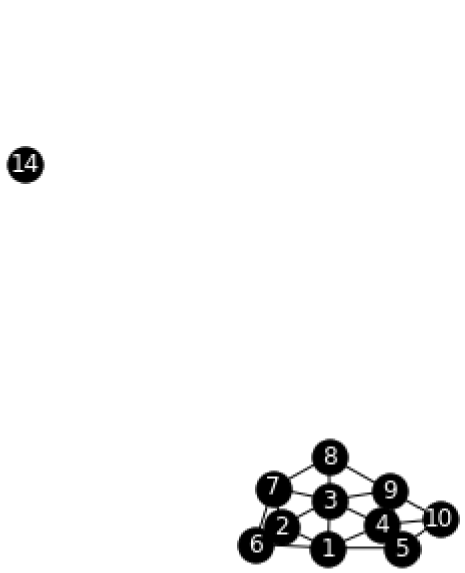
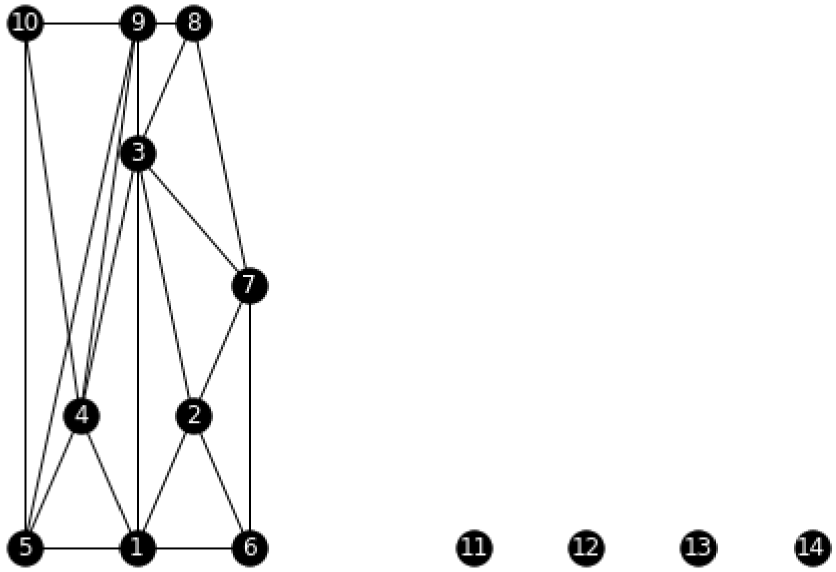
In [91]:

```
import networkx as nx
import numpy as np
import matplotlib.pyplot as plt
g = nx.read_adjlist("graph.txt", nodetype=int)
nx.draw(g, with_labels=True, node_color='black', font_color='white')
plt.show()
```



In [92]:

```
position = { 1 :(-2, 0), 2 :(-1.5, 0.25), 3 :(-2, 0.75), 4 :(-2.5, 0.25), 5 :(-3, 0)
nx.draw(g, pos=position, with_labels=True, node_color='black', font_color='white')
plt.show()
```



In [102…

```
params = []
for component in nx.connected_components(g):
    nodedegree = []
    nodeeccentricity = []
    nodediameter = []
    noderadius = []
    sub = g.subgraph(component)
    params.append(nx.number_of_nodes(sub))
    params.append(nx.number_of_edges(sub))
    for u in nx.nodes(sub):
        nodedegree.append(sub.degree(u))
        nodeeccentricity.append(nx.eccentricity(sub, v=u))
    params.append(nodedegree)
    params.append(nodeeccentricity)
    params.append(min(nodeeccentricity)) #radius
    params.append(max(nodeeccentricity)) #diameter
params = np.array(params, dtype=object)
params = params.reshape(5,6)
```

In [94]:

```
print(f"1st component \n\
number of nodes:{params[0][0]}\n\
number of edges:{params[0][1]}\n\
each node degree: '1':{params[0][2][0]},'2':{params[0][2][1]},'3':{params[0][2][2]},
each node eccentricity:'1':{params[0][3][0]},'2':{params[0][3][1]},'3':{params[0][3]
radius: {params[0][4]}\n\
diameter: {params[0][5]}\n\
2nd component \n\
number of nodes:{params[1][0]}\n\
number of edges:{params[1][1]}\n\
each node degree: '11':{params[1][2][0]}\n\
each node eccentricity:'11':{params[1][2][0]}\n\
radius: {params[1][4]}\n\
diameter: {params[1][5]}\n\
3rd component \n\
number of nodes:{params[2][0]}\n\
number of edges:{params[2][1]}\n\
each node degree: '12':{params[2][2][0]}\n\
each node eccentricity:'12':{params[1][2][0]}\n\
radius: {params[2][4]}\n\
diameter: {params[2][5]}\n\
4th component \n\
number of nodes:{params[3][0]}\n\
number of edges:{params[3][1]}\n\
each node degree: '13':{params[3][2][0]}\n\
each node eccentricity:'13':{params[3][2][0]}\n\
radius: {params[1][4]}\n\
diameter: {params[1][5]}\n\
5th component \n\
number of nodes:{params[4][0]}\n\
number of edges:{params[4][1]}\n\
each node degree: '14':{params[4][2][0]}\n\
each node eccentricity:'14':{params[4][2][0]}\n\
radius: {params[4][4]}\n\
diameter: {params[4][5]}")
```

```
1st component
number of nodes:10
number of edges:21
each node degree: '1':5,'2':4,'3':6,'4':5,'5':4,'6':3,'7':4,'8':3,'9':5,'10':3
each node eccentricity:'1':2,'2':3,'3':2,'4':2,'5':3,'6':3,'7':3,'8':2,'9':3,'10':3
radius: 2
diameter: 3
2nd component
number of nodes:1
number of edges:0
each node degree: '11':0
each node eccentricity:'11':0
radius: 0
diameter: 0
3rd component
number of nodes:1
number of edges:0
each node degree: '12':0
each node eccentricity:'12':0
radius: 0
```

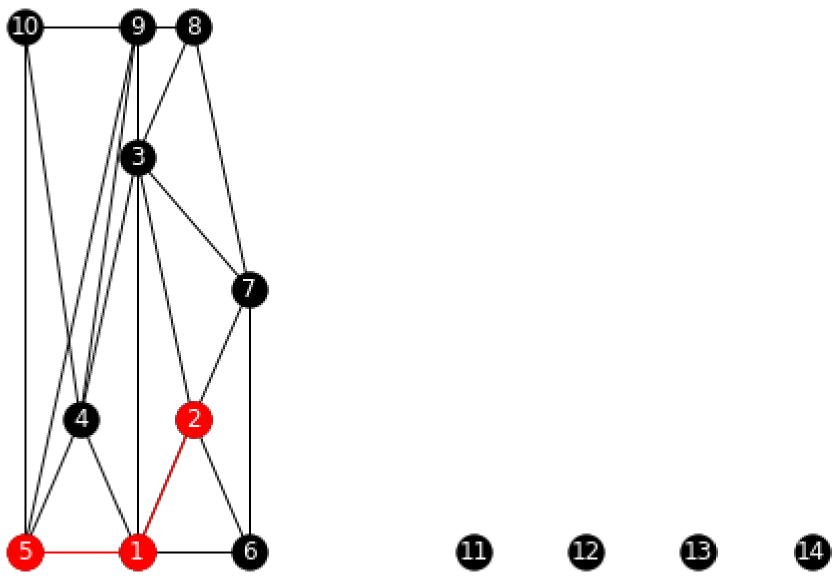
```
diameter: 0
4th component
number of nodes:1
number of edges:0
each node degree: '13':0
each node eccentricity:'13':0
radius: 0
diameter: 0
5th component
number of nodes:1
number of edges:0
each node degree: '14':0
each node eccentricity:'14':0
radius: 0
diameter: 0
```

In [95]:

```
diameter = []
red_nodes = []
red_edges = []
for component in nx.connected_components(g):
    diam_nodes = []
    sub = g.subgraph(component)
    ecc= nx.eccentricity(sub)
    diameter = max(ecc.values())
    for key, value in ecc.items():
        if value == diameter:
             diam_nodes.append(key)
        if len(diam_nodes) == 2:
            break
    try:
        diameter = nx.shortest_path(sub, source=diam_nodes[0], target=diam_nodes[1])
        red_nodes = red_nodes + diameter #add diameter nodes to list of nodes, which
        i = 0
        for el in diameter: #creating list of edges, which should be colored
            red_edges.append((diameter[i], diameter[i+1]))
            i+=1
    except IndexError: #pass, if component is trivial
        pass
```

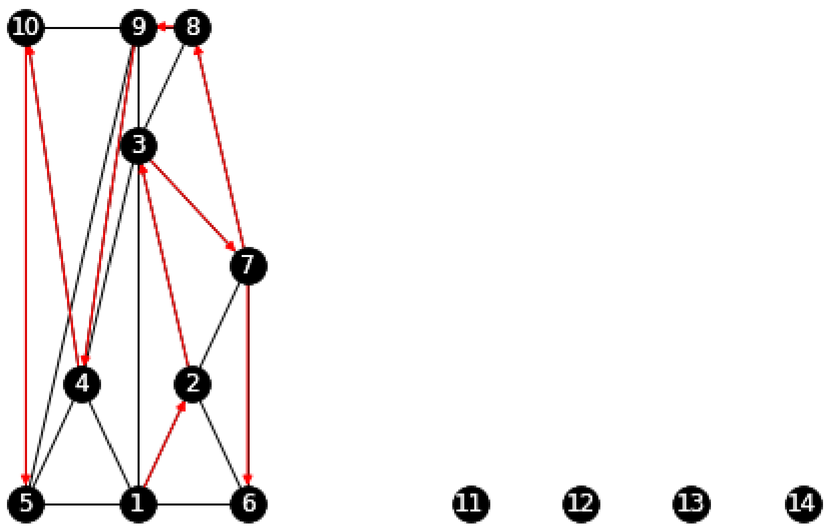
In [96]:

```
nx.draw(g, pos=position, with_labels=True, node_color='black', font_color='white')
nx.draw_networkx_nodes(g, pos=position, nodelist=red_nodes, node_color="red")
nx.draw_networkx_edges(g, pos=position, edgelist=red_edges, edge_color="red",)
plt.draw()
```



In [117…

```
nx.draw(g, pos=position, with_labels=True, node_color='black', font_color='white')
for component in nx.connected_components(g):
    sub = g.subgraph(component)
    nodes = list(component)
    tree = nx.dfs_tree(sub, nodes[0])
    nx.draw(tree, pos=position, with_labels=True, node_color='black', edge_color ='r
plt.draw()
```



In [ ]: