Decision problems about regular languages:

What can you write a computer program to tell you about the regular language as input given a regular language as input?

Input: A complete description of a DFA (or an NFA, or a reg. exp, or a right linear grammar)

We can write a program that:

a) Tells you if the language has any blank strings at all.

b) How check if you can get to a final state from an initial state.
b) Tells you if the language is every string on the alphabet.

By checking that every reachable state is final.

c) Tells if the language is finite or infinite.

By checking if DFA has a loop that can reach a final state and is reachable from an init. state.

d) Tells if 2 languages are the same.

$L_1$ and $L_2$ are the same if $L_1 \cap \overline{L_2} = \emptyset$ and $L_2 \cap \overline{L_1} = \emptyset$
So we can "build" the DFA for $L_1 \cap L_2$ and for $L_2 \cap L_1$ (we showed how to do this Friday) and use (a).

e) Tells you if the language contains the string "abba".
   Just run DFA on "abba".

Example of getting a reg exp from an NFA:
We can get rid of more states, but we don't feel like it right now.
Quick overview of first several weeks

Ways of getting from any of these methods for defining a language to any other

right regular grammars

reg exp

NFA

DFA
Kinds of objects:

- letters
- Strings
- languages - sets of strings
- DFAs/NFAs/reg exp/grammars/

From a DFA, we get a language - the set of all the strings it accepts.