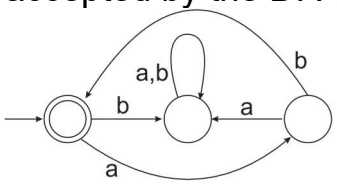
Part III: DFAs and NFAs

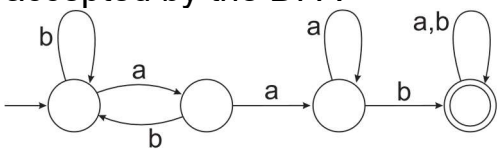
You may find draw.io useful for drawing automata. If you know of a similar and better tool, please let me know. You can insert images from https://automatonsimulator.com too as long as the edge annotations are clearly shown (note sometimes they are difficult to read).

1. By writing a regular expression or a grammar, describe the language accepted by the DFA



(ab)*

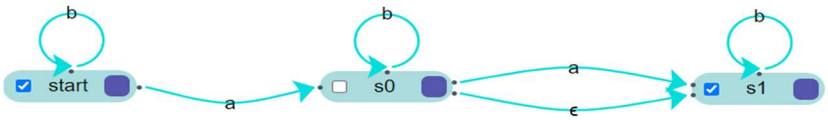
2. By writing a regular expression or a grammar, describe the language accepted by the DFA



(b|ab)*aaa* b[a,b]*

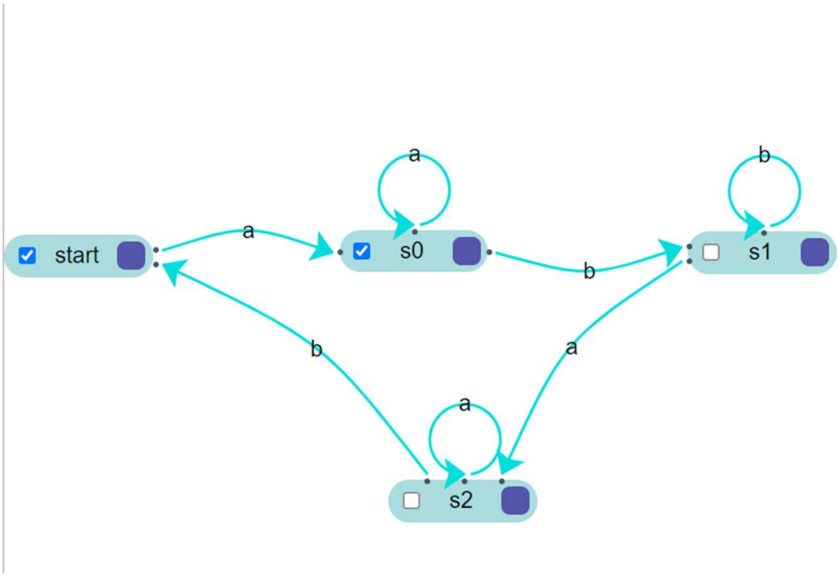
3. Construct NFAs (possibly with ϵ-moves) to recognise the languages on alphabet {a,b} such that:
   1. $L=\{w \in \{a,b\}^* \mid$ w contains at most two a's$\}$.
      So ϵ and aa and bbbbabba are in L and aaa is not.

{"type":"NFA","nfa":{"transitions":{"start":{"a":["s0"],"b":["start"]},"s0":{"a":["s1"],"":["s1"],"b":["s0"]},"s1":{"b":["s1"]}},"startState":"start","acceptStates":["start","s1"]},"states":{"start":{"isAccept":true},"s0":{"top":246,"left":249,"displayId":"s0"},"s1":{"isAccept":true,"top":248,"left":542,"displayId":"s1"}},"transitions":[{"stateA":"start","label":"a","stateB":"s0"},{"stateA":"start","label":"b","stateB":"start"},{"stateA":"s0","label":"a","stateB":"s1"},{"stateA":"s0","label":"ϵ","stateB":"s1"},{"stateA":"s0","label":"b","stateB":"s0"},{"stateA":"s1","label":"b","stateB":"s1"}],"bulkTests":{"accept":"\naa\nbbbbabba\nbabab","reject":"aaa\nbaaab"}}



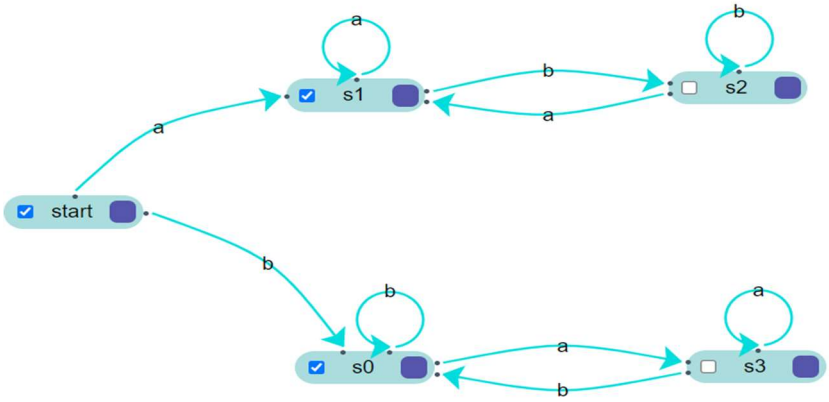
2. L={w∈{a,b}$_*$ | w contains an even number of occurrences of ab as a subword }. So ϵ and a and b and abab and abaaba and baaabab are in L, but ab and ababab and abaabab are not.

{"type":"NFA","nfa":{"transitions":{"start":{"a":["s0"]},"s0":{"b":["s1"],"a":["s0"]},"s1":{"b":["s1"],"a":["s2"]},"s2":{"b":["start"],"a":["s2"]}},"startState":"start","acceptStates":["start","s0"]},"states":{"start":{"isAccept":true},"s0":{"isAccept":true,"top":242,"left":237,"displayId":"s0"},"s1":{"top":243,"left":484,"displayId":"s1"},"s2":{"top":419,"left":272,"displayId":"s2"}},"transitions":[{"stateA":"start","label":"a","stateB":"s0"},{"stateA":"s0","label":"b","stateB":"s1"},{"stateA":"s0","label":"a","stateB":"s0"},{"stateA":"s1","label":"b","stateB":"s1"},{"stateA":"s1","label":"a","stateB":"s2"},{"stateA":"s2","label":"b","stateB":"start"},{"stateA":"s2","label":"a","stateB":"s2"}],"bulkTests":{"accept":"abab\n\nababababab\na","reject":"abaabab\nabababab"}}

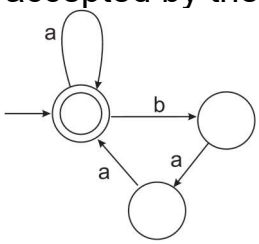


3. L={w∈{a,b}* | the first and the last letter of w are identical }

{"type":"NFA","nfa":{"transitions":{"start":{"a":["s1"],"b":["s0"]},"s1":{"a":["s1"],"b":["s2"]},"s2":{"a":["s1"],"b":["s2"]},"s3":{"a":["s3"],"b":["s0"]},"s0":{"a":["s3"],"b":["s0"]}},"startState":"start","acceptStates":["start","s1","s0"],"states":{"start":{"isAccept":true},"s1":{"isAccept":true,"top":140,"left":213,"displayId":"s1"},"s0":{"isAccept":true,"top":386,"left":220,"displayId":"s0"},"s2":{"top":133,"left":502,"displayId":"s2"},"s3":{"top":385,"left":516,"displayId":"s3"}},"transitions":[{"stateA":"start","label":"a","stateB":"s1"},{"stateA":"start","label":"b","stateB":"s0"},{"stateA":"s1","label":"a","stateB":"s1"},{"stateA":"s1","label":"b","stateB":"s2"},{"stateA":"s2","label":"a","stateB":"s1"},{"stateA":"s2","label":"b","stateB":"s2"},{"stateA":"s3","label":"a","stateB":"s3"},{"stateA":"s3","label":"b","stateB":"s0"},{"stateA":"s0","label":"a","stateB":"s3"},{"stateA":"s0","label":"b","stateB":"s0"}],"bulkTests":{"accept":"","reject":""}}

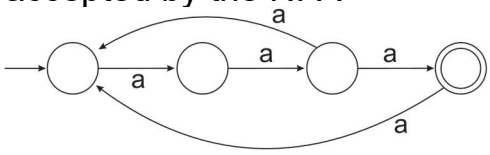


4. By writing a regular expression or a grammar, describe the language accepted by the NFA



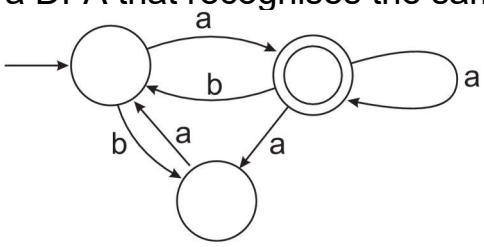
(a|baa)*

5. By writing a regular expression or a grammar, describe the language accepted by the NFA

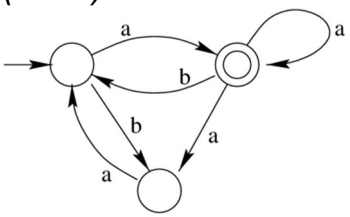


(aaa|aaaa)*aaa

6. *(Unmarked)* Use the powerset construction (see Lecture 5) to construct a DFA that recognises the same language as the following NFA:



7. *(Hard)* Let L be the language recognised by the NFA



   1. Construct a context-free grammar G that generates language L (a regular grammar is also fine, since every regular grammar is context-free).
   2. State with proof whether G is ambiguous or unambiguous.

8. *(Unmarked)* Let $G=(\{S,A,B\},\{a,b\},P,S)$ be the context-free grammar with productions:

   $S \longrightarrow aAbS \mid bBaS \mid \varepsilon$
   $A \longrightarrow aAbA \mid \varepsilon$
   $B \longrightarrow bBaB \mid \varepsilon$

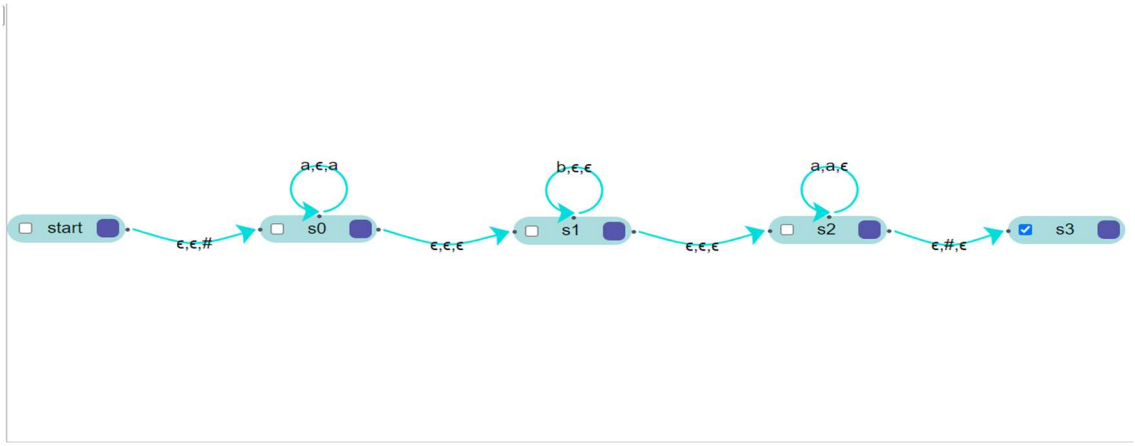
   Give a short and intuitive English description of the language determined by G (such a description does exist).

On PDAs

In the questions below, assume the alphabet is $\{a,b\}$. Your answer must state the acceptance mode used. In the questions below, the word *describe* means *draw a precise picture of* in the style I used in lectures or in the style of this webpage or this pdf, or just search the Internet for how to draw a PDA.

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

{"type":"PDA","pda":{"transitions":{"start":{"":{"":[{"state":"s0","stackPushChar":"#"}]}},"s0":{"a":{"":[{"state":"s0","stackPushChar":"a"}]},"":{"":[{"state":"s1","stackPushChar":""}]}},"s1":{"b":{"":[{"state":"s1","stackPushChar":""}]},"":{"":[{"state":"s2","stackPushChar":""}]}},"s2":{"a":{"a":[{"state":"s2","stackPushChar":""}]},"":{"#":[{"state":"s3","stackPushChar":""}]}}},"startState":"start","acceptStates":["s3"]},"states":{"start":{},"s0":{"top":246,"left":225,"displayId":"s0"},"s1":{"top":248,"left":452,"displayId":"s1"},"s2":{"top":247,"left":680,"displayId":"s2"},"s3":{"isAccept":true,"top":247,"left":893,"displayId":"s3"}},"transitions":[{"stateA":"start","label":"ϵ,ϵ,#","stateB":"s0"},{"stateA":"s0","label":"a,ϵ,a","stateB":"s0"},{"stateA":"s0","label":"ϵ,ϵ,ϵ","stateB":"s1"},{"stateA":"s1","label":"b,ϵ,ϵ","stateB":"s1"},{"stateA":"s1","label":"ϵ,ϵ,ϵ","stateB":"s2"},{"stateA":"s2","label":"a,a,ϵ","stateB":"s2"},{"stateA":"s2","label":"ϵ,#,ϵ","stateB":"s3"}],"bulkTests":{"accept":"aabaa\naaaaabbbbaaaaa\naba","reject":"aabbbaaaaa\nabaaaaa\nabbbbbaa"}}

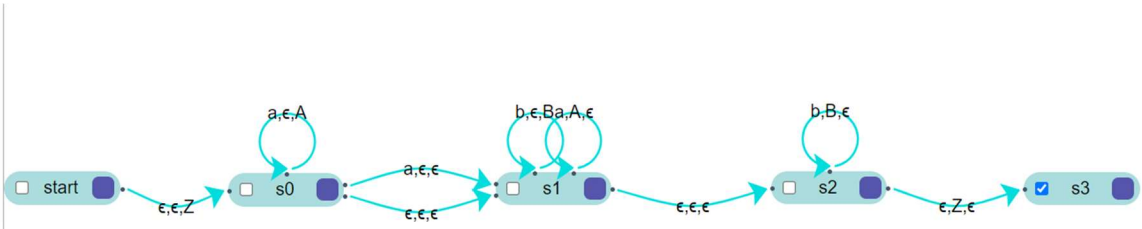


10. Assume $m,n \geq 0$. Describe a DFA that recognises $\{ a^m b^n a^m \}$. Explain why this question is different from the previous question.

There are no DFAs that recognise the given language. The reason is that the language given is not regular.

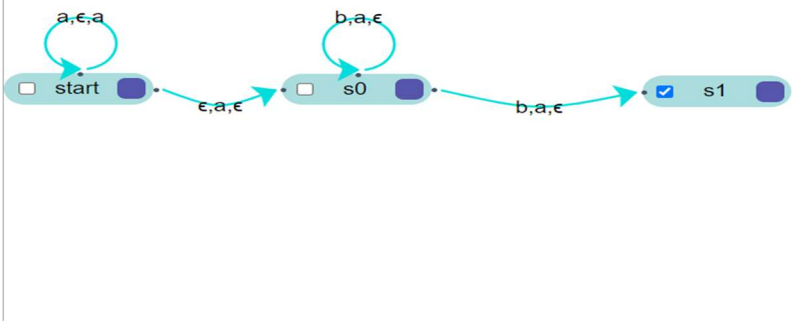
11. Describe a pushdown automaton that recognises $\{a^m b^{2n} | m,n \geq 0\}$.

{"type":"PDA","pda":{"transitions":{"start":{"":{"":[{"state":"s0","stackPushChar":"Z"}]}},"s0":{"a":{"":[{"state":"s0","stackPushChar":"A"},{"state":"s1","stackPushChar":""}]},"":{"":[{"state":"s1","stackPushChar":""}]}},"s1":{"b":{"":[{"state":"s1","stackPushChar":"B"}]},"a":{"":[],"A":[{"state":"s1","stackPushChar":""}]},"":{"":[{"state":"s2","stackPushChar":""}]}},"s2":{"":{"Z":[{"state":"s3","stackPushChar":""}]},"b":{"B":[{"state":"s2","stackPushChar":""}]}}},"startState":"start","acceptStates":["s3"]},"states":{"start":{},"s0":{"top":247,"left":203,"displayId":"s0"},"s1":{"top":246,"left":445,"displayId":"s1"},"s2":{"top":245,"left":695,"displayId":"s2"},"s3":{"isAccept":true,"top":246,"left":924,"displayId":"s3"}},"transitions":[{"stateA":"start","label":"ϵ,ϵ,Z","stateB":"s0"},{"stateA":"s0","label":"a,ϵ,A","stateB":"s0"},{"stateA":"s0","label":"a,ϵ,ϵ","stateB":"s1"},{"stateA":"s0","label":"ϵ,ϵ,ϵ","stateB":"s1"},{"stateA":"s1","label":"b,ϵ,B","stateB":"s1"},{"stateA":"s1","label":"a,A,ϵ","stateB":"s1"},{"stateA":"s1","label":"ϵ,ϵ,ϵ","stateB":"s2"},{"stateA":"s2","label":"ϵ,Z,ϵ","stateB":"s3"},{"stateA":"s2","label":"b,B,ϵ","stateB":"s2"}],"bulkTests":{"accept":"abb\naabb\naaa\naaaa","reject":"bbbbb\nbbbbbbb\nabbb"}}



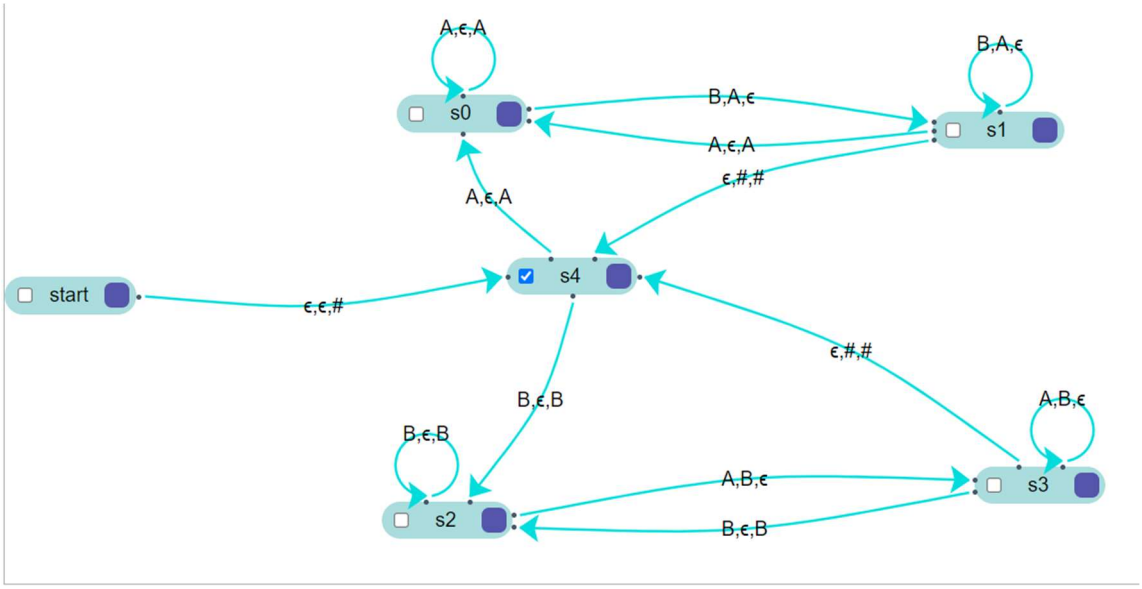
12. Describe a pushdown automaton that recognises $\{a^m b^n | m > n > 0\}$.

{"type":"PDA","pda":{"transitions":{"start":{"a":{"":[{"state":"start","stackPushChar":"a"}]},"":{"a":[{"state":"s0","stackPushChar":""}]}},"s0":{"b":{"a":[{"state":"s0","stackPushChar":""},{"state":"s1","stackPushChar":""}]}}},"startState":"start","acceptStates":["s1"]},"states":{"start":{},"s0":{"top":246,"left":196,"displayId":"s0"},"s1":{"isAccept":true,"top":248,"left":450,"displayId":"s1"}},"transitions":[{"stateA":"start","label":"a,ϵ,a","stateB":"start"},{"stateA":"start","label":"ϵ,a,ϵ","stateB":"s0"},{"stateA":"s0","label":"b,a,ϵ","stateB":"s0"},{"stateA":"s0","label":"b,a,ϵ","stateB":"s1"}],"bulkTests":{"accept":"aab\naaaaaaab","reject":"\nabba\naabbbbb"}}



13. Describe a pushdown automaton that recognises $\{w | \#_a w = \#_b w\}$, where $\#_a w$ is the number of as appearing in w and $\#_b w$ is the number of bs appearing in w.

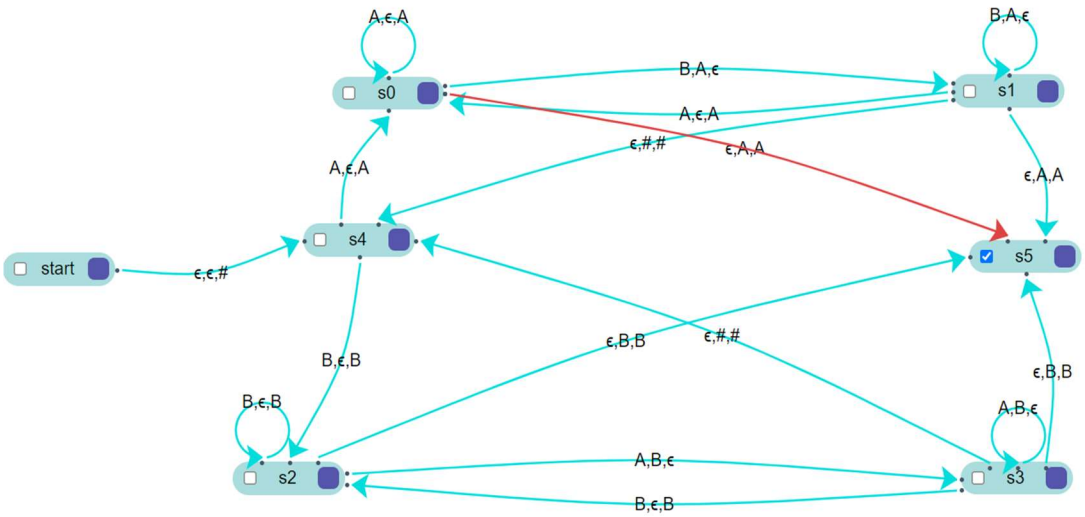
{"type":"PDA","pda":{"transitions":{"start":{"A":{"":[]},"B":{"":[]},"":{"":[{"state":"s4","stackPushChar":"#"}]}},"s0":{"A":{"":[{"state":"s0","stackPushChar":"A"}]},"B":{"A":[{"state":"s1","stackPushChar":""}]}},"s1":{"B":{"A":[{"state":"s1","stackPushChar":""}],"#":[]},"A":{"":[{"state":"s0","stackPushChar":"A"}]},"":{"#":[{"state":"s4","stackPushChar":"#"}]}},"s2":{"B":{"":[{"state":"s2","stackPushChar":"B"}]},"A":{"B":[{"state":"s3","stackPushChar":""}]}},"s3":{"B":{"":[{"state":"s2","stackPushChar":"B"}]},"A":{"B":[{"state":"s3","stackPushChar":""}],"#":[]},"":{"#":[{"state":"s4","stackPushChar":"#"}]}},"s4":{"A":{"":[{"state":"s0","stackPushChar":"A"}]},"B":{"":[{"state":"s2","stackPushChar":"B"}]}}},"startState":"start","acceptStates":["s4"]},"states":{"start":{},"s4":{"isAccept":true,"top":230,"left":402,"displayId":"s4"},"s0":{"top":100,"left":314,"displayId":"s0"},"s1":{"top":113,"left":744,"displayId":"s1"},"s2":{"top":425,"left":302,"displayId":"s2"},"s3":{"top":397,"left":777,"displayId":"s3"}},"transitions":[{"stateA":"start","label":"ϵ,ϵ,#","stateB":"s4"},{"stateA":"s0","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s0","label":"B,A,ϵ","stateB":"s1"},{"stateA":"s1","label":"B,A,ϵ","stateB":"s1"},{"stateA":"s1","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s1","label":"ϵ,#,#","stateB":"s4"},{"stateA":"s2","label":"B,ϵ,B","stateB":"s2"},{"stateA":"s2","label":"A,B,ϵ","stateB":"s3"},{"stateA":"s3","label":"B,ϵ,B","stateB":"s2"},{"stateA":"s3","label":"A,B,ϵ","stateB":"s3"},{"stateA":"s3","label":"ϵ,#,#","stateB":"s4"},{"stateA":"s4","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s4","label":"B,ϵ,B","stateB":"s2"}],"bulkTests":{"accept":"AB\nABAB\nABABAB\nBA\nAAABBB\nAABBBBBAAA","reject":"A\nB\nABA\nBB\nABABB"}}



