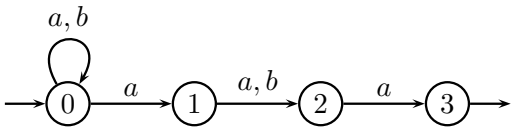
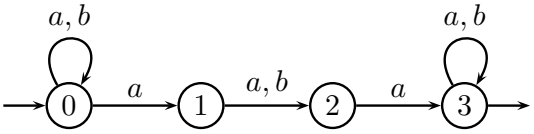
Compilation : TD n° 3

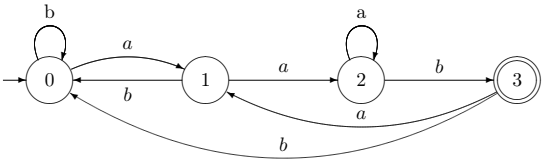
Devan SOHIER

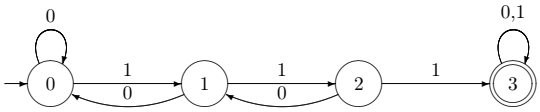
Exercice 1

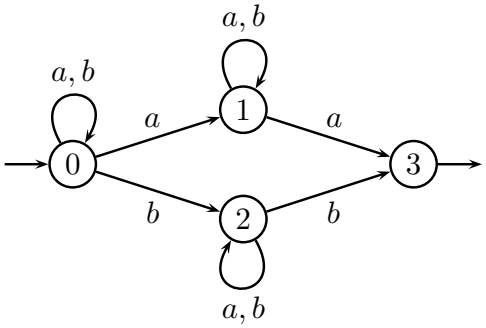
Pour chacun des automates suivants, donnez sa description formelle, et exécutez le sur un mot d'entrée accepté, et un mot refusé.

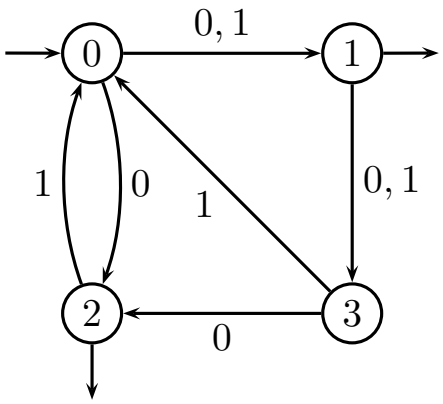












Exercice 2

Construisez par la méthode des automates standards des automates reconnaissant les langages définis par les expressions régulières ci-dessous :

1. $(a|b)^*$;
2. a^*b ;
3. ba^* ;
4. $a^*b|ba^*$;
5. $(a^*b|ba^*)^*$.
6. $1(0|1)^*|0$;
7. $1(0|1)^*1|1$;
8. $(a|b)^*aa(a|b)^*$;
9. $(b|ab)^*(a|\varepsilon)$;
10. $(a|b)(a|b|c)^*|\varepsilon$;
11. $a^+b|ab^+$;
12. $\Sigma^*(a|b)\Sigma^*$;
13. $(a^*b)^*$;
14. $\Sigma^*(a|b)\Sigma^*$.

Pour chaque automate, vous donnerez sa description formelle, et l'exécuterez sur un mot d'entrée accepté, et sur un mot refusé.

Exercice 3

Déterminisez chacun des automates de l'exercice 1.

Exercice 4 : analyse lexicale

`Lecs` construit à partir des expressions régulières un automate non déterministe par la méthode de numérotation reconnaissant la fermeture itérative de leur disjonction, et associe à chaque état terminal les instructions correspondant à l'expression régulière

On considère toujours le programme ci-dessous, qui devra transformer la séquence `for 0a111 fort9a` en `BOUCLE NOMBRE VARIABLE VARIABLE`

```
for {écrire BOUCLE}
[a-z][a-z0-9]* {écrire VARIABLE}
[0-9]+ {écrire NOMBRE}
```

1. Pour chacune des trois expressions régulières du programme, écrire par la méthode des automates standards un automate fini reconnaissant le même langage.
2. Toujours par la méthode des automates standards, en faire la disjonction.
3. Déterminiser cet automate, et pour chaque état accepteur, indiquer l'action du programme qu'il faudra effectuer en prenant en compte les règles de priorité.
4. Mêmes questions pour les programmes `Lecs` écrits au TD précédent.