1. Introduction:

A python code is written to extract the gates and information about the interconnects. The gates are considered as nodes and interconnects as edges. From the obtained Graph, shortest path from a source node to all other nodes is found out.

2. Inputs:

The file path of a benchmark file which is a netlist file is given as input. After that an option for directed and undirected graph is asked. Option should be entered by the user and accordingly the graph is plotted.

Eg.

```
inputs, outputs, gate  io =
netlist_graph.extract_gate_io(r"C:\Users\akhil\OneDrive\Documents\SEMESTER_2\VDA\Assignment_1\
ISCAS89\s27.bench")
```

```
PS C:\Users\akhil\OneDrive\Documents\SEMESTER_2\VDA\Assignment_1> & C:/Python311/python.exe c:
/Users/akhil/OneDrive/Documents/SEMESTER_2/VDA/Assignment_1/main.py
Choose one of the following:
1. Directed Graph
2. Undirected Graph
Enter your option:
```

3. Outputs:

Once the required option is chosen, the list of nodes, edges and the adjacency matrix is displayed. In the edge list, the nodes are displayed as numbers which corresponds to the index value in node list.

Below is the output when chosen option is Directed Graph.

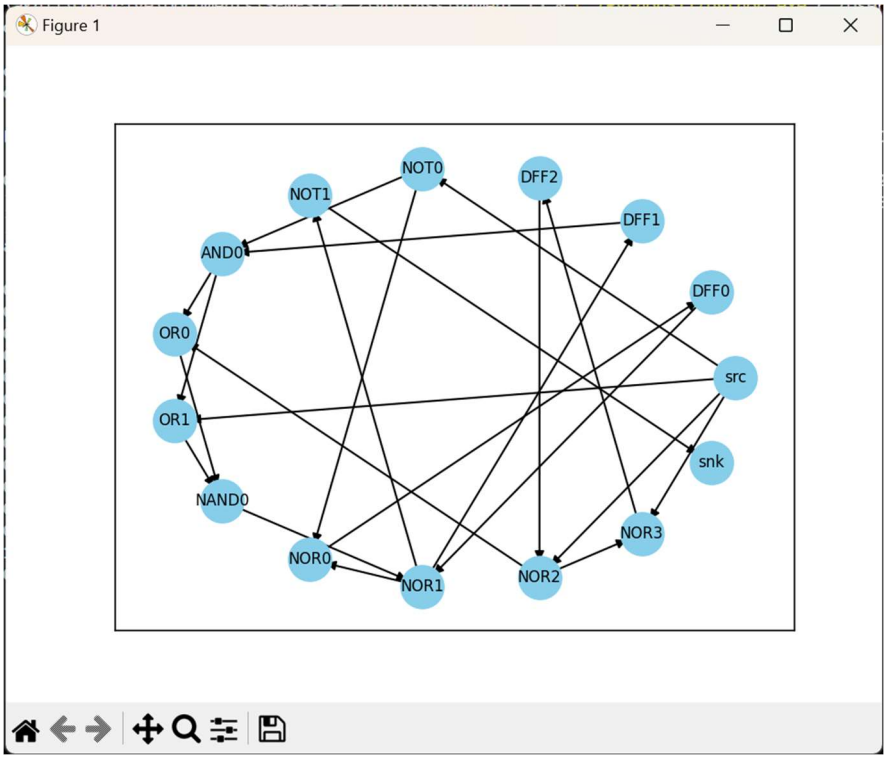
Node list and Edge list:

```
Choose one of the following:
1. Directed Graph
2. Undirected Graph
Enter your option: 1
Node List:
 ['SRC', 'DFF0', 'DFF1', 'DFF2', 'NOT0', 'NOT1', 'AND0', 'OR0', 'OR1', 'NAND0', 'NOR0', 'NOR1', 'NOR2', 'NOR3', 'SNK']
Edge List:
 [(0, 4, 'G0'), (0, 12, 'G1'), (10, 1, 'G10'), (11, 2, 'G11'), (11, 5, 'G11'), (11, 10, 'G11'), (12, 7, 'G12'), (12, 13, 'G12'), (13, 3, 'G13'), (4, 6, 'G14
'), (4, 10, 'G14'), (7, 9, 'G15'), (8, 9, 'G16'), (5, 14, 'G17'), (0, 13, 'G2'), (0, 8, 'G3'), (1, 11, 'G5'), (2, 6, 'G6'), (3, 12, 'G7'), (6, 7, 'G8'), (6,
 8, 'G8'), (9, 11, 'G9')]
```

Then starting node should be entered.

For example, dff0 is entered:

Directed Graph:



Then shortest path from DFF0 to all other nodes is printed.

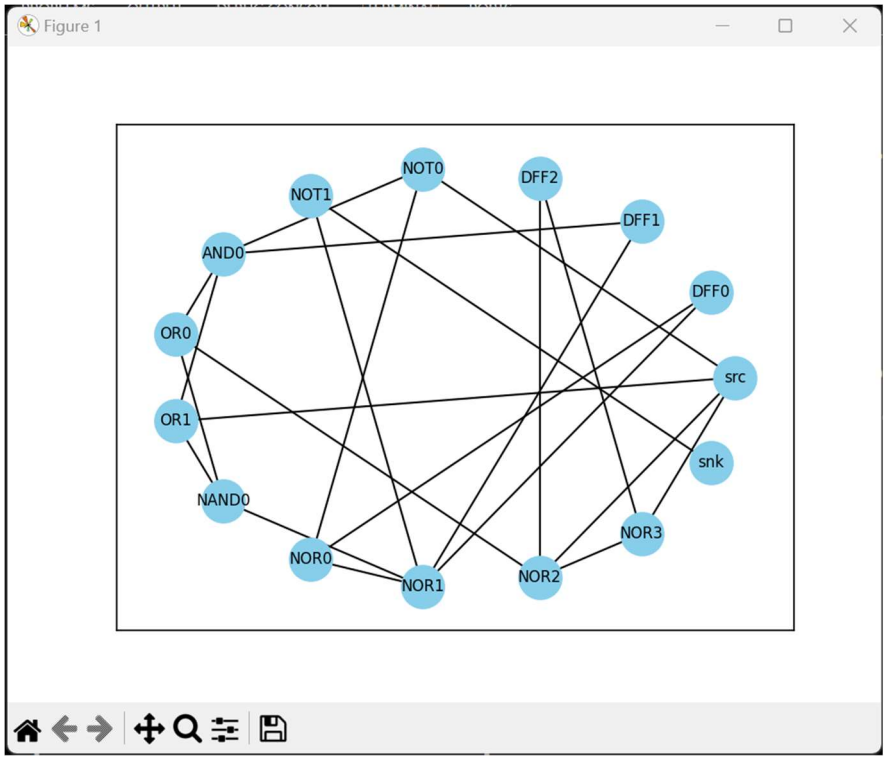
```
Shortest path from vertex DFF0
Vertex SRC: Distance = inf, Path = There is no path from DFF0 to SRC
Vertex DFF1: Distance = 2.0, Path = ['DFF0', 'NOR1', 'DFF1']
Vertex DFF2: Distance = inf, Path = There is no path from DFF0 to DFF2
Vertex NOT0: Distance = inf, Path = There is no path from DFF0 to NOT0
Vertex NOT1: Distance = 2.0, Path = ['DFF0', 'NOR1', 'NOT1']
Vertex AND0: Distance = 3.0, Path = ['DFF0', 'NOR1', 'DFF1', 'AND0']
Vertex OR0: Distance = 4.0, Path = ['DFF0', 'NOR1', 'DFF1', 'AND0', 'OR0']
Vertex OR1: Distance = 4.0, Path = ['DFF0', 'NOR1', 'DFF1', 'AND0', 'OR1']
Vertex NAND0: Distance = 5.0, Path = ['DFF0', 'NOR1', 'DFF1', 'AND0', 'OR0', 'NAND0']
Vertex NOR0: Distance = 2.0, Path = ['DFF0', 'NOR1', 'NOR0']
Vertex NOR1: Distance = 1.0, Path = ['DFF0', 'NOR1']
Vertex NOR2: Distance = inf, Path = There is no path from DFF0 to NOR2
Vertex NOR3: Distance = inf, Path = There is no path from DFF0 to NOR3
Vertex SNK: Distance = 3.0, Path = ['DFF0', 'NOR1', 'NOT1', 'SNK']
```

Suppose, if the entered choice was 2 (undirected graph):

```
1. Directed Graph
2. Undirected Graph
Enter your option: 2
Node List:
 ['SRC', 'DFF0', 'DFF1', 'DFF2', 'NOT0', 'NOT1', 'AND0', 'OR0', 'OR1', 'NAND0', 'NOR0', 'NOR1', 'NOR2', 'NOR3', 'SNK']
Edge List:
 [(0, 4, 'G0'), (0, 12, 'G1'), (10, 1, 'G10'), (11, 2, 'G11'), (11, 5, 'G11'), (11, 10, 'G11'), (12, 7, 'G12'), (12, 13, 'G12'), (13, 3, 'G13'), (4, 6, 'G14
'), (4, 10, 'G14'), (7, 9, 'G15'), (8, 9, 'G16'), (5, 14, 'G17'), (0, 13, 'G2'), (0, 8, 'G3'), (1, 11, 'G5'), (2, 6, 'G6'), (3, 12, 'G7'), (6, 7, 'G8'), (6,
 8, 'G8'), (9, 11, 'G9')]
Enter the starting node: dff0
```

Entered starting node is dff0.

Undirected Graph:



Shortest path from DFF0 to all other nodes is printed.

```
Shortest path from vertex DFF0
Vertex SRC: Distance = 3.0, Path = ['DFF0', 'NOR0', 'NOT0', 'SRC']
Vertex DFF1: Distance = 2.0, Path = ['DFF0', 'NOR1', 'DFF1']
Vertex DFF2: Distance = 5.0, Path = ['DFF0', 'NOR0', 'NOT0', 'SRC', 'NOR2', 'DFF2']
Vertex NOT0: Distance = 2.0, Path = ['DFF0', 'NOR0', 'NOT0']
Vertex NOT1: Distance = 2.0, Path = ['DFF0', 'NOR1', 'NOT1']
Vertex AND0: Distance = 3.0, Path = ['DFF0', 'NOR1', 'DFF1', 'AND0']
Vertex OR0: Distance = 3.0, Path = ['DFF0', 'NOR1', 'NAND0', 'OR0']
Vertex OR1: Distance = 3.0, Path = ['DFF0', 'NOR1', 'NAND0', 'OR1']
Vertex NAND0: Distance = 2.0, Path = ['DFF0', 'NOR1', 'NAND0']
Vertex NOR0: Distance = 1.0, Path = ['DFF0', 'NOR0']
Vertex NOR1: Distance = 1.0, Path = ['DFF0', 'NOR1']
Vertex NOR2: Distance = 4.0, Path = ['DFF0', 'NOR0', 'NOT0', 'SRC', 'NOR2']
Vertex NOR3: Distance = 4.0, Path = ['DFF0', 'NOR0', 'NOT0', 'SRC', 'NOR3']
Vertex SNK: Distance = 3.0, Path = ['DFF0', 'NOR1', 'NOT1', 'SNK']
```

It can be seen that there the shortest paths in both directed graph and undirected graph are different. This is because there is no directed edge from DFF0 to src, DFF2, NOT2, NOR2 and NOR3.