Describe a non-deterministic bottom-up parser for the following grammar, where S is the start state:

1. $S \to AS$
2. $S \to b$
3. $A \to a$

Hint: You describe your parser by giving names to the states and the transitions. You must describe the initial stack. (Either empty or containing a bottom-of-stack symbol, or perhaps something else.) For each transition, you describe what the state is before the transition, what is popped off the stack, what is read from the input file (never more than one symbol), what is pushed onto the stack, what the new state is, and what the output is. The output of a step is always either empty or the name of one of the three grammar rules. For example, if the input string is *aaab*, the output string is 3332111.

I will be very flexible in grading this. That means that you can pop any number of symbols and push any number of symbols in one step. However, never read more than one symbol in a step. (You can read zero symbols in a step.)

This description can either be in the form of a table or a diagram. I recommend a table in this case.

You can do it with only about eight transition rules, perhaps fewer. You can do it with just one state, although some design philosophies call for more states.

You may assume (if you wish) that the input file ends with an "end-of-file" symbol, for which I want you to use the symbol '$'. But, since the parser is non-deterministic, the end-of-file symbol is *not* necessary.

You must tell me how your parser accepts the input string. There are three choices:

1. By empty stack.
2. By final state.
3. By empty stack and final state.

Pick ONE and tell me which one.