Exercises: Chapter 3

Exercise 3.1: Fair Coins and Biased Coins

Question 3.1.1

I flip a fair coin. What is the probability that it lands heads?

```
var model = function() {
  // Your code here
}

var logProb = Infer(
  {method:'enumerate'},
  model
).score('H')
Math.exp(logProb)
```

Question 3.1.2

I also have a biased coin, with Pr(Heads) = 0.9. I hand you one of the coins (either biased or fair) chosen uniformly randomly without telling you which. You flip it three times.

Given that first two coin flips landed on heads, what is the posterior distribution for the next flip?

```
var model = function() {
  // Your code here
}

viz.table(
  Infer({method:'enumerate'}, model)
)
```

Question 3.1.3

Given that all three flips landed on heads, what is the probability that the coin was biased?

```
var model = function() {
  // Your code here
}

viz.table(
  Infer({method:'enumerate'}, model)
)
```

Question 3.1.4

Given that the first two flips were different, what is the probability that the third flip will be heads?

```
var model = function() {
  // Your code here
}

viz.table(
  Infer({method:'enumerate'}, model)
)
```

Exercise 3.2: Conditioning and Intervention

In the earlier Medical Diagnosis section we suggested understanding the patterns of symptoms for a particular disease by changing the prior probability of the disease such that it is always true (also called the **do()* operator**):

```
var lungCancer = flip(0.01);
var cold = flip(0.2);
var cough = (
  (cold && flip(0.5))
  || (lungCancer && flip(0.3))
)
cough
```

Question 3.2.1

Show that *intervening* (setting) on `lungCancer` has the same effect as *conditioning* on `lungCancer` has the same effect on `cough` in this example.

Create a table showing the marginal probabilities. What must be true about the causal structure for this to be the case?

```
// Your code here
```

Question 3.2.2

This time, modify the program so that intervening and conditioning produce different results. Create a table showing the marginal probabilities. Under what circumstances does intervening produce different results from conditioning?

Hint: you do not need to introduce any new variables. Think about what other questions you can ask in this example.

```
// Your code here
```

Exercise 3.3: Computing Marginals

Find the marginal distribution of the return values from these programs mathematically (by hand).

Question 3.3.1

```
viz.table(
  Infer(
    {method: "enumerate"},
    function() {
      var a = flip()
      var b = flip()
      condition(a || b)
      return a
    }
  )
)
```

Question 3.3.2

```
var smilesModel = function() {
  var nice = mem(
    function(person) { flip(.7) }
  )

  var smiles = function(person) {
    return nice(person)
      ? flip(.8)
      : flip(.5)
  }

  condition(
    smiles('alice')
    && smiles('bob')
    && smiles('alice')
  )
  return nice('alice')
}

viz.table(
  Infer({method: "enumerate"}, smilesModel)
)
```

Exercise 3.4: Extending the Smiles Model

Question 3.4.1

Describe (using ordinary English) the `smilesModel` program in Question 3.3.2

Question 3.4.2

Extend `smilesModel` to create a version of the model considers two additional factors:

1. People will smile 80% of the time if they want something from you and 50% if they do not.
2. *Nice* people will only want something from you 20% of the time; non-nice people 50% of the time.

Don't forget that nice people also smile more often!

Hint: Which variables change at different times for the same person? Which values *depend* on other values?

```
var extendedSmilesModel = function() {
  var nice = mem(
    function(person) { flip(.7) }
  )
  var smiles = function(person, wants) {
    return nice(person)
      ? flip(.8)
      : flip(.5)
  }
  return smiles('alice')
}

Infer(
  {method: "enumerate"},
  extendedSmilesModel
)
```

Question 3.4.3

Suppose you've seen Bob five times this week and each time, he was not smiling. But today, you see Bob and he *is* smiling. Use this `extendedSmilesModel` model to compute the posterior belief that Bob wants something from you today.

Hint: How will you represent the same person (Bob) smiling *multiple times*? What features of Bob will stay the same each time he smiles (or doesn't) and what features will change?

```
var extendedSmilesModel = function() {
  // copy your code frome above

  // make the appropriate observations

  // return the appropriate query
  return ...

viz.table(
  Infer(
    {method: "enumerate"},
    extendedSmilesModel
```

```
  )
)
```

Exercise 3.5: Sprinklers and Rain

Question 3.5.1

I have a particularly bad model of the sprinkler in my garden. It is supposed to water my grass every morning, but is turns on only half the time (at random, as far as I can tell). Fortunately, I live in a city where it also rains 30% of days.

