Programming assignment 2

Consider a set of pictures from 360° camera mounted inside a merry-go-round. They were taken at night and only one seat is visible which emits light – the seat in the shape of a jelly fish. We know that the merry-go-round rotates clockwise and that all the pictures were taken during a single cycle of merry-go-round. Given that the first picture is P1, your task is to sort the rest in the chronological order. Assume that the move of jelly fish is monotonous in horizontal direction, i.e., it always moves in clockwise direction, never backwards.

Input files: text files, each of them with 10 rows and 80 columns representing the brightness level in the given parts of the picture. Jelly fish can be recognized by high brightness (value 1 to 9) on a black background (value 0).

```
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000005950000000000000000000000000000000000000000
00000000000000000000000000000000000009990000000000000000000000000000000000000000
00000000000000000000000000000000000005950000000000000000000000000000000000000000
00000000000000000000000000000000000003930000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000
```

Directions

In order to capture the movement of the jelly fish between two pictures, use Earth-mover distance, also known as Wasserstein distance. It can be formulated as a min-cost flow problem on a certain network. Use NetworkX package (https://networkx.org) to perform the optimization required to find this flow.

About Earth-mover distance: Consider two distributions over a discrete set of points P: μ in [0,1]^P and λ in [0,1]^P. Since both are distributions and sum of μ(p) over all points in P is 1 (and the same holds for λ), there is surely a way to transform μ into λ: we denote $f_{p,q}$ the probability mass transfered from point p to q for each pair of points p,q from P, in order to transform μ into λ:

For each p in P: $\mu(p) = \Sigma_q f_{p,q}$ and for each q in P: $\lambda(q) = \Sigma_p f_{p,q}$.

Note that $f_{p,p}$ denotes the probability mass which remains at the same place.
Let d(p,q) denote the distance between p and q. Then, the cost of the transformation f is

$\Sigma_{p,q}\, d(p,q)\, f_{p,q}$

The earth-mover distance of μ and λ is then the cost of the cheapest transformation between μ and λ.

Technical instructions:

Your code should implement function sort_files() which returns ordered list of the pictures, something like this:

```
['P1.txt', 'P2.txt', 'P3.txt', 'P10.txt', 'P4.txt', 'P13.txt', 'P14.txt', 'P9.txt',
  'P15.txt', 'P7.txt', 'P12.txt', 'P11.txt', 'P5.txt', 'P6.txt', 'P8.txt']
```

You should also define and use function comp_dist(file1, file2) which returns the Earth-mover distance between file1 and file2. I may want to access it during the grading process, especially if your output is not completely correct.

Make sure your code never breaks down. If you know that it cannot handle e.g. P10.txt, skip this file and don't include it in the output. This way, you will not get a full score, but you will still get points for the partial solution.

Before submission, always try your code in vocareum using "Run" button. Incorrect output format, syntax and indentation errors, etc will result into 0 points.

files.zip