B [1]:
```
import random

import networkx as nx
import matplotlib.pyplot as plt

def max_flow():
    G = nx.DiGraph()
    G.add_edge("1", "2", weight=9)
    G.add_edge("1", "3", weight=11)
    G.add_edge("1", "4", weight=15)
    G.add_edge("2", "4", weight=6)
    G.add_edge("2", "5", weight=11)
    G.add_edge("3", "4", weight=4)
    G.add_edge("3", "6", weight=13)
    G.add_edge("4", "5", weight=7)
    G.add_edge("4", "6", weight=5)
    G.add_edge("4", "7", weight=10)
    G.add_edge("5", "8", weight=17)
    G.add_edge("6", "7", weight=3)
    G.add_edge("6", "8", weight=12)
    G.add_edge("7", "8", weight=9)
    print(nx.dag_longest_path(G, weight="weight"))
    print(nx.dag_longest_path_length(G, weight="weight"))
    nx.draw(G)
    plt.show()
    return 1


def min_flow_time():
    G = nx.DiGraph()
    x = 150/8

    G.add_edge("1", "2", weight=10*(1 - 0.02*x))
    G.add_edge("1", "3", weight=20*(1 - 0.04*x))
    G.add_edge("2", "3", weight=12*(1 - 0.03*x))
    G.add_edge("2", "4", weight=14*(1 - 0.06*x))
    G.add_edge("3", "4")
    G.add_edge("3", "5", weight=16*(1 - 0.05*x))
    G.add_edge("4", "3")
    G.add_edge("4", "5", weight=6*(1 - 0.01*x))
    print(nx.shortest_path(G, "1", "5", weight="weight"))
    print(nx.shortest_path_length(G, "1", "5", weight="weight"))
    nx.draw(G)
    plt.show()
    return 1

def min_flow_cost():
    T = 1.5
    C=50
    G = nx.DiGraph()
    G.add_edge("1", "2", weight=35+C - T*2*2)
    G.add_edge("1", "3", weight=22+C - T*4*1.5)
    G.add_edge("2", "3", weight=45+C - T*6*8)
    G.add_edge("2", "4", weight=32+C - T*4*6)

    G.add_edge("3", "5", weight=24+C - T*6*3)
```

```
    G.add_edge("4", "3")
    G.add_edge("4", "5", weight=65+C - T*3*2.5)
    print(nx.dag_longest_path(G, weight="weight"))
    print(nx.dag_longest_path_length(G, weight="weight"))
    print("optimised")
    G.add_edge("3", "4")
    print(nx.shortest_path(G, "1", "5", weight="weight"))
    print(nx.shortest_path_length(G, "1", "5", weight="weight"))
    print(T * len(nx.shortest_path(G, "1", "5", weight="weight")))

if __name__ == '__main__':
    #max_flow()
    #min_flow_time()
    min_flow_cost()
```

```
['1', '2', '4', '5']
228.75
optimised
['1', '3', '5']
110.0
4.5
```

B [ ]: