Today we will implement the Viterbi algorithm for finding the most likely hidden states path corresponding to a list of observables:

1. Initialize the first row of a matrix V with initialProbabilities* emission probabilities
2. Write a loop that computes for V a row of probabilities for each observable.
   The equation is max(previous_row_probability*transition_probability*emission probability)
   The matrix should contain a column for each state. You should save the most likely previous state and the probability for every state. This can be done by having two matrices
3. Return the most probable state squence, finding the most probable state in the last row and then tracking the most probable states that led to it.
4. Put the algorithm you wrote in a class with a
   a. A constructor (__init__) function that accepts the initial probabiities, the transition matrix and the emission matrix
   b. An “assert” statement to make sure that the input values are reasonable: http://stackoverflow.com/questions/5142418/what-is-the-use-of-assert-in-python
   c. A “run” function that accepts a sequence of observables and returns a sequence of internal states
5. Run the algorithm on the initial data from the following link: https://github.com/phvu/misc/blob/master/viterbi/test1.py
6. What is dynamic programming and why is Viterbi considered Dynamic programming https://en.wikipedia.org/wiki/Dynamic_programming

Reference for implementation:
Psedo-code: https://en.wikipedia.org/wiki/Viterbi_algorithm
Code:
https://github.com/phvu/misc/tree/master/viterbi