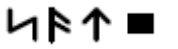
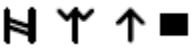
YOUR NAME: REGISTRATION #

(L) Transducing Runes (1/5) [10 points]

Before the Roman alphabet was introduced to Northern Europe, much of Scandinavia and what is now Great Britain used a writing system called Runic. These symbols have recently gained increasing popularity because the fantasy author J.R.R. Tolkien adapted an Anglo-Saxon Runic writing called Futhorc in his series *Lord of the Rings* (and *The Hobbit*).

This problem is about mathematical constructs that we can use to turn Roman text (i.e., what English is written in) into runes. This is not a simple substitution, however, because there is not a one-to-one connection between Roman letters and runes. For example, these words become the following runes. To make things cleaner, we're assuming that every word written in Roman characters is followed by a # to mark the end of the word. You can assume that every input Latin word will be terminated by a #, and that this becomes ■ in runes.

Roman	Runic
sat#	
eat#	
heat#	
east#	

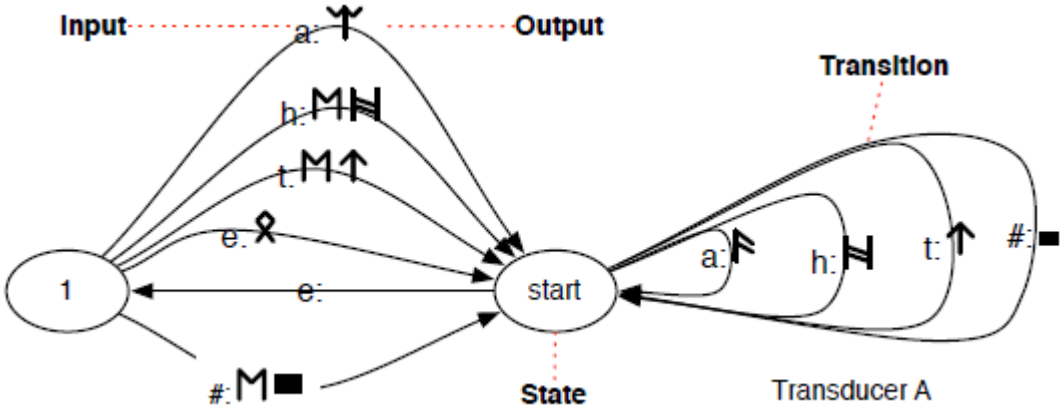
Specifically, there are a number of runes that are equivalent to two Roman characters. To keep things simple, we'll start with a very limited alphabet.

a		ea	
e		ee	
h		th	
s		st	
t		#	■

The tool that we're going to use is called a **transducer**, a logical tool that is used in morphological processing (e.g., to remove suffixes and prefixes from words) in natural language processing technology.

YOUR NAME: REGISTRATION #

(L) Transducing Runes (2/5)



The key components of a transducer are states, transitions, inputs, and outputs. We always start in the "start" state. In the example transducer below, this is the right circle with the label "start" inside it. We transition to different states based on the input that we get.

In this problem, our input is Roman characters. For example, if we're in the "start" state and see either *h*, *a*, or *t*, then we transition from the "start" state to the "start" state (simple!). If, however, we were in the "start" state and saw the character *e*, we would transition to state "1".

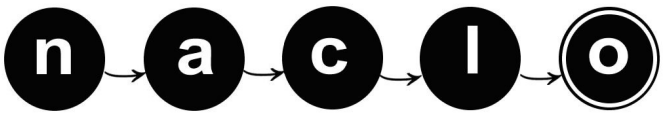
Which transition we use is based on the input we receive. When we transition, we also can output. In the start state,

- if we see *h* we output ᚺ;
- if we see *a* we output ᚨ;
- if we see *t* we output ᛏ;
- if we see # we output ■;
- but if we see *e* we output nothing.

Transitions are depicted with an arrow. Each arrow has a label that shows the input and output. To the left of the colon (:) is the input, and the output is to the right (possibly empty, as in the case of *e* in the start state).

Different states can have different transitions; we output different runes based on input. In state "1"; for example, if we then see *a*, we output ᛉ, which allows us to turn the input of *e* followed by *a* into the correct rune. Thus, if we're in state "1" it means that we might need to turn a sequence of characters into a single rune, but we won't know for sure until we see the next character.

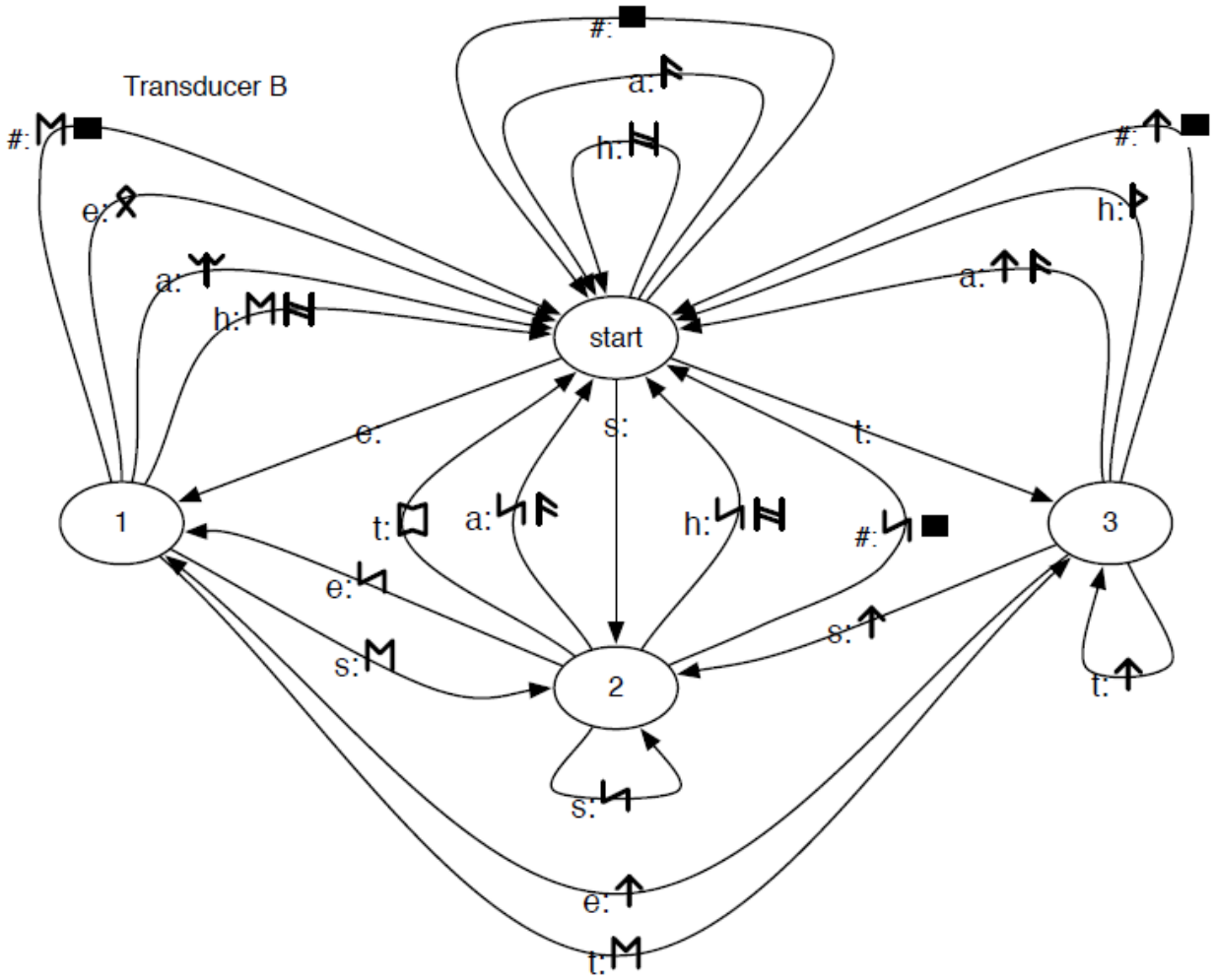
If you're unclear on the concept, trace *eat#* and *heat#* through this simple transducer and make sure you get outputs that match the example runes.



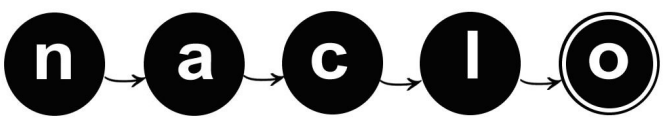
YOUR NAME: REGISTRATION #

(L) Transducing Runes (3/5)

L1. Below is a transducer for the letters *a*, *e*, *h*, *s*, *t*, and #. Given a sequence of Roman characters, give the states that you would visit while transducing those characters. The first is done as an example.



A)	he#	start	start	1	start			
B)	stash#	start						
C)	heath#	start						
D)	thee#	start						



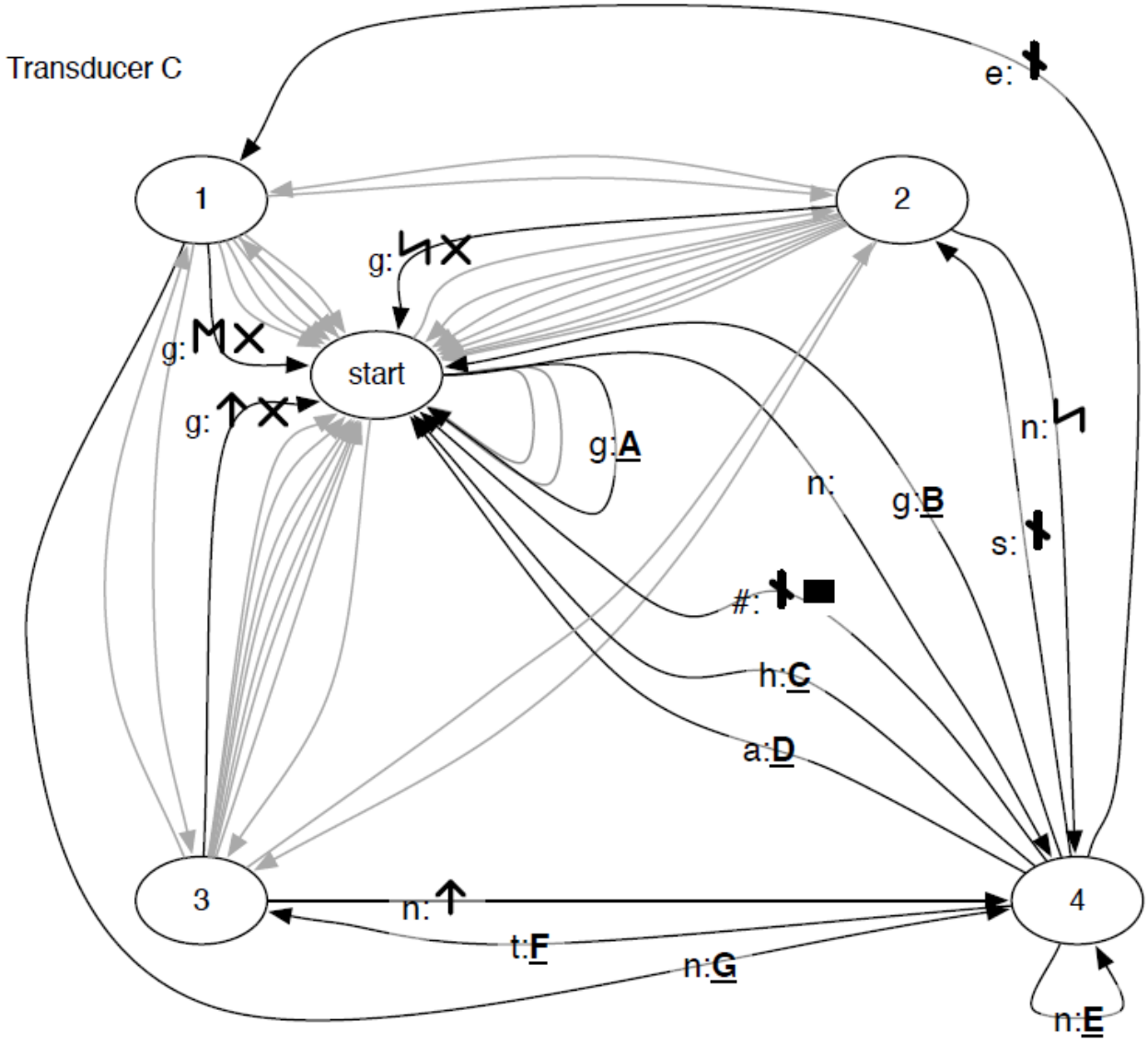
YOUR NAME: REGISTRATION #

(L) Transducing Runes (4/5)

L2. We're going to make our transducer a little more complicated, by adding additional runes. The additional runes we'll add correspond to the letters *n*, *g*, and *ng*.

n g ng

Below is what this transducer looks like. It's getting more complex, so we're not going to show all of it. Instead, we'll show transitions that were in the previous transducer in gray without the inputs and outputs. We also won't give the outputs for some of the transitions; some of the outputs have been replaced by bold, upper-case, underlined Roman letters; you'll fill in those missing runes on the next page.



YOUR NAME: REGISTRATION #

(L) Transducing Runes (5/5)

What is the correct output for the transitions in the above transducer? Use the numbered runes below. CAUTION: Answers can be repeated, outputs may require more than one rune, and *order matters*.

(a) g : A _____

(b) g : B _____

(c) h : C _____

(d) a : D _____

(e) n : E _____

(f) t : G _____

(g) n : H _____

1. 2. 3. 4. 5. 6. 7. 8.

L3. Consider the number of states and transitions in a transducer needed to represent different alphabets. The table has the number of states and transitions for the transducers previously shown (don't forget about the end of the word marked #).

Transducer	Single Runes	Double Runes	States	Transitions
A	(a), (e), (h), (t)	(ea), (ee)	2	10
B	(a), (e), (h), (s), (t)	(ea), (ee), (st), (th)	4	24
C	(a), (e), (h), (g), (n), (s), (t)	(ea), (ee), (ng), (st), (th)	5	?
D	(a), (d), (e), (h), (g), (n), (s), (t)	(ea), (ee), (nd), (ng), (st), (th)	?	?

A)	How many transitions does transducer C have?	
B)	How many states does transducer D have?	
C)	How many transitions does transducer D have?	