EXERCISE 1: Seam Carving

Consider the given 3x4 image and the corresponding energies matrix.

- A *Vertical Seam* is a path of pixels connected from the top row to the bottom row , where a pixel at column x and row y can only be connected to the pixels $(x-1, y+1)$, $(x, y+1)$ and $(x+1, y+1)$.
- The *Seam Energy* is the sum of the energies of the pixels in the seam.
- A *Minimum Energy Vertical Seam* is the vertical seam with the minimum energy.

(15,10,16)	(31,15,19)	(15,10,3)
(5,18,0)	(80,18,0)	(120,100,80)
(35,20,12)	(36,17,13)	(15,10,3)
(5,1,13)	(13,1,16)	(120,110,40)

RGB Values of the 3x4 Image

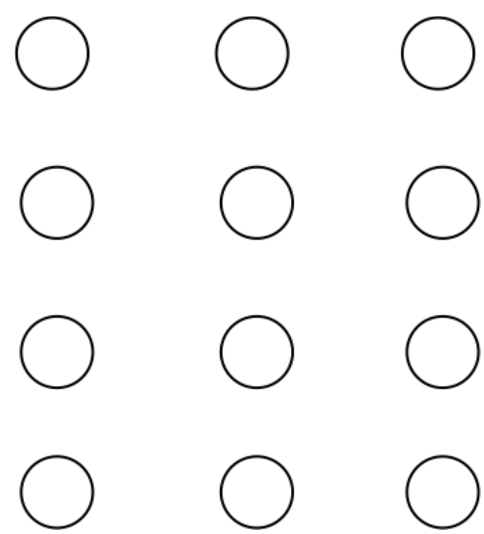
32	72	45
123	163	75
32	75	41
156	161	9

Energy Values (Rounded)

A. Mark the minimum energy vertical seam in the given energies matrix. What is the energy of this seam?

B. In order to find the minimum energy vertical seam, you will have to find the shortest path from any pixel in the top row to any pixel in the bottom row.

Draw the implicit graph which the energies matrix represents. Show all the edges and edge weights.

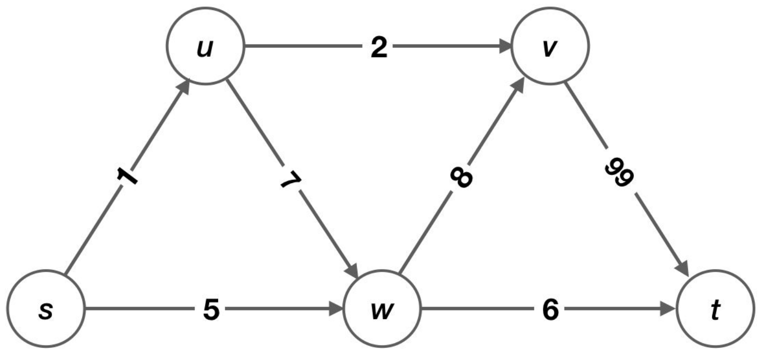


C. Assume that the image is of size $W \times H$, what is the order of growth of the running time of finding the minimum energy vertical Seam using ***Dijkstra's*** algorithm (use *W* and *H*)?

EXERCISE 2: Shortest Teleport Path (Design Question)

Given an edge-weighted digraph G with nonnegative edge weights, a source vertex s and a destination vertex t, find a shortest path from s to t where you are permitted to teleport across one edge for free. That is, the weight of a path is the sum of the weights of all of the edges in the path, excluding the largest one.

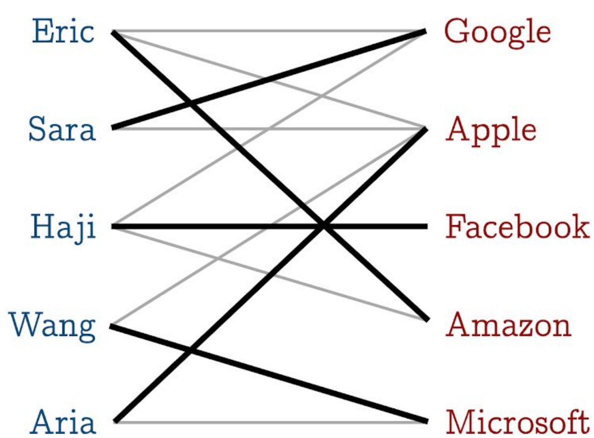
Example. In the edge-weighted digraph below, the shortest path from s to t is $s \to w \to t$ (with weight 11) but the the shortest teleport path is $s \to u \to v \to t$ (with weight 3).



EXERCISE 3: Bipartite Matching (Design Question)

Consider the set X of job applicants and the set Y of companies with job openings. Consider also the set of job offers (x_i, y_j) sent out by the companies to the applicants. Design an algorithm for finding which job offers should be accepted so that every company hires one applicant and every applicant is hired by one company.

As shown in the example below, this problem can be modeled as a *Bipartite Graph*, where edges represent job offers. Accepted offers are marked in bold.

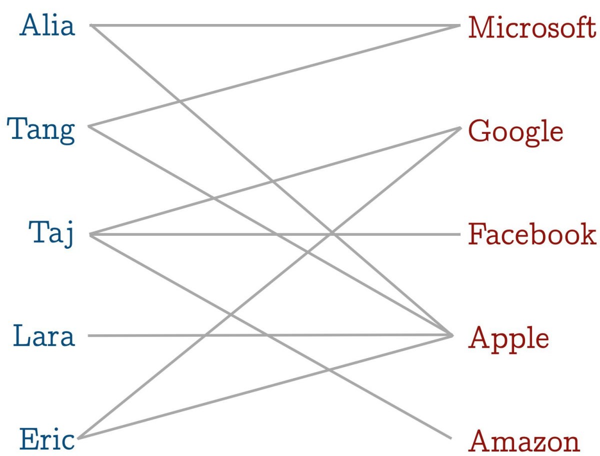


Perfect Matching

A. Model the problem as a **Max-Flow** problem by converting the bipartite graph into a flow network. Show your work by drawing the flow network corresponding to the "Perfect Matching" example above.

- Mark the `source` and `sink` vertices and the edge `capacities`.
- Describe how a max-flow in the network corresponds to a matching in the bipartite graph.

B. Show the result of using ***Ford-Fulkerson*** to find the maximum matching for the following (*different*) instance of the problem.



C. What is the order of growth of the running time of the algorithm as a function of the number of edges E and the number of vertices V in the bipartite graph?

D. Find the ***minimum-cut*** in the max-flow network you have computed in part (B).