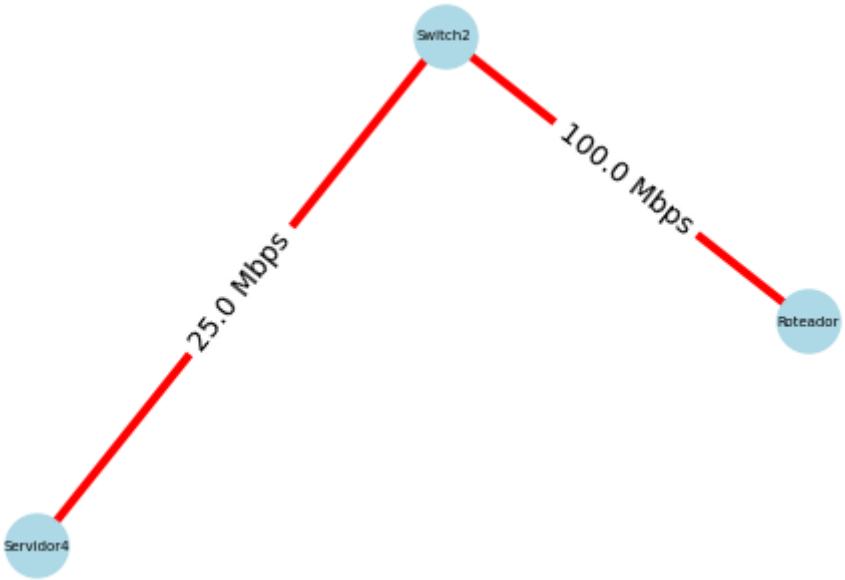
```python
# A.)

import networkx as nx
import matplotlib.pyplot as plt

grafo = nx.Graph()

# nós dispositivos
dispositivos = [
    "Roteador", "Switch1", "Switch2", "Servidor1", "Servidor2",
"Servidor3", "Servidor4", "Servidor5", "Servidor6"
]
grafo.add_nodes_from(dispositivos)

# arestas com pesos (inverso largura de banda, latência)
conexoes = [
    ("Roteador", "Switch1", 1/100),
    ("Roteador", "Switch2", 1/100),
    ("Switch1", "Servidor1", 1/50),
    ("Switch1", "Servidor2", 1/50),
    ("Switch2", "Servidor3", 1/25),
    ("Switch2", "Servidor4", 1/25),
    ("Switch1", "Servidor5", 1/50),
    ("Switch2", "Servidor6", 1/25)
]
for origem, destino, peso in conexoes:
    grafo.add_edge(origem, destino, weight=peso)

# solicita origem e destino
print("Dispositivos disponíveis:", dispositivos)
origem = input("Digite o dispositivo de origem: ")
destino = input("Digite o dispositivo de destino: ")

# verifica se o caminho é válido
if origem not in dispositivos or destino not in dispositivos:
    print("Dispositivo de origem ou destino inválido.")
else:
    try:
        # calcula o caminho mais curto usando Dijkstra
        caminho_curto = nx.dijkstra_path(grafo, source=origem,
target=destino, weight="weight")
        print(f"O caminho mais curto entre {origem} e {destino} é:
{caminho_curto}")

        # representação em grafos mostrando apenas o caminho
solicitado
        pos = nx.spring_layout(grafo)
        plt.figure(figsize=(6, 4))
```

```
        # Visualização dos nós e arestas no caminho
        caminho_edges = list(zip(caminho_curto, caminho_curto[1:]))
        nx.draw_networkx_nodes(grafo, pos, nodelist=caminho_curto,
node_size=500, node_color="lightblue")
        nx.draw_networkx_edges(grafo, pos, edgelist=caminho_edges,
width=3, edge_color="red")
        nx.draw_networkx_labels(grafo, pos, font_size=5,
font_family="sans-serif")

        # Visualização dos pesos (largura de banda)
        nx.draw_networkx_edge_labels(
            grafo, pos, edge_labels={(origem, destino):
f"{1/peso:.1f} Mbps" for origem, destino, peso in conexoes if
(origem, destino) in caminho_edges or (destino, origem) in
caminho_edges}
        )

        plt.title(f"Caminho Mais Curto entre {origem} e {destino}")
        plt.axis("off")
        plt.show()

    except nx.NetworkXNoPath:
        print(f"Não há caminho disponível entre {origem} e
{destino}.")
```



```python
# B)

import networkx as nx
import matplotlib.pyplot as plt

grafo = nx.Graph()

# nós dispositivos
dispositivos = [
    "Roteador", "Switch1", "Switch2", "Switch3", "Switch4",
"Servidor1", "Servidor2", "Servidor3", "Servidor4", "Servidor5",
"Servidor7", "Servidor8", "Servidor9", "Servidor10"
]
grafo.add_nodes_from(dispositivos)

# arestas com pesos (inverso largura de banda, latência)
conexoes = [
    ("Roteador", "Switch1", 1/100),
    ("Roteador", "Switch2", 1/80),
    ("Roteador", "Switch3", 1/60),
    ("Roteador", "Switch4", 1/40),
    ("Switch1", "Servidor1", 1/10),
    ("Switch1", "Servidor2", 1/15),
    ("Switch2", "Servidor3", 1/20),
    ("Switch2", "Servidor4", 1/25),
    ("Switch3", "Servidor5", 1/30),
    ("Switch3", "Servidor6", 1/35),
    ("Switch4", "Servidor7", 1/40),
    ("Switch4", "Servidor8", 1/45),
    ("Switch1", "Servidor9", 1/10),
    ("Switch2", "Servidor10", 1/15)
]
for origem, destino, peso in conexoes:
    grafo.add_edge(origem, destino, weight=peso)

# solicita origem e destino
print("Dispositivos disponíveis:", dispositivos)
origem = input("Digite o dispositivo de origem: ")
destino = input("Digite o dispositivo de destino: ")

# verifica se o caminho é válido
if origem not in dispositivos or destino not in dispositivos:
    print("Dispositivo de origem ou destino inválido.")
else:
    try:
        # calcula o caminho mais curto usando Dijkstra
        caminho_curto = nx.dijkstra_path(grafo, source=origem,
target=destino, weight="weight")
```

```
        print(f"O caminho mais curto entre {origem} e {destino} é:
{caminho_curto}")

        # representação em grafos mostrando apenas o caminho
solicitado
        pos = nx.spring_layout(grafo)
        plt.figure(figsize=(6, 4))

        # Visualização dos nós e arestas no caminho
        caminho_edges = list(zip(caminho_curto, caminho_curto[1:]))
        nx.draw_networkx_nodes(grafo, pos, nodelist=caminho_curto,
node_size=500, node_color="lightblue")
        nx.draw_networkx_edges(grafo, pos, edgelist=caminho_edges,
width=3, edge_color="red")
        nx.draw_networkx_labels(grafo, pos, font_size=5,
font_family="sans-serif")

        # Visualização dos pesos (largura de banda)
        nx.draw_networkx_edge_labels(
            grafo, pos, edge_labels={(origem, destino):
f"{1/peso:.1f} Mbps" for origem, destino, peso in conexoes if
(origem, destino) in caminho_edges or (destino, origem) in
caminho_edges}
        )

        plt.title(f"Caminho Mais Curto entre {origem} e {destino}")
        plt.axis("off")
        plt.show()

    except nx.NetworkXNoPath:
        print(f"Não há caminho disponível entre {origem} e
{destino}.")
```