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

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

{"type":"PDA","pda":{"transitions":{"start":{"A":{"":[]},"B":{"":[]},"":{"":[{"state":"s4","stackPushChar":"#"}]}},"s0":{"A":{"":[{"state":"s0","stackPushChar":"A"}]},"B":{"A":[{"state":"s1","stackPushChar":""}]},"":{"A":[{"state":"s5","stackPushChar":"A"}]}},"s1":{"B":{"A":[{"state":"s1","stackPushChar":""}],"#":[]},"A":{"":[{"state":"s0","stackPushChar":"A"}]},"":{"#":[{"state":"s4","stackPushChar":"#"}],"A":[{"state":"s5","stackPushChar":"A"}]}},"s2":{"B":{"":[{"state":"s2","stackPushChar":"B"}]},"A":{"B":[{"state":"s3","stackPushChar":""}]},"":{"B":[{"state":"s5","stackPushChar":"B"}]}},"s3":{"B":{"":[{"state":"s2","stackPushChar":"B"}]},"A":{"B":[{"state":"s3","stackPushChar":""}],"#":[]},"":{"#":[{"state":"s4","stackPushChar":"#"}],"B":[{"state":"s5","stackPushChar":"B"}]}},"s4":{"A":{"":[{"state":"s0","stackPushChar":"A"}]},"B":{"":[{"state":"s2","stackPushChar":"B"}]}}},"startState":"start","acceptStates":["s5"]},"states":{"start":{},"s4":{"top":219,"left":282,"displayId":"s4"},"s0":{"top":86,"left":309,"displayId":"s0"},"s1":{"top":84,"left":892,"displayId":"s1"},"s5":{"isAccept":true,"top":234,"left":908,"displayId":"s5"},"s2":{"top":435,"left":216,"displayId":"s2"},"s3":{"top":434.6000061035156,"left":900,"displayId":"s3"}},"transitions":[{"stateA":"start","label":"ϵ,ϵ,#","stateB":"s4"},{"stateA":"s0","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s0","label":"B,A,ϵ","stateB":"s1"},{"stateA":"s0","label":"ϵ,A,A","stateB":"s5"},{"stateA":"s1","label":"B,A,ϵ","stateB":"s1"},{"stateA":"s1","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s1","label":"ϵ,#,#","stateB":"s4"},{"stateA":"s1","label":"ϵ,A,A","stateB":"s5"},{"stateA":"s2","label":"B,ϵ,B","stateB":"s2"},{"stateA":"s2","label":"A,B,ϵ","stateB":"s3"},{"stateA":"s2","label":"ϵ,B,B","stateB":"s5"},{"stateA":"s3","label":"B,ϵ,B","stateB":"s2"},{"stateA":"s3","label":"A,B,ϵ","stateB":"s3"},{"stateA":"s3","label":"ϵ,#,#","stateB":"s4"},{"stateA":"s3","label":"ϵ,B,B","stateB":"s5"},{"stateA":"s4","label":"A,ϵ,A","stateB":"s0"},{"stateA":"s4","label":"B,ϵ,B","stateB":"s2"}],"bulkTests":{"accept":"ABB\nAABBBB\nABBB\nAAABBB\nAABBBB\nBAAA","reject":"AABB\nAAABBB"}}


A,ϵ,A
B,A,ϵ
B,A,ϵ
s0
s1
A,ϵ,A
ϵ,#,#
ϵ,A,A
A,ϵ,A
ϵ,A,A
s4
s5
start
ϵ,ϵ,#
B,ϵ,B
ϵ,B,B
ϵ,#,#
ϵ,B,B
B,ϵ,B
A,B,ϵ
A,B,ϵ
s2
s3
B,ϵ,B