LIGN 165
Final Exam

Please put your answers into a file called `lastname-firstname.clj`. Be careful to follow the instructions exactly and be sure that all of your function definitions use the precise names, number of inputs and input types, and output types as requested in each question. **If you do not follow these instructions, we cannot give you credit for your answer.**

For running your code, you have two options. You can run your code on the course website. Alternatively, you can install Clojure on your local machine and write and debug the answers to each problem in a local copy of `lastname-firstname.clj`. You can find information about installing and using Clojure here `https://clojure.org/`.

You may collaborate with up to three other students on this final. You must submit your work individually. If you do not submit a copy of the problem set under your own name, you will not get credit. When you submit your work, you must indicate who you worked with, and what each of your individual contributions were.

Note that all problems after Problem 5 are optional.

Background: Throughout this problem set we will be working with the Hidden Markov Model. In an HMM, there is a sequence of hidden states $h^{(1)}, ..., h^{(n)}$, and a sequence of observed words, $w^{(1)}, ..., w^{(n)}$. The set of all possible hidden states is S, and the set of all possible words is O. Thus $h^{(i)} \in S$ and $w^{(i)} \in O$ for all i. We defined the distribution on hidden states and words as follows in class:

$$\Pr(W^{(1)}, \ldots, W^{(k)}, H^{(1)}, \cdots, H^{(k)}) = \prod_{i=1}^{k} \Pr(W^{(i)} \mid H^{(i)} = h^{(i)}, \vec{\theta}_{O,h^{(i)}}) \Pr(H^{(i)} \mid H^{(i-1)} = h^{(i-1)}, \vec{\theta}_{S,h^{(i-1)}}) \quad (1)$$

Note that we are assuming (for the time being) that the parameters $\vec{\theta}_{S,h^{(i-1)}}$ and $\vec{\theta}_{O,h^{(i)}}$ are fixed, i.e. there is no uncertainty about their value. $\vec{\theta}_{S,h}$ represents the probability distribution over the next hidden state given that the current hidden state is h. $\vec{\theta}_{O,h}$ represents the probability distribution over the current observation given that the current hidden state is h.

We will assume throughout that there are three hidden states (categories) and three words. These are defined by:

```
(def hidden-states '(Start N V))
```

```
(def vocabulary '(Call me Ishmael))
```

Note that, to make the exercises less painful, we are not including a `stop` state. There are three state transition distributions that we need to define, one for each of the hidden states: $\Pr(H^{(i)} \mid H^{(i-1)} = Start)$, $\Pr(H^{(i)} \mid H^{(i-1)} = N)$, and $\Pr(H^{(i)} \mid H^{(i-1)} = V)$. As stated in the definition of the HMM probabilities above, these distributions are specified by the parameters $\vec{\theta}_{S,Start}$, $\vec{\theta}_{S,N}$, and $\vec{\theta}_{S,V}$:

```
(def theta-transition-Start '(0 0.9 0.1))
```

```
(def theta-transition-N '(0 0.3 0.7))
```

```
(def theta-transition-V '(0 0.8 0.2))
```

```
(def theta-transition-dists-1
  (list theta-transition-Start theta-transition-N theta-transition-V))
```

This means that the hidden state N has probability 0.3 of transitioning to hidden state N, and probability 0.7 of transitioning to V. Note that none of the states ever transition to the start hidden state. This means that the start state can only occur as the first hidden state in a sequence, and never afterwards.

We also need to specify the observation distribution for the hidden states N and V (the hidden state start cannot produce observations, so it is ignored here). They are defined by:

```
(def theta-observation-Start '(0.33 0.33 0.33))
;; note: the Start state never produces any observations, so this is being included only for convenience.

(def theta-observation-N '(0.1 0.5 0.4))

(def theta-observation-V '(0.8 0.1 0.1))

(def theta-observation-dists-1
    (list theta-observation-Start
        theta-observation-N
        theta-observation-V))
```

This means, for example, that the hidden state N has probability 0.4 of emitting the word $Ishmael$. The following helper functions will be useful for writing your code.

```
(defn dist-lookup [state states dists]
  (if (= state (first states))
      (first dists)
      (dist-lookup state (rest states) (rest dists))))

(defn log2 [n]
  (/ (Math/log n) (Math/log 2)))

(defn logsumexp [log-vals]
  (let [mx (apply max log-vals)]
    (+ mx
      (log2
        (apply +
            (map (fn [z] (Math/pow 2 z))
              (map (fn [x] (- x mx)) log-vals)))))))

(defn logscore-categorical [outcome outcomes params]
  (if (= outcome (first outcomes))
    (log2 (first params))
    (logscore-categorical outcome (rest outcomes) (rest params))))
```

Problem 1: Suppose that at time t the hidden state is $h^{(t)}$. Then we can compute the probability that the next hidden state is $h^{(t+1)}$ and the next observed word is $w^{(t+1)}$. This probability is equal to:

$$\Pr(W^{(t+1)} = w^{(t+1)}, H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)})$$
$$= \Pr(W^{(t+1)} = w^{(t+1)} \mid H^{(t+1)} = h^{(t+1)}) \Pr(H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)})$$

Write a procedure score-next-state-word which computes the log of this probability. It should take five arguments: the current hidden state, the next hidden state, and next observed word, the hidden state transition distributions, and the observation distributions. A call to (score-next-state-word current-hidden next-hidden next-observed theta-transition-dists theta-observation-dists) should return the log probability of the next hidden state and next observed word, given the current hidden state.

