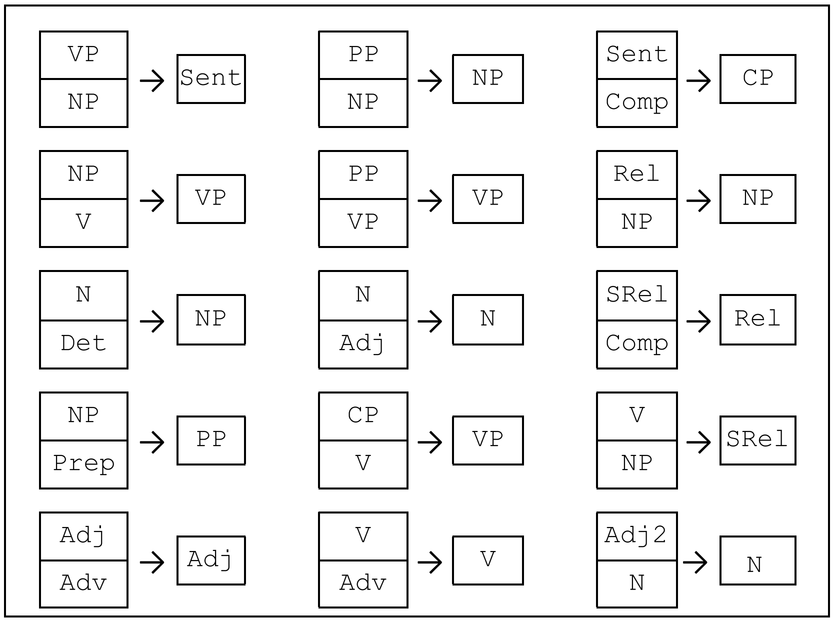
(R) A Make-Shift Code (1/4) [15 points]

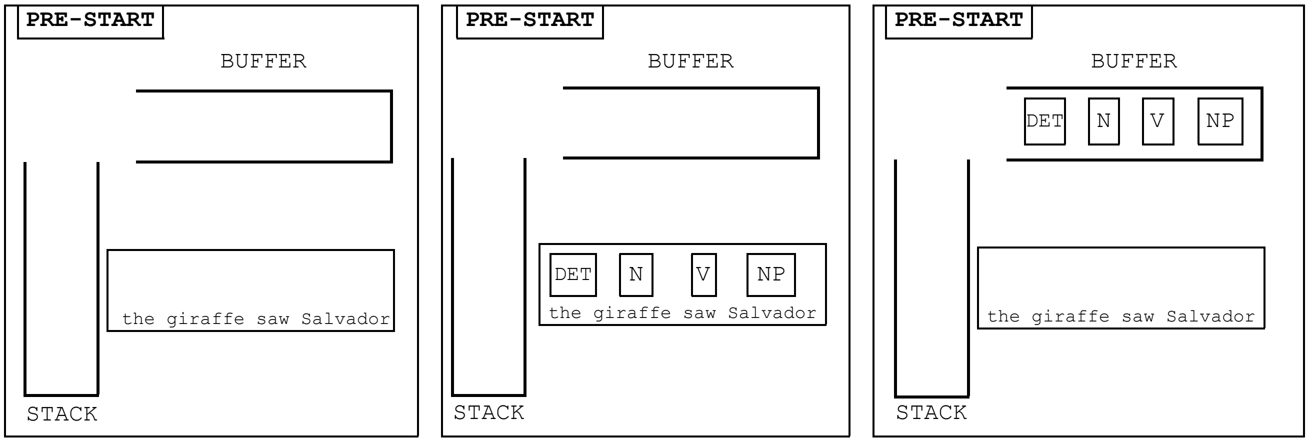
Alice and Bob belong to a secret organization called People Avoiding Really Sinister Eavesdroppers (PARSE). Unfortunately for them, all of their messages are intercepted by an enemy named Eve (who loves to eavesdrop), so Alice and Bob come up with a scheme for encrypting their messages to prevent Eve from reading them.

