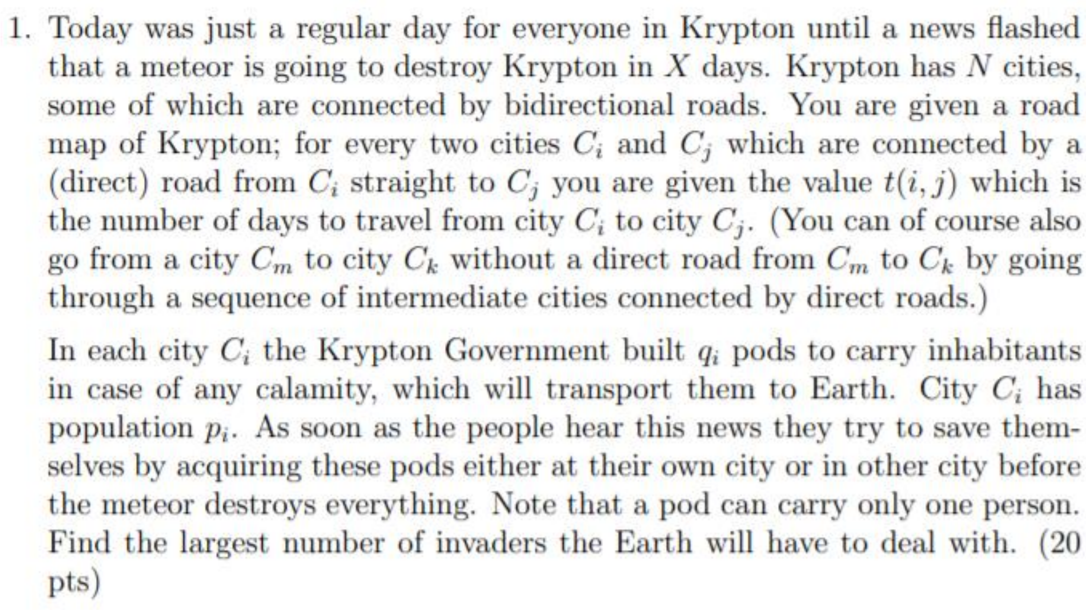
HOMEWORK 5 – Q1

MINGLANG XIE

z5228006



1. Today was just a regular day for everyone in Krypton until a news flashed that a meteor is going to destroy Krypton in X days. Krypton has N cities, some of which are connected by bidirectional roads. You are given a road map of Krypton; for every two cities C_i and C_j which are connected by a (direct) road from C_i straight to C_j you are given the value $t(i,j)$ which is the number of days to travel from city C_i to city C_j. (You can of course also go from a city C_m to city C_k without a direct road from C_m to C_k by going through a sequence of intermediate cities connected by direct roads.)

   In each city C_i the Krypton Government built q_i pods to carry inhabitants in case of any calamity, which will transport them to Earth. City C_i has population p_i. As soon as the people hear this news they try to save themselves by acquiring these pods either at their own city or in other city before the meteor destroys everything. Note that a pod can carry only one person. Find the largest number of invaders the Earth will have to deal with. (20 pts)

Solution:

This is a typical max flow problem.
First, use BFS algorithm, for each city C_i find the set of cities you can reach within X days.

```
def BFS (graph, vertex, X):
    que         # queue
    que.push(vertex, X)
    vertex_trip = []
    while que.notEmpty():
        temp, X = que.pop()

        if (tmp.NotVisited()):
            for all edge connected to tmp and day_to_travel < remain_day:
                que.push(newVertex, X - day_to_travel)
            set vertex visited
            vertex_trip.append(temp)
    return vertex_trip

for all vertex in Graph:
    vertex_trip[vertex] = BFS (Graph, vertex, X)
```

Make a bipartite graph with vertices corresponding to all cities both on the left and on the right side but with different interpretation: on the left vertices represent populations of the corresponding cities; on the right the vertices represent the set of pods in the corresponding cities. Introduce a super source and a super sink and connect each city C_i on the left side with the super source by a directed edge of capacity equal to the p_i. Connect each city C_i on the right side with the super sink with a directed edge of capacity equal to q_i. Further, connect each city C_i with the set of cities that you can reach within X days from city C_i, and using directed edges of capacity equal to q_i. Now just use max flow algorithm and look at occupied edges to determine largest number of invaders the Earth will have to deal with.