], 1: [65, 1], 78: [65, 78], 0: [65, 1, 0], 2: [65, 1, 2], 3: [65, 1, 3], 4: [65,
(66, {66: [66], 4: [66, 4], 1: [66, 4, 1], 7: [66, 4, 7], 34: [66, 4, 34], 85: [66, 4, 85], 0:
(67, {67: [67], 55: [67, 55], 22: [67, 55, 22], 17: [67, 55, 22, 17], 8: [67, 55, 22, 17, 8], 4
(68, {68: [68], 23: [68, 23], 88: [68, 88], 98: [68, 98], 11: [68, 23, 11], 60: [68, 23, 60], 8
(69, {69: [69], 25: [69, 25], 15: [69, 25, 15], 26: [69, 25, 26], 30: [69, 25, 30], 36: [69, 25
(70, {70: [70], 8: [70, 8], 1: [70, 8, 1], 17: [70, 8, 17], 62: [70, 8, 62], 84: [70, 8, 84], 0
(71, {71: [71], 19: [71, 19], 5: [71, 19, 5], 37: [71, 19, 37], 0: [71, 19, 5, 0], 6: [71, 19,
(72, {72: [72], 5: [72, 5], 0: [72, 5, 0], 6: [72, 5, 6], 14: [72, 5, 14], 19: [72, 5, 19], 32
(73, {73: [73], 46: [73, 46], 0: [73, 46, 0], 74: [73, 46, 74], 86: [73, 46, 86], 1: [73, 46, 0
(74, {74: [74], 46: [74, 46], 0: [74, 46, 0], 73: [74, 46, 73], 86: [74, 46, 86], 1: [74, 46, 0
(75, {75: [75], 0: [75, 0], 1: [75, 0, 1], 5: [75, 0, 5], 10: [75, 0, 10], 46: [75, 0, 46], 50
(76, {76: [76], 21: [76, 21], 14: [76, 21, 14], 40: [76, 21, 40], 79: [76, 21, 79], 5: [76, 21
(77, {77: [77], 52: [77, 52], 94: [77, 94], 5: [77, 52, 5], 64: [77, 52, 64], 0: [77, 52, 5, 0]
(78, {78: [78], 65: [78, 65], 93: [78, 93], 1: [78, 65, 1], 0: [78, 65, 1, 0], 2: [78, 65, 1, 2
(79, {79: [79], 21: [79, 21], 14: [79, 21, 14], 40: [79, 21, 40], 76: [79, 21, 76], 5: [79, 21
(80, {80: [80], 23: [80, 23], 11: [80, 23, 11], 60: [80, 23, 60], 68: [80, 23, 68], 81: [80, 23
(81, {81: [81], 23: [81, 23], 11: [81, 23, 11], 60: [81, 23, 60], 68: [81, 23, 68], 80: [81, 23
(82, {82: [82], 0: [82, 0], 1: [82, 0, 1], 5: [82, 0, 5], 10: [82, 0, 10], 46: [82, 0, 46], 50
(83, {83: [83], 1: [83, 1], 0: [83, 1, 0], 2: [83, 1, 2], 3: [83, 1, 3], 4: [83, 1, 4], 8: [83
(84, {84: [84], 8: [84, 8], 1: [84, 8, 1], 17: [84, 8, 17], 62: [84, 8, 62], 70: [84, 8, 70], 0
(85, {85: [85], 4: [85, 4], 1: [85, 4, 1], 7: [85, 4, 7], 34: [85, 4, 34], 66: [85, 4, 66], 0:
(86, {86: [86], 46: [86, 46], 0: [86, 46, 0], 73: [86, 46, 73], 74: [86, 46, 74], 1: [86, 46, 0
(87, {87: [87], 44: [87, 44], 7: [87, 44, 7], 91: [87, 44, 91], 4: [87, 44, 7, 4], 15: [87, 44
(88, {88: [88], 68: [88, 68], 23: [88, 68, 23], 98: [88, 68, 98], 11: [88, 68, 23, 11], 60: [88
(89, {89: [89], 1: [89, 1], 0: [89, 1, 0], 2: [89, 1, 2], 3: [89, 1, 3], 4: [89, 1, 4], 8: [89
(90, {90: [90], 15: [90, 15], 92: [90, 92], 7: [90, 15, 7], 25: [90, 15, 25], 4: [90, 15, 7, 4]
(91, {91: [91], 44: [91, 44], 7: [91, 44, 7], 87: [91, 44, 87], 4: [91, 44, 7, 4], 15: [91, 44
(92, {92: [92], 90: [92, 90], 15: [92, 90, 15], 7: [92, 90, 15, 7], 25: [92, 90, 15, 25], 4: [9
(93, {93: [93], 78: [93, 78], 65: [93, 78, 65], 1: [93, 78, 65, 1], 0: [93, 78, 65, 1, 0], 2:
(94, {94: [94], 77: [94, 77], 52: [94, 77, 52], 5: [94, 77, 52, 5], 64: [94, 77, 52, 64], 0: [9
(95, {95: [95], 45: [95, 45], 27: [95, 45, 27], 59: [95, 45, 59], 10: [95, 45, 27, 10], 0: [95
(96, {96: [96], 0: [96, 0], 1: [96, 0, 1], 5: [96, 0, 5], 10: [96, 0, 10], 46: [96, 0, 46], 50
(97, {97: [97], 6: [97, 6], 5: [97, 6, 5], 18: [97, 6, 18], 24: [97, 6, 24], 54: [97, 6, 54], 5
(98, {98: [98], 68: [98, 68], 23: [98, 68, 23], 88: [98, 68, 88], 11: [98, 68, 23, 11], 60: [98
(99, {99: [99], 11: [99, 11], 1: [99, 11, 1], 12: [99, 11, 12], 23: [99, 11, 23], 0: [99, 11, 1
```

In [14]:
```
import community.community_louvain as community
#print(dir(community))
G = nx.powerlaw_cluster_graph(100, 1, .4, seed=101)
partition = community.best_partition(G)
for i in set(partition.values()):
    print("Society ",i)
    members = list_nodes = [nodes for nodes in partition.keys() if partition[nodes] ==
    print(members)
values = [partition.get(node) for node in G.nodes()]
nx.draw_spring(G, cmap=plt.get_cmap('jet'), node_color=values, node_size=30, with_labe
plt.show()
print("Level of modularity: ", community.modularity(partition, G))
```

```
Society  0
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  1
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  2
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  3
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  4
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  5
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  6
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  7
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  8
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]
Society  9
[1, 2, 9, 16, 20, 28, 29, 35, 57, 65, 78, 83, 89, 93]


d:\python\lib\site-packages\networkx\drawing\nx_pylab.py:611: MatplotlibDeprecationWarning: isi
  if cb.is_numlike(alpha):
```



```
Level of modularity:  0.7941026425874911
```

In [ ]: