In [1]:

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

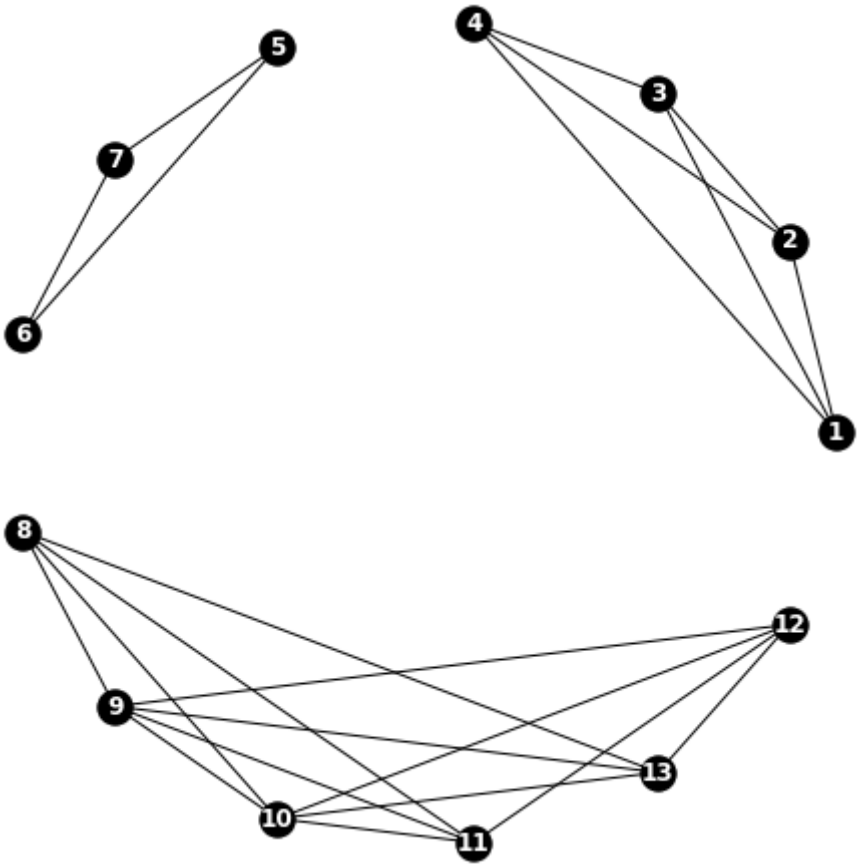
In [2]:

```
def set_own_pos():
    pos = {x: [float(x) // 2, math.sin(float(x) * (math.pi / 1.5))] for x in G.nodes()}
    pos["8"] = [3.4, 0]
    pos["9"] = [4.4, -0.1]
    pos["10"] = [5.1, 0.3]
    pos["13"] = [5, 2]
    return pos


def new_edges(tuple_):
    tmp_node = []
    for elements in tuple_:
        for node_ in elements:
            if node_ not in tmp_node:
                tmp_node.append(node_)
    G.remove_edges_from(tuple_)

    nx.draw(G, pos=set_own_pos(), with_labels=True, font_weight='bold', node_color="black",
            font_color="w")
    G.add_edges_from(tuple_)

    nx.draw_networkx_edges(G, pos=set_own_pos(), edgelist=tuple_, edge_color='r')
    return tmp_node


def part3_4():
    n = 1
    tmp_edges_global = []
    for x in (nx.connected_components(G)):
        tmp = nx.subgraph(G, x)
        print(f"-------Begin of the component {n}-------")

        x = sorted(x, key=float)

        print(f'nodes = {len(x)} ')
        print(f'edges = {len(G.edges(x))}')
        eccentr = nx.eccentricity(tmp)
        for node_ in x:
            print(f"~~~~~~~~Node {node_}~~~~~~~~")
            print(f'''degree: {G.degree[node_]}''')
            print(f'''eccentricity: {eccentr[node_]}''')
        print("")
        diam = nx.diameter(tmp, eccentr)
        print(f'''Radius of the component {n} = {nx.radius(tmp, eccentr)}''')
        print(f'''Diameter of the component {n} = {diam}''')
        tmp_edges = []

        for node1 in x:
            for node2 in x:
                for path in (nx.all_simple_edge_paths(tmp, node1, node2)):
                    if len(list(path)) == diam:
                        tmp_edges.append(path)
        random_edges = random.choice(tmp_edges)
        tmp_edges_global.extend(random_edges)

        print(f"--------End of the component {n}--------\n")
        n += 1

    tmp_node = new_edges(tmp_edges_global)
```

```
    nx.draw_networkx_nodes(G, pos=set_own_pos(), nodelist=tmp_node, node_color='r')
```

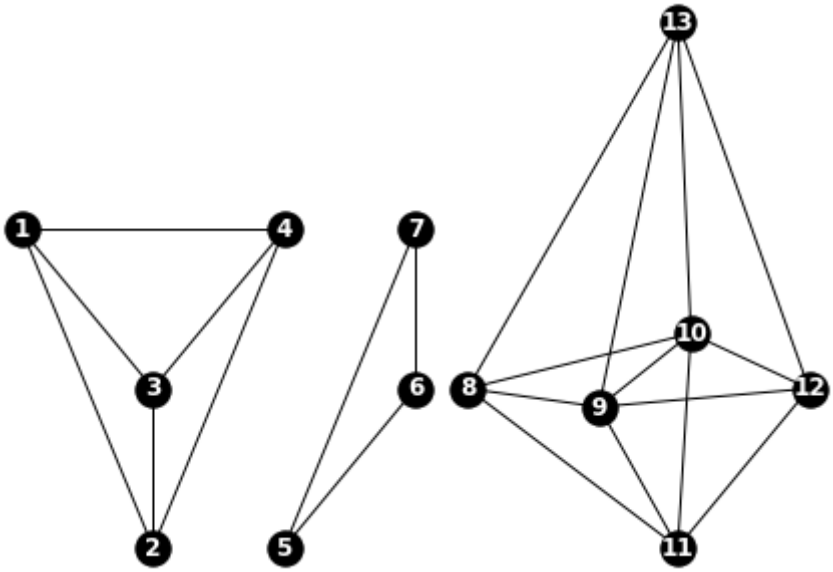
In [5]:

```
"""
PART 1
"""
my_graph = nx.Graph()
f = open("data.txt", "r")
plt.figure(figsize=(8, 8))

plt.axes().set_aspect("equal", adjustable="datalim")
G = nx.read_adjlist("data.txt")

nx.draw_circular(G, with_labels=True, font_weight='bold', node_color="black", edge_color="b
plt.savefig('simple.png')
```



In [4]:

```
"""
PART 2
"""

nx.draw(G, pos=set_own_pos(), with_labels=True, font_weight='bold', node_color="black", edg
        font_color="w")
plt.savefig('processed.png')
```

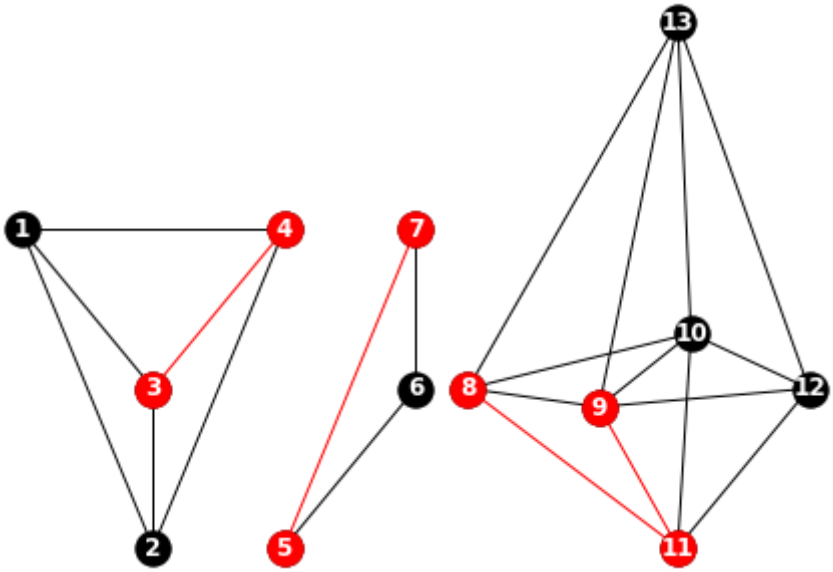


In [6]:

```
"""
PART 3-4
"""

part3_4()
plt.savefig('diam.png')
plt.show()
plt.close()
```

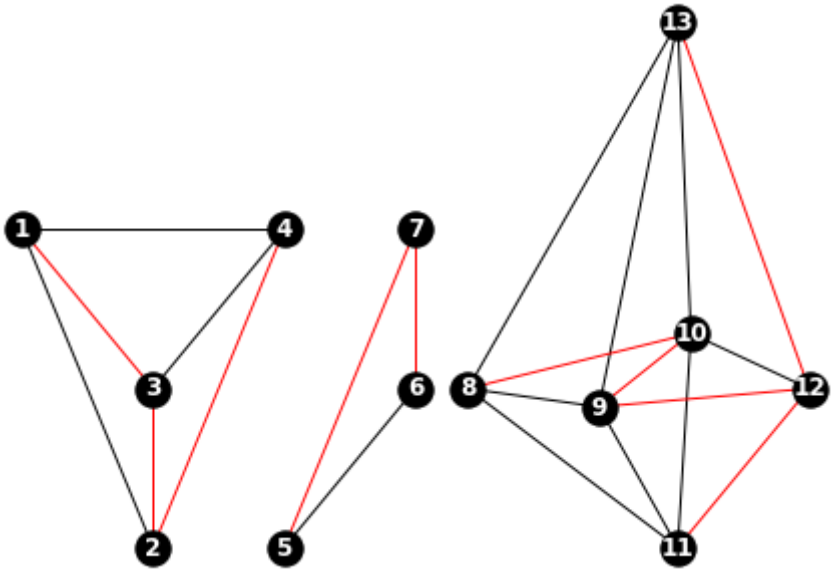
```
-------Begin of the component 1-------
nodes = 4
edges = 6
~~~~~~~~Node 1~~~~~~~~
degree: 3
eccentricity: 1
~~~~~~~~Node 2~~~~~~~~
degree: 3
eccentricity: 1
~~~~~~~~Node 3~~~~~~~~
degree: 3
eccentricity: 1
~~~~~~~~Node 4~~~~~~~~
degree: 3
eccentricity: 1

Radius of the component 1 = 1
Diameter of the component 1 = 1
--------End of the component 1--------

-------Begin of the component 2-------
nodes = 3
edges = 3
~~~~~~~~Node 5~~~~~~~~
degree: 2
eccentricity: 1
~~~~~~~~Node 6~~~~~~~~
degree: 2
eccentricity: 1
~~~~~~~~Node 7~~~~~~~~
degree: 2
eccentricity: 1

Radius of the component 2 = 1
Diameter of the component 2 = 1
--------End of the component 2--------

-------Begin of the component 3-------
nodes = 6
edges = 13
~~~~~~~~Node 8~~~~~~~~
degree: 4
eccentricity: 2
~~~~~~~~Node 9~~~~~~~~
degree: 5
eccentricity: 1
~~~~~~~~Node 10~~~~~~~~
degree: 5
eccentricity: 1
~~~~~~~~Node 11~~~~~~~~
```

```
degree: 4
eccentricity: 2
~~~~~~~~Node 12~~~~~~~~
degree: 4
eccentricity: 2
~~~~~~~~Node 13~~~~~~~~
degree: 4
eccentricity: 2

Radius of the component 3 = 1
Diameter of the component 3 = 2
--------End of the component 3--------
```



In [8]:

```
"""
PART 5
"""
tmp_edges = []
for el in (nx.dfs_edges(G)):
    tmp_edges.append(el)
new_edges(tmp_edges)
plt.savefig('forest.png')
```



In [ ]: