PS1 Theory

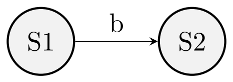
Erlend Paulsen Skaaden

Task 1

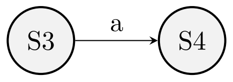
Subtask 1: RegEx

The following RegEx will suffice.
R = `(a?a?b)*a?a?`

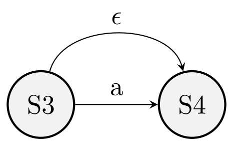
Subtask 2: RegEx to NFA



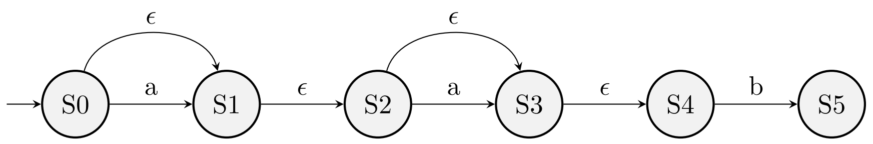
`b`



`a`

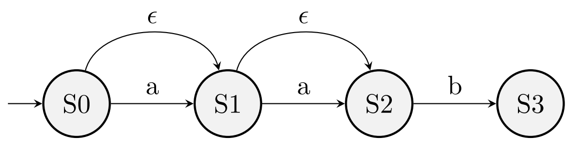


`a?`

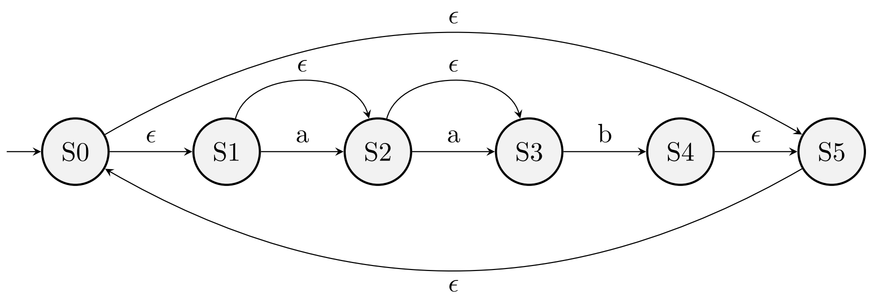


`a?a?b`

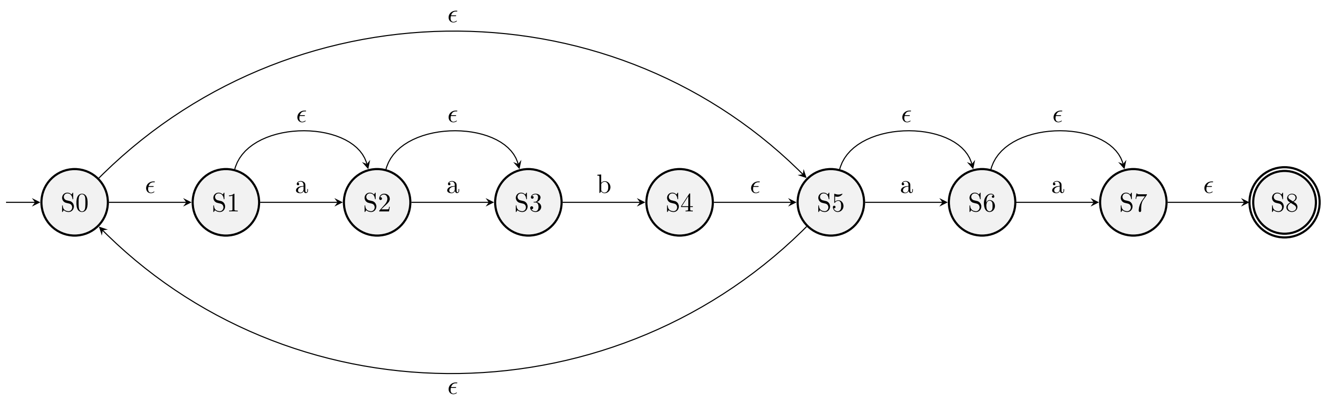
Which can be shortend to:



`a?a?b`

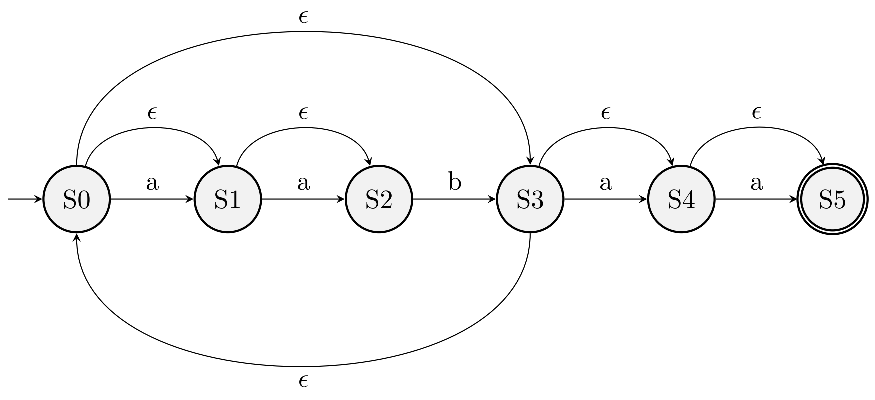


`(a?a?b)*`



`(a?a?b)*a?a?`

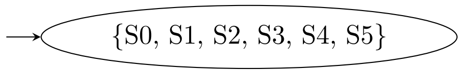
Removing unnecessary ϵ-edges gives the NFA:



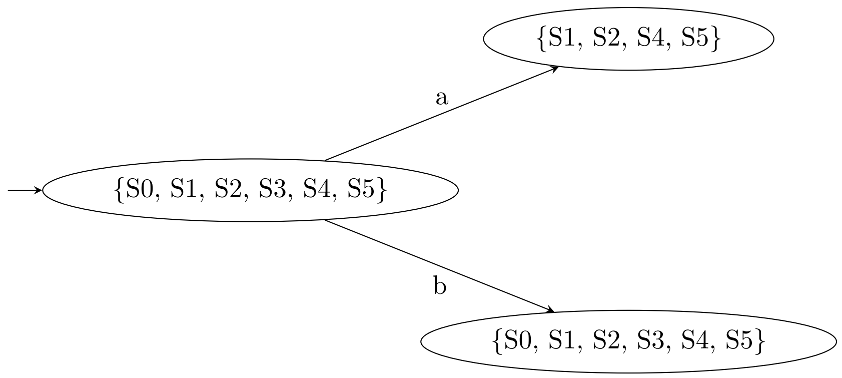
`(a?a?b)*a?a?`

Subtask 3: NFA to DFA

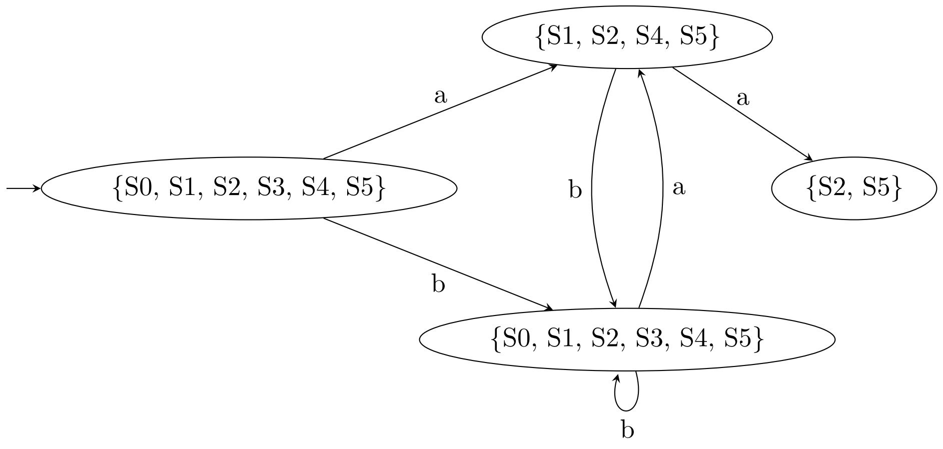
Firstly, we have to find the ϵ-closure:
$closure(\{S0\}) = \{S0, S1, S2, S3, S4, S5\}$



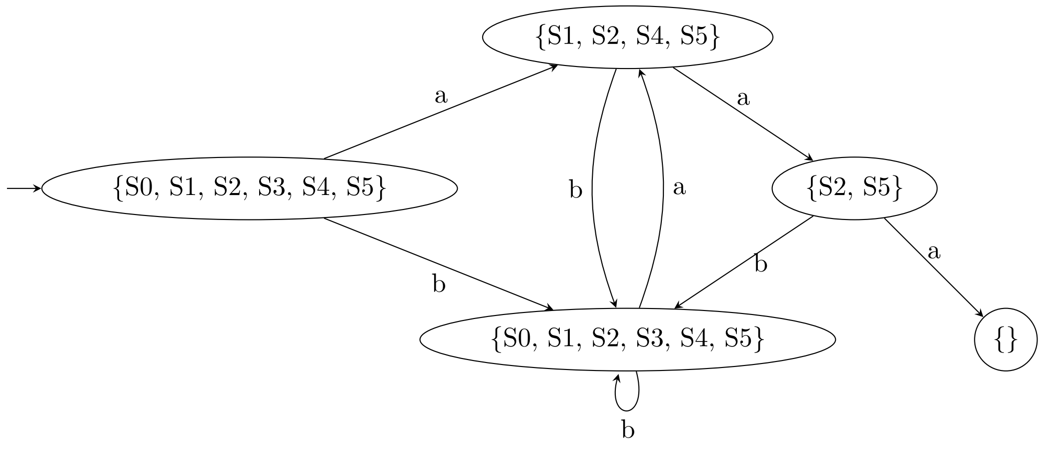
Now we find where we can end up for each input.
$move(\{S0, S1, S2, S3, S4, S5\}, a) = \{S1, S2, S4, S5\}$
$move(\{S0, S1, S2, S3, S4, S5\}, b) = \{S3\}$
We also need to add the ϵ-closure for each of the new sets.
$closure(\{S1, S2, S4, S5\} = \{S1, S2, S4, S5\}$
$closure(\{S3\} = \{S0, S1, S2, S3, S4, S5\}$



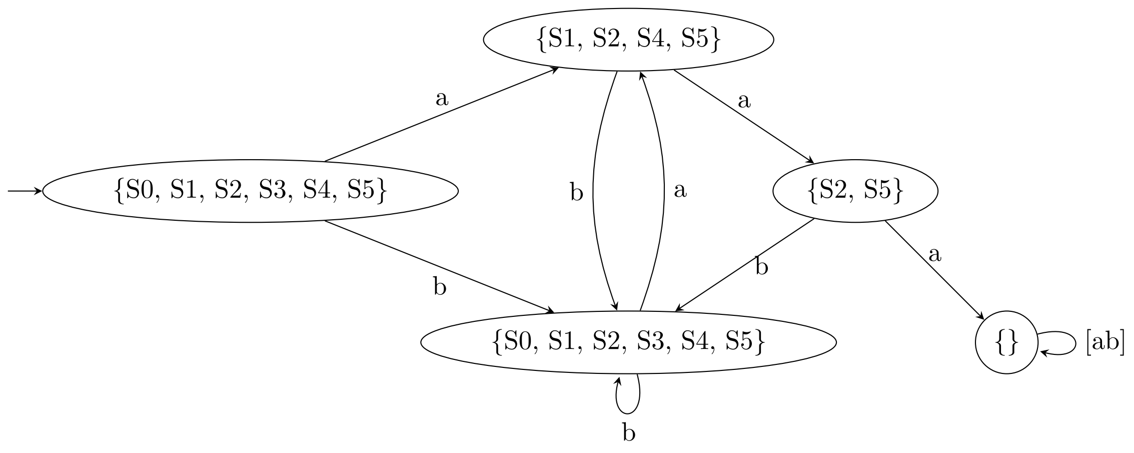
We now do the same for $\{S0, S1, S2, S3, S4, S5\}$ and $\{S1, S2, S4, S5\}$.
$closure(move(\{S1, S2, S4, S5\}, a)) = \{S2, S5\}$
$closure(move(\{S1, S2, S4, S5\}, b)) = \{S0, S1, S2, S3, S4, S5\}$
$closure(move(\{S0, S1, S2, S3, S4, S5\}, a)) = \{S1, S2, S4, S5\}$
$closure(move(\{S0, S1, S2, S3, S4, S5\}, b)) = \{S0, S1, S2, S3, S4, S5\}$



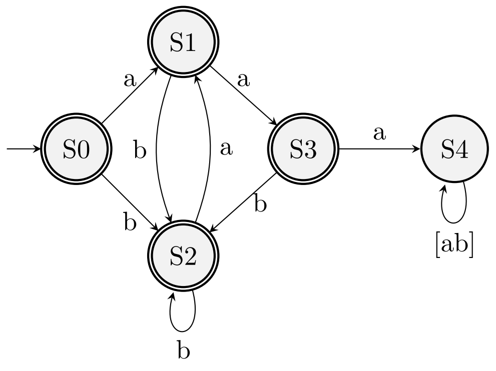
Next we do it for $\{S2, S5\}$:
$closure(move(\{S2, S5\}, a)) = \{\}$
$closure(move(\{S2, S5\}, b)) = \{S0, S1, S2, S3, S4, S5\}$



Since every state needs exactly one edge per input, we add loops for a and b input. Note that [ab] is a shorthand for a or b.



Since every DFA state contains the NFA accepting state (unless for the empty DFA state), every DFA state becomes an accepting state. Finally we can name the states, and obtain the DFA:

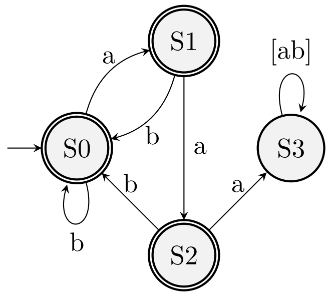


DFA for the regex: `(a?a?b)*a?a?`

Subtask 4: Minimizing the DFA

We start by grouping the states into accepting and non-accepting states.
$Accepting = \{S0, S1, S2, S3\}$
$Nonaccepting = \{S4\}$
We can see from the DFA that $S0$ and $S2$ are equivalent, that is, they both transition to the same states with the same inputs. None of the other pairs of states are equivalent. We can split these into new groups:
$G1 = \{S0, S2\}$
$G2 = \{S1, S3\}$
$G3 = \{S4\}$

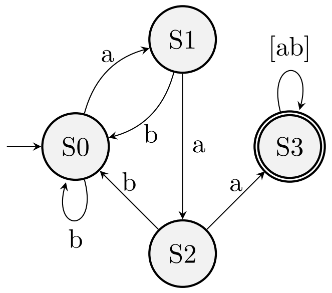
We continue to find differences in the groups until all groups contain only equivalent states. Notice that the pair in $G2$ are not equivalent. We again split this group, so we obtain:
$G1 = \{S0, S2\}$
$G2 = \{S1\}$
$G3 = \{S3\}$
$G4 = \{S4\}$
These groups contain only identical states, and can be our new states. Our minimized DFA is then:



Minimized DFA

Subtask 5: $\mathcal{L}$*

I assume that you can invert the accepting and rejecting states. You then get a single accepting state that can only be reached by creating a sequence of 3 a's. When the state is reached, you can create any sequence of a's and b's. The inverted DFA will look like:



Inverted DFA

A regex that requires a sequence of 3 a's at one point can be: `((a|b)*aaa(a|b)*)+`, which I found mostly by looking at the reversed DFA. It was easier finding and constructing the inverted DFA.

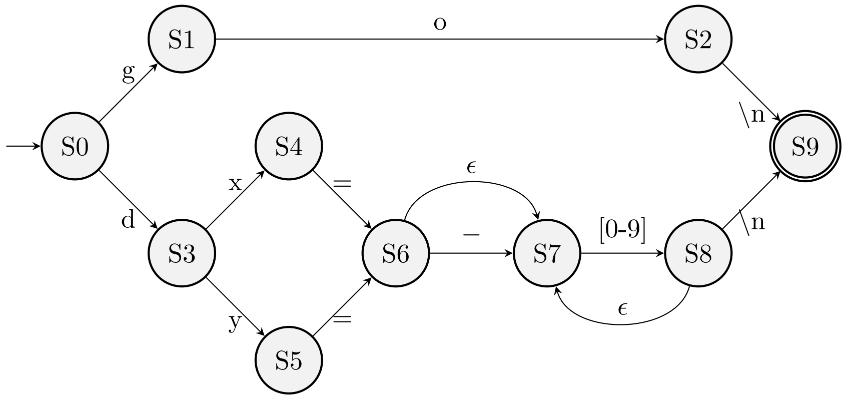
Task 2

Subtask 1: RegEx

The following regexes will suffice, assuming that e.g dx=0 is allowed:
`dx=-?[0-9]+\n`
`dy=-?[0-9]+\n`
`go\n`
We can combine the regexes to match a single line:
`(go|d(x|y)=-?[0-9]+)\n`

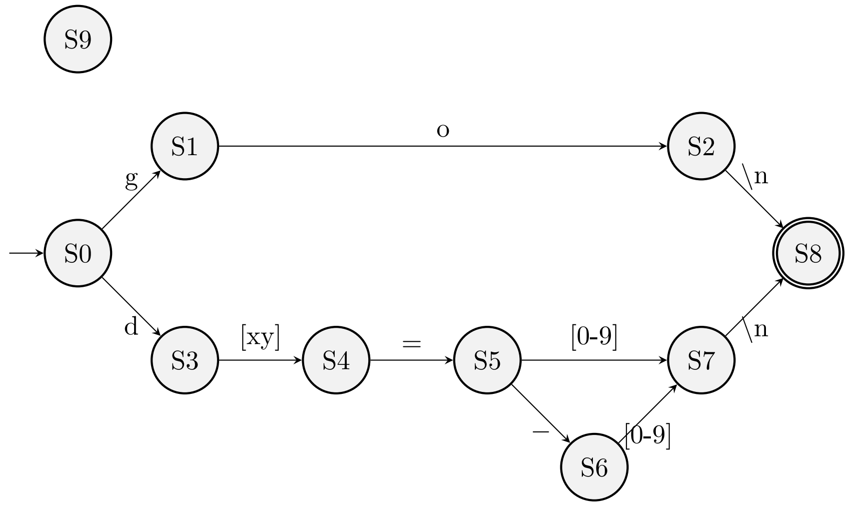
Subtask 2: DFA

From the regex above, I first created the NFA:



NFA (minimized) for `(go|d(x|y)=-?[0-9]+)\n`

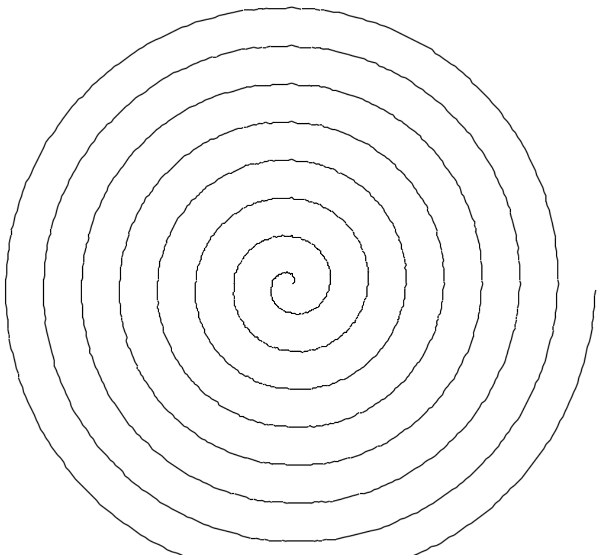
I then created the DFA and minimized it. In the drawing, I have not added the lines to the invalid state, because it would lead to utter chaos, so there are implicit lines from each state to the invalid state, where the characters that lead to the invalid state are all the symbols in the language, minus the characters that are relevant for a given state (i.e the edges). The invalid state is S9. The following is a minimized DFA for the language:



DFA (minimized) for `(go|d(x|y)=-?[0-9]+)\n`

Subtask 3: Implementation

The rest of the assignment is in the file scanner.c. The error was on line 5890. It gave this spiral:



Spiral created by spiral.txt