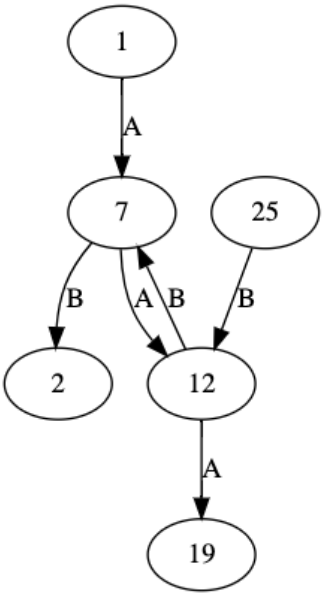
Graphs (with graphviz)

The illustration to the right shows a "graph"

- each circle is a "node"
- each line is an "edge"
- arrows means edges are "directed" (we would call this a "directed graph")
- Image we are showing two bus routes, A and B. They share some transfer points (stops 7 and 12).



Drawing graphs in Python:

- https://pypi.org/project/graphviz/
- pip install graphviz
- install: https://www.graphviz.org/download/ (should be able to run **`dot --help`** from shell)
- we need Digraph for directed graphs: from graphviz import Digraph

Using Digraph.node(...) and Digraph.edge(...)

```
1  from graphviz import Digraph
2
3  # TODO: delete this if computed from earlier...
4  routes = {'A': [1, 7, 12, 19], 'B': [25, 12, 7, 2]}
5
6  stops = []
7  for rstops in routes.values():
8      stops.extend(rstops)
9  stops = sorted(set(stops))
10
11 g = Digraph()
12
13 # draw nodes
14 for stop in stops:
15     g.node(str(stop))
16
17 # draw edges
18 for route, stops in routes.items():
19     for i in range(len(stops) - 1):
20         g.edge(str(stops[i]), str(stops[i+1]), str(route))
21
22 g
```

Filtering All Stops to Endpoints and Transfers (run before pg 1 code)

```python
from collections import defaultdict

# KEY=route, VAL=list of stops
routes = {
    "A": [1, 5, 7, 9, 12, 19],
    "B": [25, 12, 11, 7, 2]
}

stop_counts = defaultdict(int)
important = set() # end points and transfer points

for stops in routes.values():
    # end points
    important.add(stops[0])
    important.add(stops[-1])
    for stop in stops:
        stop_counts[stop] += 1

# transfer points appear twice
for stop in stop_counts:
    if stop_counts[stop] > 1:
        important.add(stop)

# filter original routes to only include important points
for k in routes:
    routes[k] = [stop for stop in routes[k] if stop in important]
routes
```

Node and edge styles

- call before .edge and .node calls
- g.attr('node', shape="square")
- g.attr('node', width="0", height="0", margin="0.05")
- g.attr('edge', penwidth="3")
- g.attr('edge', color="#FF00FF")
- note that all arguments are strings
- A color #RRGGBB gives red, green, blue mixes. It uses hex (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F). This is base 16. A is 10 (decimal), F is 15 (decimal). So FF is maximum for a given color.
- See more docs here: https://www.graphviz.org/doc/info/attrs.html (UsedBy of "ENG" means attr can be set for edge, node, and graph)