PART A - Umbrella domain

1) The set of unobserved variable (only one) for a given time slice t is in our case the variable R_t (RAIN) which denotes if it rains or not that day.

2) The observable variable is U_t (Umbrella) which denotes if the director brings an umbrella or not.

3) Transition model/ dynamic model P(X_t | X_(t-1))
[[0.7 0.3]
[0.3 0.7]]

The observation model P(E_t | X_t) have matrices: if the umbrella is observed and if the umbrella is not. The matrices show for each state (rain or no), how likely it is that the state caused umbrella to be brought.

[[0.9 0.0]
[0.0 0.2]]
We have observed umbrella today, huzzah!
(https://en.wikipedia.org/wiki/Huzzah)

[[0.1 0.0]
[0.0 0.8]]
Umbrella.exe was not found.

4) Assumptions of this model:
   a) **Markov assumption:** (first order) Each next state is only dependent on a fixed number of previous states, in this case only the previous state.
      → Reasonable as rain one day often means rain the next. We could possibly increase accuracy by having 2nd or 3rd order as we often see periods of rain of this length.
   b) **Stationary process**: States change due to a stationary process, which is a process governed by unchanging rules.
      → Rain is ultimately governed by facts such as humidity, temperature, air pressure and such. These rules don't change over time, and it is a safe assumption to make.
   c) **Sensor markov assumption:** Sensory data/observable data only depends on current state.
      i) The eyes of the observer is not affected by previous rainfall (I hope) and it is therefore safe to assume observations are independent of previous rainfall as well.

PART B - FORWARD IMPLEMENTATION

```
Choose test from menu:
 0 - Exit
 1 - B.2: verify implementation of 'Forward'
 2 - nothing
-------------------------
Answer: 1
Testing forward-algorithm
Probability of raining on day 1 [[0.81818182]]
Probability of raining on day 2 [[0.88335704]]
Probability of raining on day 3 [[0.19066794]]
Probability of raining on day 4 [[0.730794]]
Probability of raining on day 5 [[0.86733889]]
```

```
observations = [True, True, False, True, True]  # Used umbrella both days
prob_raining = np.matrix("0.5; 0.5")            # Starting assumption (same as made in book)
```

Documentation of results after running the test. Test is located in the main() function.

PART C - FORWARD BACKWARD

```
observations = [True, True, False, True, True]  # A given set of observations
prob_raining = np.matrix("0.5; 0.5")            # Starting assumption (same as made in book)
smooth_vector = forwardBackward(observations, prob_raining)
```

```
Smooth vector: [matrix([[0.5],
        [0.5]]), matrix([[0.86733889],
        [0.13266111]]), matrix([[0.82041905],
        [0.17958095]]), matrix([[0.30748358],
        [0.69251642]]), matrix([[0.82041905],
        [0.17958095]]), matrix([[0.86733889],
        [0.13266111]])]
-------------------------
```

After running the forward-backward algorithm with 5 observations we got this smoothed-vector. Day 1 predictions are slightly different from what we achieved in the last part, using filtering.

```
Backward message nr 1
 [[1]
 [1]]
Backward message nr 2
 [[0.69]
 [0.41]]
Backward message nr 3
 [[0.4593]
 [0.2437]]
Backward message nr 4
 [[0.090639]
 [0.150251]]
Backward message nr 5
 [[0.06611763]
 [0.04550767]]
```

Here are all the backward-vectors that were produced during the algorithm, in order of being produced.