Problem Wk.11.1.1: Observation Models

In this problem, we look at defining observation models. The first three problems ask you to define conditional distributions on the color that the robot will observe in a room, given the actual color of that room. Be sure to read the Design Lab 11 handout before working on this problem.

Look at the software documentation for the dist module. Throughout this problem, the dist module has been imported, so you can get at all the contents by using dist.x.

You can assume that the following functions and variables, described in the lab handout, are already defined:

- makeObservationModel(hallwayColors, obsDist) -- see lab handout.
- standardHallway -- list of the colors of each room in the hallway, going from left to right. In this problem, this is
  ['white', 'white', 'green', 'white', 'white'].
- possibleColors -- list of all the possible colors that could be observed. In this problem, this is
  ['black', 'white', 'red', 'green', 'blue']

In your answers, feel free to define any auxiliary functions that you may need.

---

**Part 1: White == Green**

Define an observation noise model (that is, a conditional distribution over the observed color given the actual color of a room) in which white and green are indistinguishable, in the sense that the robot is just as likely to see white as to see green when it is in a white square or a green square, but where all other colors are observed perfectly. This function should return a dist.DDist over observed colors, given the actual color passed in as an argument.

```python
def whiteEqGreenObsDist(actualColor):
    pass
```

---

**Part 2: White <-> Green**

Give an observation noise distribution in which white always looks green, green always looks white, and all other colors are observed perfectly.
**Part 3: Noisy**

Define an observation noise distribution in which there is a probability of 0.8 of observing the actual color of a room, and there is a probability of 0.2 of seeing one of the remaining colors (equally likely which other color you see). All possible colors are available in the list called `possibleColors`.

```python
def whiteVsGreenObsDist(actualColor):
    pass
```

**Part 4: Observation Models**

The observation noise distributions that you have defined so far encode the error model for observation in a way which is independent of location. But, ultimately, we need to be able to obtain a probability distribution over the possible observations (colors) at a particular location along the hallway. The function `makeObservationModel` allows us to construct such a model, given a list of colors, such as `standardHallway`, and an observation noise distribution, such as you've written above.

Enter the expression for creating the full observation model for the `standardHallway` (assume it's already defined), with the observation noise distribution `noisyObs` (assume it's already defined).

```python
noisyObsModel = None
```