Their scheme is based on a device called a **shift-reduce parser**. This device makes use of a set of rules (below, left) and a table of part-of-speech tags (below, right):



Tag	Words that can have that tag
NP	Salvador
N	giraffe, glass, saw, orange, felt, spots
V	saw, felt, heard
Comp	that
Det	the
Adj	glass, soft, orange
Adj2	extraordinaire
Adv	really
Prep	on
PP	here

The shift-reduce parser has two segments, called the **stack** and the **buffer**, and it takes a series of words as its input. To run the device, you first input the words, assign a tag from the part-of-speech tag table (above, right) to each word, and then move these tags (in order) onto the buffer. If the machine had the input sequence “the giraffe saw Salvador”, these pre-processing steps would look like this:



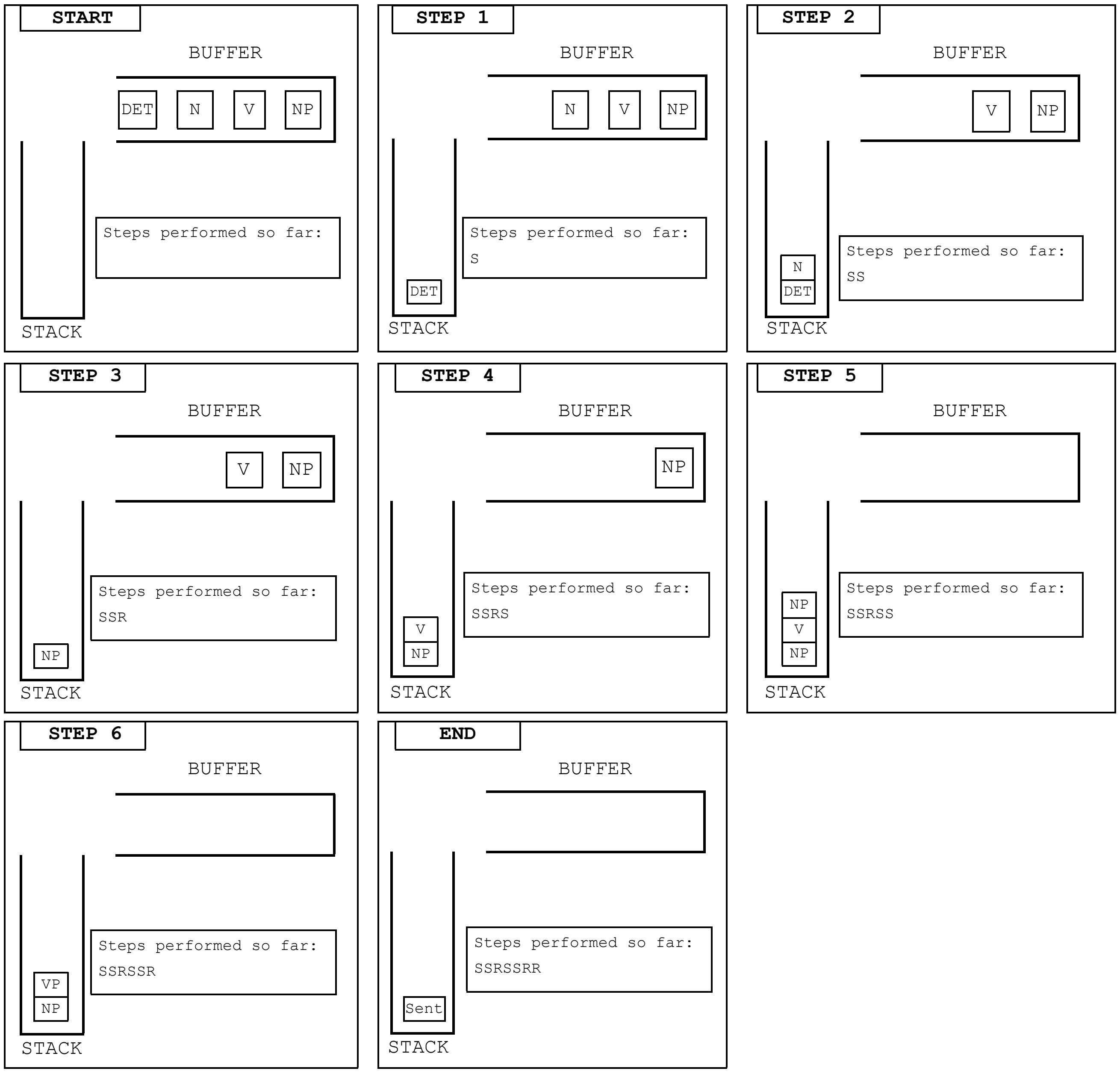
Note that there might be more than one option for a word’s tag–for example, “saw” could be N or V.

(R) A Make-Shift Code (2/4)

Now the main operation of the machine begins. At each step of its processing, the machine performs one operation, which is either a **shift** operation or a **reduce** operation. These two operations are defined as follows:

- **Shift**: Take the first item in the buffer and place it on top of the stack.
- **Reduce**: Combine the top two items in the stack using one of the rules from the rule set.

If the machine reaches a point where all the words have been removed from the buffer, and the only thing remaining on the stack is Sent, then the machine has created a successful parse for its input sentence. For our example, the shift-reduce processing would look as follows (we will use the abbreviations S for Shift and R for Reduce):



(R) A Make-Shift Code (3/4)

Because the parser ends up with nothing in the buffer and with only Sent in the stack, this was a successful parse of the input sentence. Note that the parser was not guaranteed to succeed in this case--if it had shifted at step 3 instead of reducing, it would have failed.

The way that Alice and Bob use this device to send messages is based on the sequence of Shift and Reduce operations that the shift-reduce parser performs (in the diagrams, this sequence is listed under "Steps performed so far"). Specifically, they break the final sequence of steps into chunks of 5 and then decode those chunks of 5 based on the following code:

A = RRRRR	F = RRRSS	K = RRRRS	P = RRSSR	U = SSSSS	Z = RRSRS
B = RRRSR	G = SSSSR	L = SRRSS	Q = SSRSR	V = SRSRS	
C = SSRSS	H = SSSRS	M = RSSSS	R = SSRRS	W = RRSRR	
D = SRSRR	I = SRRRS	N = SRSSR	S = RRSSS	X = RSRRS	
E = SSSRR	J = RSSRR	O = SRRRR	T = SSRRR	Y = SRRSR	

Therefore, the example worked out above would not actually yield a complete message because its final sequence of steps is SSRSSRR, so Alice or Bob could interpret the first 5 characters (SSRSS) as C, but there would still be two characters (RR) left dangling.

With this code, Alice and Bob successfully sent many messages to each other, where each message would name the country where their organization's next meeting would be. Eve continued to intercept these messages, and after years of careful analysis she managed to crack the code.

R1. The first message Eve intercepted after she broke the code was this sentence:

```
Salvador saw the orange spots on the giraffe
```

With mounting excitement, she decoded the message...but was utterly perplexed at the result. *Does this mean that PARSE owns some sort of spaceship?!?!*, she wondered to herself as she threw away the useless message.

a. What did Eve think the message said?

b. Where were Alice and Bob actually meeting?

(R) A Make-Shift Code (4/4)

R2. A few weeks later, Eve intercepted a second message between Alice and Bob:

```
the glass saw that the really soft orange felt really heard the giraffe
```

Once again, she decoded the message, and this time it actually made sense! As soon as she figured it out, she hopped on a plane to the country it named and waited there for Alice and Bob, but they never showed up! At least the trip wasn't a total waste--it was Eve's first visit to Africa.

a. Where had Eve gone?

b. Where were Alice and Bob actually meeting?

R3. It's time for Alice to choose a new place to meet. When she and Bob last met, it took them a long time to find each other because all that Alice knew was the country where Bob wanted to meet, and countries are pretty big. To fix this, Alice has decided to send Bob the name of a city instead of a country. She's made a list of the six cities she most wants to visit: Ottawa, Quito, Oslo, Fez, Irkutsk, and Perth. The map below shows where these cities are located:

It turns out that only one of these six cities is a possible place for Alice and Bob to meet.

a. Which of these six cities is the one where they can meet?

b. Why wouldn't the other five cities work?

c. Write a sentence that Alice could send to Bob to represent the name of the city that would work.

Note: The shift-reduce parsing algorithm is used in the real world to parse both human languages and programming languages. A parsing algorithm (such as this one) that builds up an analysis of a sequence by putting together individual units of the sequence is known as a bottom-up parsing algorithm. The opposite type of strategy, namely starting with the Sent symbol and trying to expand it until the desired sequence is created, is known as top-down parsing.