Hint: use the functions dist-lookup and logscore-categorical defined above.

Problem 2: Suppose that at time t the hidden state is $h^{(t)}$. Then we can compute the *marginal* probability of the word $w^{(t+1)}$ at time $t+1$ given the hidden state at time t. This probability is equal to:

$$\Pr(W^{(t+1)} = w^{(t+1)} \mid H^{(t)} = h^{(t)}) \tag{2}$$

$$= \sum_{h^{(t+1)}} \Pr(W^{(t+1)} = w^{(t+1)}, H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)}) \tag{3}$$

$$= \sum_{h^{(t+1)}} \Pr(W^{(t+1)} = w^{(t+1)} \mid H^{(t+1)} = h^{(t+1)}) \Pr(H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)}) \tag{4}$$

That is, we sum over all of the possible hidden states that could appear at time $t+1$.

Using the procedure `score-next-state-word` that you wrote in Problem 1, write a function `compute-next-observation-marginal`, which takes four arguments: `current-state`, `next-observation`, `theta-transition-dists`, and `theta-observation-dists`. When (`compute-next-observation-marginal current-state next-observation theta-transition-dists theta-observation-dists`) is called, it should return the log of the marginal probability of the next observed word given the current hidden state.

Hint: use `logsumexp`.

Problem 3: Suppose that at time t the hidden state is $h^{(t)}$. Then we can compute the probability of the next k hidden states and next k observed words given the hidden state at time t. This probability is equal to:

$$\Pr(W^{(t+1)}, \ldots, W^{(t+k)}, H^{(t+1)}, \cdots, H^{(t+k)} \mid H^{(t)} = h^{(t)}) \tag{5}$$

$$= \prod_{i=t}^{t+k-1} \Pr(W^{(i+1)} = w^{(i+1)} \mid H^{(i+1)} = h^{(i+1)}) \Pr(H^{(i+1)} = h^{(i+1)} \mid H^{(i)} = h^{(i)}) \tag{6}$$

$$= \Pr(H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)}) \Pr(W^{(t+1)} = w^{(t+1)} \mid H^{(t+1)} = h^{(t+1)}) \tag{7}$$

$$\cdot \Pr(W^{(t+2)}, \ldots, W^{(t+k)}, H^{(t+2)}, \cdots, H^{(t+k)} \mid H^{(t+1)} = h^{(t+1)}) \tag{8}$$

Note that the equation shows that there is a recursive formula for this probability.

Write a procedure `score-next-states-words` which computes this probability. The function should take five arguments: the current hidden state, a list of k hidden states, a list of k observed words, the transition distributions, and the observation distributions. When (`score-next-states-words current-hidden next-hidden-states next-words theta-transition-dists theta-observation-dists`) is called, it should return the log probability that the k hidden states and k observed words will appear immediately following the current hidden state.

You should use recursion to write the function.

Problem 4: Suppose the at time t the hidden state is $h^{(t)}$. Then we can compute the marginal probability of the next k observed words given this hidden state, i.e. summing out all of the possible settings of the hidden states. This probability is equal to:

$$\Pr(W^{(t+1)}, \ldots, W^{(t+k)}) \mid H^{(t)} = h^{(t)}) \tag{9}$$

$$= \sum_{h^{(t+1)}, \ldots h^{(t+k)}} \Pr(W^{(t+1)}, \ldots, W^{(t+k)}, H^{(t+1)}, \cdots, H^{(t+k)} \mid H^{(t)} = h^{(t)}) \tag{10}$$

$$= \sum_{h^{(t+1)}, \ldots h^{(t+k)}} \prod_{i=t}^{t+k} \Pr(W^{(i+1)} = w^{(i+1)} \mid H^{(i+1)} = h^{(i+1)}) \Pr(H^{(i+1)} = h^{(i+1)} \mid H^{(i)} = h^{(i)}) \tag{11}$$

$$= \sum_{h^{(t+1)}} \Pr(H^{(t+1)} = h^{(t+1)} \mid H^{(t)} = h^{(t)}) \Pr(W^{(t+1)} = w^{(t+1)} \mid H^{(t+1)} = h^{(t+1)}) \cdot \Pr(W^{(t+2)}, \ldots, W^{(t+k)}) \mid H^{(t+1)} = h^{(t+1)})$$

$$\tag{12}$$

The final equation is very important. It shows that the marginal probability of the words following hidden state $h^{(t)}$ can be computed in a recursive manner. Write a function `compute-next-words-marginal`, which computes the

3

log marginal probability of a sequence of observed words given the current hidden state. The function should take four arguments: the current hidden state, a list of words, the transition distributions, and the observation distributions. When (`compute-next-words-marginal current-hidden next-words theta-transition-dists theta-observation-dists`) is called, it should return the log of the marginal probability of the list of words given the hidden state, as defined in the equation above.

