1. (8 points) Run the CKY algorithm on the grammar $G$

$$
S \rightarrow AB | a \\
A \rightarrow a \\
B \rightarrow AB | SA | b
$$

and the input string $z = aaba$. Show the final values of all the sets $T_{i,j}$. If $z$ belongs to $L(G)$, then show a derivation tree.

2. (8 points) (Problem 4 in Homework 9 on pg. 310 of our text) Prove that an r.e. set is recursive if and only if there is an enumeration machine that enumerates it in increasing order. (Clarification: Increasing order means that if $k<l$, then all strings of length $k$ are enumerated before all strings of length $l$, and all strings of the same length are enumerated in lexicographic order.)

3. (6 points) Let $\Lambda$ denote the set of all DFAs which accept an infinite language. That is, $\Lambda = \{ M \mid M \text{ is a DFA and } |L(M)| = \infty \}$. Is $\Lambda$ a recursive set? That is, is it decidable whether the language accepted by an arbitrary DFA is finite or infinite? If $\Lambda$ is recursive, then you don't need to give an explicit program. You only need to describe an algorithm to decide on membership in $\Lambda$. 
1. A derivation tree is:

```
   S
  /  \
 A  B
 /  \  |
S  A  a
/ \  |
A  B  a
|   |
a  b
```

2. Assume there is an enumeration machine $M$ that lists $L$ in increasing order. Then on input $z$, total TM $M'$ just runs $M$ and waits until either
   - it finds that $M$ lists $z$, in which case $M'$ accepts $z$ by entering state $t$, or
   - $M$ lists a string which is later in the order than $z$, in which case $M'$ rejects $z$ by entering state $r$.

One of the cases must happen after a finite number of steps, so $M'$ is total. Assume there is a total Turing machine $M$ that accepts $L$. We design $M'$ to
   - list $\Sigma^*$ in order $y_0, y_1, \ldots$.
   - for each $y_i$
     - if $M'$ goes to $t$ then print $y_i$ on output tape

Since $M$ is total, every $y_i$ will be processed in a finite amount of time.

Assume there is a TM that enumerates $L$ in increasing order. For any $z$, another TM $M$ could witness the listing until either $z$ is listed (in which case $z \in L$) or a $y$ is listed and $|y| > |z|$ (in which case $z \notin L$). Since $M$ is total, $L$ is recursive.

3. $\Lambda$ is recursive. $L(M)$ is infinite if and only if there is a loop accessible from a state on a path from the initial state of $M$ to a final state of $M$. The algorithm to decide on membership in $\Lambda$ is:
   - remove every state for which there is no path from the initial state of $M$
   - remove every state for which there is no path to a final state of $M$

$M \in \Lambda$ if and only if $M$ contains a loop
Another way to solve the problem is to compute a regular expression describing $L(M)$. $L(M)$ is infinite if and only if the regular expression contains a $\ast$. 