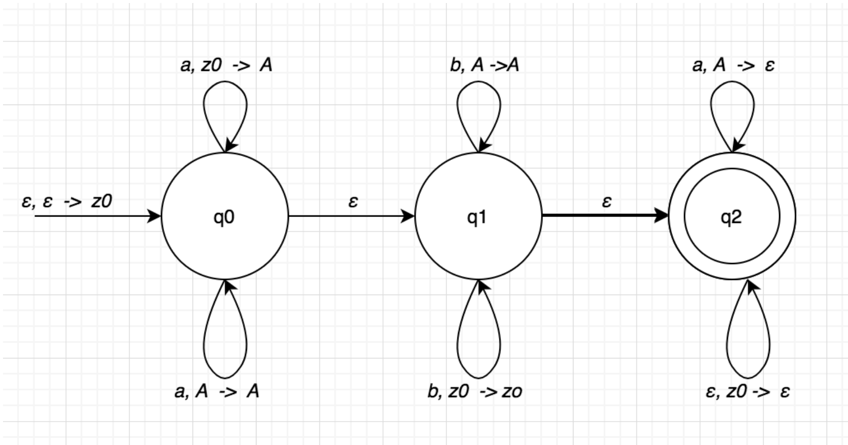
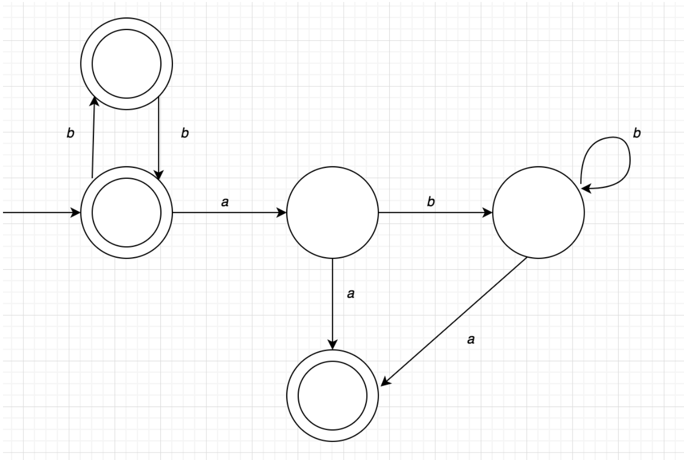
Question V: PDAs

1. Describe a pushdown automaton that recognises $\{a^m b^n a^m | m,n \geq 0\}$.



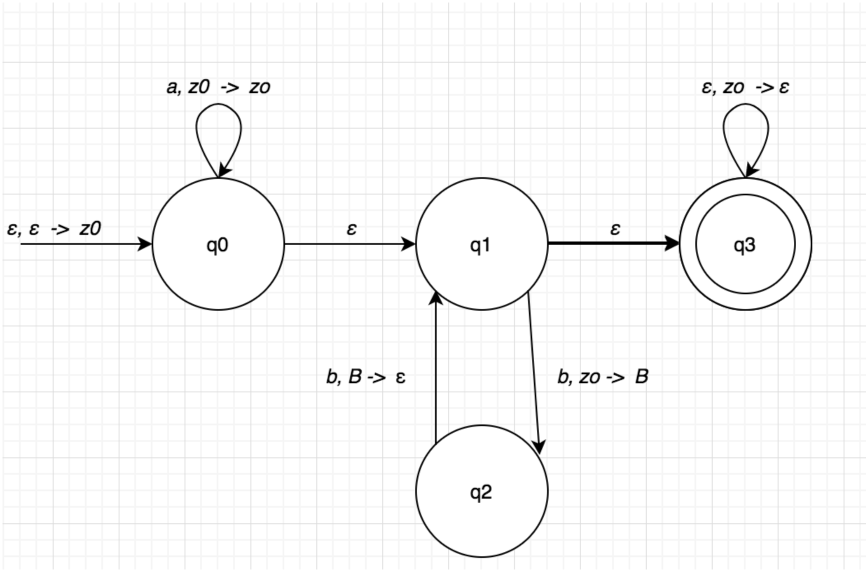
Here the PDA starts with pushing special character "z0" to the stack.
Following the PDA must keep a memory of how many a's it has seen so in state "q0" if an input of "a" is seen the PDA will push an "A" onto the stack. If the top of the stack is "z0" push an "A". If an "A" already exists on the stack then another "A" is pushed.
The PDA can at any point take the Epsilon movement to state "q1" where any occurrence of "b" can be seen. Although ambiguously written here if the input is "b" and the stack top is either a "A" or "z0" no change is made to the stack.
Again at some stage an Epsilon movement can be made to final state "q2" where it will pop "A"s from the stack until it sees a "z0" as the stack top where it will accept it is an even number on both sides.

