1. (10 points) Several families go out to dinner together. To increase their social interaction, they would like to sit at tables so that no two members of the same family are at the same table. Show how to formulate finding a seating arrangement that meets this objective as a max flow problem. Assume that the dinner contingent has *p* families and the i_{th} family has *a(i)* members. Also assume that *q* tables are available and that the j_{th} table has a seat capacity of *b(j)*. Please describe the formulation in network flow model and in mathematical form.

(u, v) means one directed edge from node u to node v.

Capacity (u, v) means capacity of this directed edge.

Flow (u, v) means flow passing through this directed edge.

V: the set includes all nodes

E: the set includes all edges

Node s: source

Node t: target

Capacity (s, family(i)) = a(i), $i \in \{1,2,3,\dots,p\}$

Capacity (family(i), table(j)) = 1, $i \in \{1,2,3,\dots,p\}, j \in \{1,2,3,\dots,q\}$

Capacity (table(j), t) = b(j), $j \in \{1,2,3,\dots,q\}$

In this graph, (u, v) is any nodes pair which is connected by one directed edge.

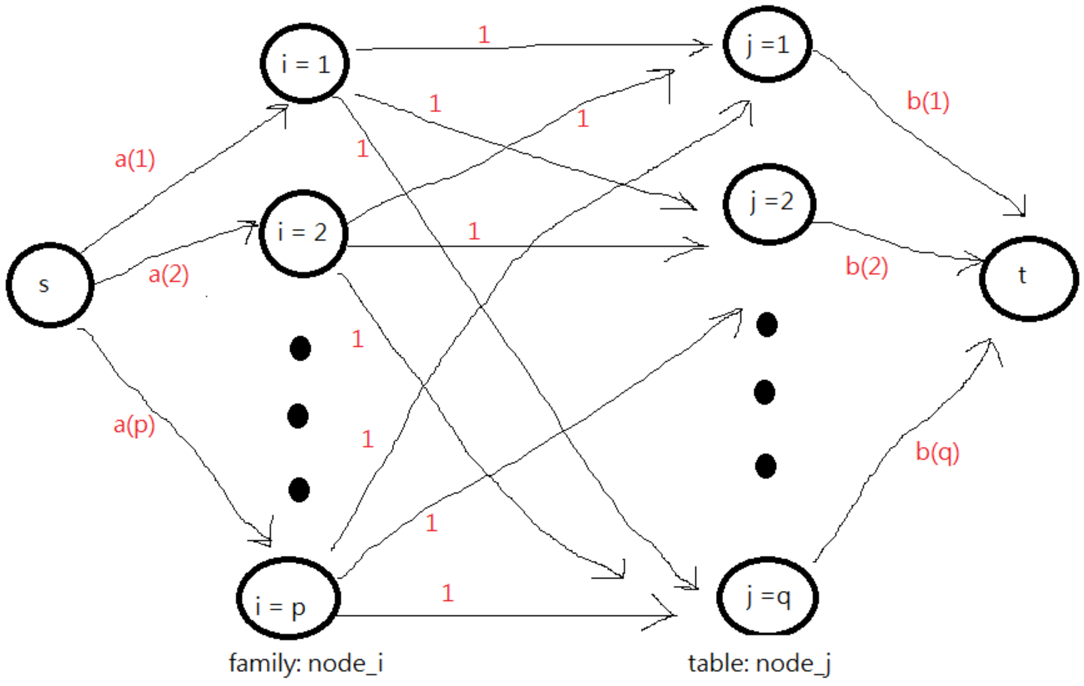
Maximize Objective Function (**total flow from source**):

$$\sum_{v:(s,v)\in E} Flow(s,v)$$

Under **two constraints** (**capacity limit & input flow equals output flow at V-{s,t}**)

$$0 \le \mathrm{Flow(u,v)} \le \mathrm{Capacity(u,v)}$$

$$\sum_{u:(u,v)\in E} Flow(u,v) = \sum_{u:(v,u)\in E} Flow(v,u) \quad ,v \in V - \{s,t\}$$



2. (20 points) Consider a graph with vertices V = {1, 2, 3, 4, 5, 6, 7, 8}, undirected edges and their associated weights are

(6, 1)	7
(6, 3)	3
(1, 3)	4
(1, 2)	3
(1, 4)	7
(1, 7)	8
(2, 3)	4
(2, 4)	1
(2, 5)	8
(2, 7)	3
(3, 4)	6
(4, 5)	2
(5, 7)	4
(5, 8)	6
(4, 7)	5
(7, 8)	9

Please write C/C++ program of the Dijkstra's algorithm to find the shortest path from vertex 6 to vertex 8. Please report both the edges along the path and the path length. You are required to submit your source code, which should be able to compile at Linux system.

Because there are some undirected edge loops exist in this graph. I need to add a termination condition which is all labels of nodes still remain the same as those in the previous while loop.

Also, the page9, the last 2 lines, which enqueues v to Q, should execute when label(v) is update to smaller number. Otherwise, it will be struck into node 3, node 1, and node 2 enqueue/dequeue loops.

Find the shortest path: node 6➔node3➔node2➔node4➔node5➔node8

the length of node 6➔node3: 3

the length of node 3➔node2: 4

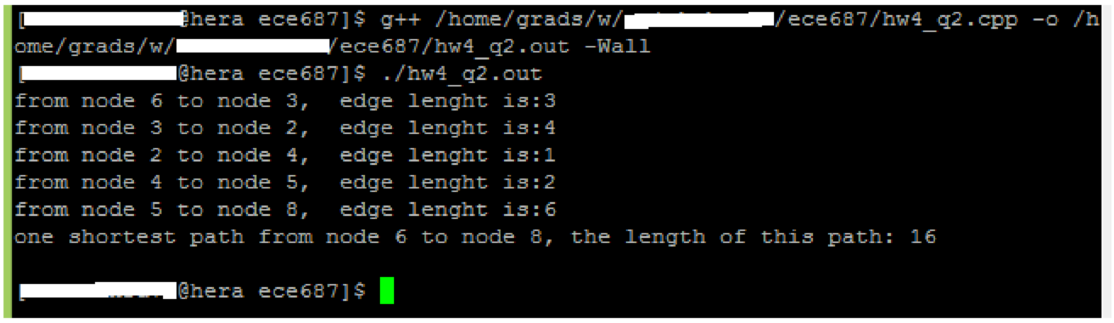
the length of node 2➔node4: 1

the length of node 4➔node5: 2

the length of node 5➔node8: 6

the length of this shortest path is 16.

Simulation result:



```
[[redacted]@hera ece687]$ g++ /home/grads/w/[redacted]/ece687/hw4_q2.cpp -o /h
ome/grads/w/[redacted]/ece687/hw4_q2.out -Wall
[[redacted]@hera ece687]$ ./hw4_q2.out
from node 6 to node 3,  edge lenght is:3
from node 3 to node 2,  edge lenght is:4
from node 2 to node 4,  edge lenght is:1
from node 4 to node 5,  edge lenght is:2
from node 5 to node 8,  edge lenght is:6
one shortest path from node 6 to node 8, the length of this path: 16

[[redacted]@hera ece687]$ 
```