Questions from Lesson

1. Consider the DFA from last time, accepting the \textit{abba} language:

\begin{center}
\begin{tikzpicture}[node distance=2cm,initial text=,initial where=left, every state/.style={minimum size=1cm,thick,draw,circle},accepting by arrow, accepting state=4, accepting=blue!50!black]
  \node[state,initial] (0) {$0$};
  \node[state] (1) [right of=0] {$1$};
  \node[state] (2) [right of=1] {$2$};
  \node[state,accepting] (3) [right of=2] {$3$};
  \node[state] (4) [below of=1] {$4$};
  \path[->] (0) edge node {$a$} (1);
  \path[->] (1) edge node {$b$} (2);
  \path[->] (2) edge node {$b$} (3);
  \path[->] (3) edge node {$b$} (4);
  \path[->] (4) edge[loop below] node {$a,b$} (4);
  \path[->] (0) edge[loop left] node {$a$} (0);
  \path[->] (1) edge node {$a$} (4);
  \path[->] (2) edge node {$a$} (4);
\end{tikzpicture}
\end{center}

Write down a DFA accepting the \textit{complement} of this language (the strings NOT accepted by the original DFA).
2. Here is a DFA accepting the strings that start and end with $b$:

![DFA Diagram]

Here is a DFA accepting the strings containing $baa$ inside them:

![DFA Diagram]
(a) Can you build a DFA accepting the *intersection* of these two languages (the strings accepted by *both* DFAs)?
(b) Can you build a DFA accepting the union of these two languages (the strings accepted by either DFA)?
3. Is every language a regular language? If not, can you provide an example of a language that is not regular?

Extra Questions

4. The language of legal bracketings is a collection of strings using the letters $a$ (left bracket) and $b$ (right bracket) following the rule that every $b$ in the string must have a matching $a$ coming before it.

Which of the following strings belong to this language?

(a) $a$
(b) $ba$
(c) $abab$
(d) $abba$
(e) $aabb$
(f) $ababb$
(g) $abaababb$

Is the language of legal bracketings regular? Why or why not?
5. The steps we talked about for building a DFA accepting the union or intersection of two regular languages always work, but sometimes it creates more states than we really need. When we built a DFA accepting the strings that start and end with $b$, and contain $baa$ somewhere inside, the resulting DFA had nine states.

Can you write down a DFA accepting the same language, but with only six states?

6. Suppose we have two regular languages, which we’ll call $L_1$ and $L_2$. Consider the language of all strings belonging to $L_1$ but not $L_2$. Is this language always regular?

If so, describe a process for building a DFA accepting this language, given DFAs accepting $L_1$ and $L_2$. If not, explain why not.