2. Assume m,n≥0m,n≥0. Describe a DFA that recognises {ambnam}.
Explain why this question is different from the previous question.



Although you can draw DFA's for palindromes you would have to have an infinite amount of states to accept all occurrences of a's and then match them after 0 to the infinity occurrences of "b".
Thus unless you have a machine with infinite memory you could not construct a DFA to accept all possibilities. However a PDA can keep a memory of how many times "a" has occurred so it can match the a's from the left hand side on the right by popping the a's added to the stack on the left occurrence until it sees a special character which indicates the end of the occurrences of "a" are equivalent.

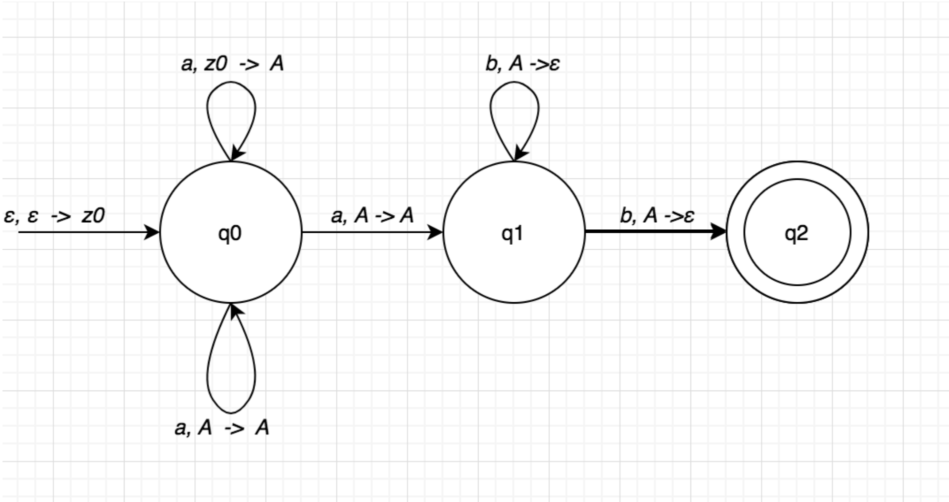
3. Describe a pushdown automaton that recognises {amb2n|m,n≥0}.



Here the language to recognise is any amount of a's with an occurrence of b's that is two times n. First a special character "z0" is inserted into the stack beginning state "q0". From here any occurrence of "a" has no change to the stack value. From here an epsilon move can be used to state "q1". Here there must always be an even occurrence of "b" so therefore if b2n where n is 1 you would have to have two occurrences of "b" in the input to make a valid input. Therefore the PDA always satisfies the language.

*In stage "q1" can take epsilon to final state "q3" where it will pop the "z0" from the stack. This PDA can also keep the input empty string as 2 * 0 = 0 satisfying any occurrence of "a" and no "b".*

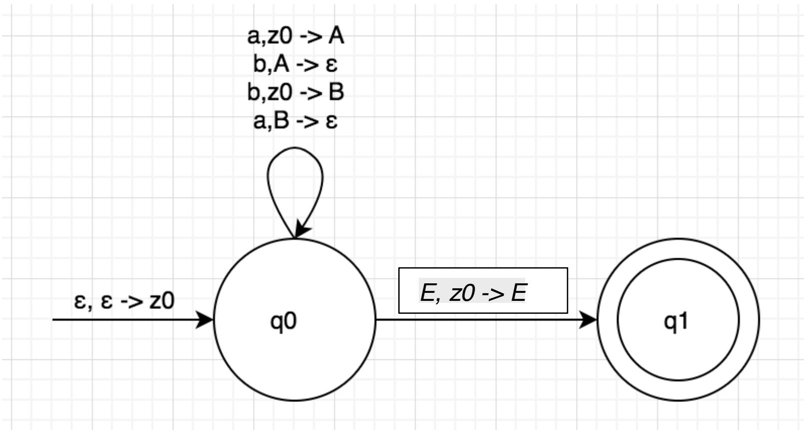
4. Describe a pushdown automaton that recognises $\{a^m b^n | m>n>0\}$.



Here the PDA takes in an initial value onto the stack "z0" moving into state "q0". Here in this state you can either push an "A" on top of "z0" or push another "A" onto another "A".

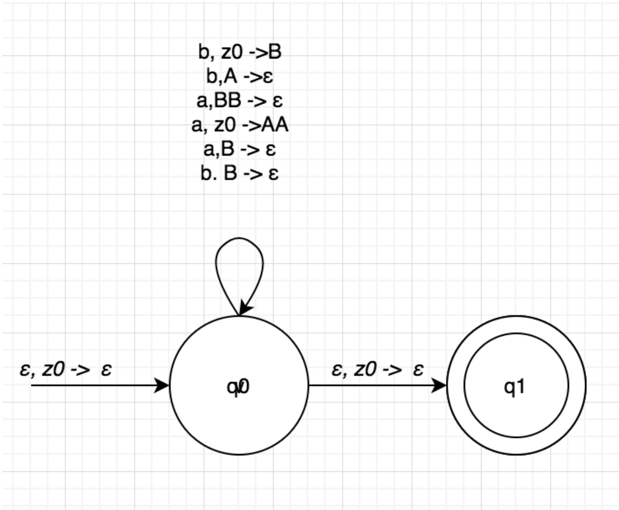
Transiting from state "q0" to "q1" you push another "A" onto the stack and here whenever you any occurrence of "b" you pop an "A" from the stack. To transition into the final state an input of "b" will pop an "A" that is in the stack. Here the final states accepts by acceptance as there will always be a value of "A" on the stack to show more occurrences of "a".

5. Describe a pushdown automaton that recognises $\{w \mid \#_a w = \#_b w\}$, where $\#_a w$ is the number of a's appearing in w and $\#_b w$ is the number of b's appearing in w.



Here we transition into state "q0" and push "z0" onto the stack. Once in this state there are four transitions from "q0" back to "q0". Here we can always keep an even occurrence of "a" and "b". Where if you input an "a" where "z0" is the top of the stack you push an "A" and similarly when you input a b with "z0" on the stack top then you push a "B".
It is implied in this PDA that if you see an "a" or "b" where there is already a value other than "z0" on the stack then there is no stack manipulation.
The only way to get to the final state "q1" there has to be an empty stack meaning equal "a" and "b" inputs.

6. Describe a pushdown automaton that recognises $\{w \mid \#_a w = 2\#_b w\}$.



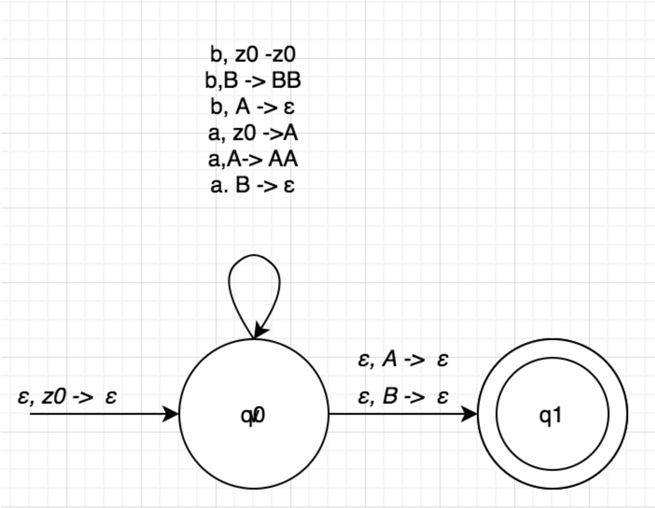
Here "z0" is pushed onto the stack going to state "q0". From here inputting an "a" first will push "AA" onto the stack where inputting a "b" will result in "B" being pushed to the stack.

From here the transition rules of the PDA will always make sure for every occurrence of "a" will subsequently need two "b"s to be generated.

Finally the stack has to be empty to move into final state "q1".

This also satisfies the equation the 0 occurrences of "a" will result in 0 occurrences of "b".

7. Describe a pushdown automaton that recognises $\{w | \#_a w \neq \#_b w\}$.



Here "z0" is pushed onto the stack going to state "q0". From here inputting an "a" first will push "A" onto the stack where inputting a "b" will result in "B" being pushed to the stack.

Here if you insert an "a" and see "A" on the stack you will push "AA" onto the stack and if a "B" is on top you will pop the stack top.

This also stands for if you input an "b".

To move to the final state there has to be either an "A" or "B" on the stack to bring it into the final "accepting" state "q1".