One day I check my lawn and see that it is wet, meaning that either it rained that morning or my sprinkler turned on (or both).

Answer the following questions, either using the Rules of Probability or by writing your own sprinkler model in WebPPL.

- What is the probability that it rained?
- What is the probability that my sprinkler turned on?

Question 3.5.2

My neighbour Kelsey, who has the same kind of sprinkler, tells me that her lawn was also wet that same morning. What is the new posterior probability that it rained?

Question 3.5.3

To investigate further we poll a selection of our friends who live nearby, and ask if their grass was wet this morning.

Kevin and Manu and Josh, each with the same sprinkler, all agree that their lawns were wet too.

Write a model to reason about all 5 people (including me and Kelsey), and then use it to find the probability that it rained:

```
// Your code here
```

Exercise 3.6: Casino Game

Consider the following game. A machine randomly gives Bob a letter L from the word "game" with different probabilities and Bob has a different probability of winning depending on which letter he got:

ℓ	$\Pr(L = \ell)$	$\Pr(\text{Win} \mid L = \ell)$	$\Pr(L = \ell \mid \text{Win})$
`g`	0.05	1	
`a`	0.45	1/4	
`m`	0.05	1/9	
`e`	0.45	1/16	

Suppose that we observe Bob winning, but we don't know what letter he got. How can we use the observation that he won to update our beliefs about which letter he got?

Let's express this formally. Before we begin, a bit of terminology: the set of letters that Bob could have gotten, $\{\texttt{g}, \texttt{a}, \texttt{m}, \texttt{e}\}$, is called the **hypothesis space** – it's our set of hypotheses about the letter.

Question 3.6.1

In English, what does the posterior probability $\Pr(L = \ell \mid \text{Win})$ represent? What does it mean for a letter to have the highest posterior?

Question 3.6.2

Manually calculate $\Pr(L = \ell \mid \text{Win})$ for each hypothesis in the table above.

Remember to normalize! Make sure that summing all of your $\Pr(L = \ell \mid \text{Win})$ values gives you 1.

Question 3.6.3

Now let's write this model in WebPPL using `Infer`.

Fill in the `...`'s in the code below to compute $\Pr(L = \ell \mid \text{Win})$, and visualize the resulting distribution.

It might be helpful to comment out the `condition` statement so you can compare visually the prior (no `condition` statement) to the posterior (with `condition`).

Make sure that your WebPPL answers and hand-computed answers agree – note that this demonstrates the equivalence between the program view of conditional probability and the distributional view.

```
// Define some variables and utility functions
var checkVowel = function(letter) {
  _.includes(['a', 'e', 'i', 'o', 'u'], letter)
}
var letterVals = ['g', 'a', 'm', 'e'];
var letterProbs = map(
  function(letter) {
    checkVowel(letter) ? 0.45 : 0.05
  },
  letterVals
)
var letters = Categorical({
  vs: letterVals,
  ps: letterProbs
})

// Compute Pr(L = l | Win)
var distribution = Infer(
  {method: 'enumerate'},
  function() {
    var letter = sample(letters)
    var position = letterVals.indexOf(letter) + 1
    var winProb = 1 / Math.pow(position, 2)
    condition(...)
    return ...
  }
)

viz.table(distribution);
```

Question 3.6.4

Which is higher, Pr(Vowel | Win) or Pr(Consonant | Win)? Answer this using the WebPPL code you wrote *Hint:* use the `checkVowel` function.

```
// Define some variables and utility functions
var checkVowel = function(letter) {
  _.includes(['a', 'e', 'i', 'o', 'u'], letter)
```

```
}
var letterVals = ['g', 'a', 'm', 'e']
var letterProbs = map(
  function(letter) {
    checkVowel(letter) ? 0.45 : 0.05
  },
  letterVals
)
var letters = Categorical({
  vs: letterVals,
  ps: letterProbs
})

// Compute Pr(L = l | win)
var distribution = Infer(
  {method: 'enumerate'},
  function() {
    var letter = sample(letters)
    var position = letterVals.indexOf(letter) + 1
    var winProb = 1 / Math.pow(position, 2)
    condition(...)
    return ...
  }
)

viz.table(distribution)
```

Question 3.6.5

What difference do you see between your code and the mathematical notation? What are the advantages and disadvantages of each? Which do you prefer?