You should use recursion to write your function.

Problem 5: Find a sequence of two words $w^{(1)}, w^{(2)}$ such that the marginal probability of this sequence is different than the marginal probability of its reverse. That is, find $w^{(1)}, w^{(2)}$ such that:

$$\Pr(W^{(1)} = w^{(1)}, W^{(2)} = w^{(2)} \mid H^{(0)} = Start) \neq \Pr(W^{(1)} = w^{(2)}, W^{(2)} = w^{(1)} \mid H^{(0)} = Start)$$

Use your solution to Problem 4 to confirm that your sequences have different probabilities, and explain why this is true.

Problem 6 (OPTIONAL): In the next several problems, we will be using your solution to Problem 4 to perform Bayesian inference in the HMM. Our goal will be to compute the posterior distribution over hidden states given a sequence of words. Recall that Bayes' Rule tells us how to calculate this posterior distribution:

$$\Pr(H^{(1)} = h^{(1)}, \cdots, H^{(t)} = h^{(t)} \mid W^{(1)} = w^{(1)}, \ldots, W^{(t)} = w^{(t)}) = \tag{13}$$

$$\frac{\Pr(H^{(1)} = h^{(1)}, \cdots, H^{(t)} = h^{(t)}) \Pr(W^{(1)} = w^{(1)}, \ldots, W^{(t)} = w^{(t)} \mid H^{(1)} = h^{(1)}, \cdots, H^{(t)} = h^{(t)})}{\Pr(W^{(1)} = w^{(1)}, \ldots, W^{(t)} = w^{(t)})} \tag{14}$$

Write a procedure `compute-hidden-prior`, which takes two arguments: a list of k hidden states and the hidden state transition distributions. When (`compute-hidden-prior hidden-states theta-transition-dists`) is called, it should return the log prior probability of the hidden state sequence, i.e. the log value of

$$\Pr(H^{(1)} = h^{(1)}, \cdots, H^{(k)} = h^{(k)})$$

.

Problem 7 (OPTIONAL): In this problem, we will continue with the Bayesian calculation from Problem 6. Write a procedure `compute-likelihood-of-words`, which takes three arguments: a list of k hidden states, a list of k words, and the observation distributions. When (`compute-likelihood-of-words hidden-states words theta-observation-dists`) is called, it should return the log probability of the k words being generated from the hidden states, i.e. the log value of $\Pr(W^{(1)} = w^{(1)}, \ldots, W^{(k)} = w^{(k)} \mid H^{(1)} = h^{(1)}, \cdots, H^{(k)} = h^{(k)})$.

Problem 8 (OPTIONAL): We are finally ready to solve our Bayesian inference problem. Write a procedure `compute-hidden-posterior`, which takes four arguments: a list of k hidden states, a list of k words, the hidden state transition distributions, and the observation distributions. When (`compute-hidden-posterior hidden-states words theta-transition-dists theta-observation-dists`) is called, it should return the log posterior probability of the k hidden states given the k words, i.e. it should return the log value of $\Pr(H^{(1)} = h^{(1)}, \cdots, H^{(k)} = h^{(k)} \mid W^{(1)} = w^{(1)}, \ldots, W^{(k)} = w^{(k)})$.
Hint: Use your solutions to Problems 4, 6, and 7.

Problem 9 (OPTIONAL): Suppose that we use the function `compute-next-words-marginal` that you wrote in Problem 4, in order to compute the marginal probability of a sequence of k observed words. How many times is `compute-next-words-marginal` called during the execution of the program? What does this mean about the run-time of the program?
Hint: your formula should involve the number of hidden states and the length k of the sequence of words.

Problem 10 (Extra Credit):   In class, we introduced *memoization* as a method for sharing computation and/or reducing the run-time of programs. Memoization refers to a technique where we intercept calls to a function with some arguments, check if the function has been called with those arguments before and, if not, call the function on those arguments, get the result, store it in a table and return it. On subsequent calls to that function with those arguments, we simply look up the result in the table and avoid recomputing the value. In Clojure, functions can be memoized by a call to `memoize`.

Using `memoize`, define a memoized version of your function `compute-next-words-marginal`. Name this function `compute-next-words-marginal-mem`.

Problem 11 (Extra Credit):   Suppose that we use the memoized function `compute-next-words-marginal-mem` that you wrote in the previous problem, in order to compute the marginal probability of a sequence of k observed words. How many times is `compute-next-words-marginal-mem` called during the execution of the program? How does this differ from the non-memoized version of the procedure?