|  |  |
| --- | --- |
| Onderdeel | Opmerkingen |
| 1a. hardcoded voorbeelden van (N)DFA en regulier expressies | Ja, maar deze zijn niet in de GUI zichtbaar. We zijn begonnen met een console app en op het laatst is dat veranderd naar een windows form app. Als de applicatie wordt gestart zie je in “output” hardcoded voorbeelden  Er is wel een knop voor een standaard DFA |
| 1b. Optie: parser voor reguliere expressies | Zit erin, maar haakjes in haakjes pakt hij niet zo goed. Zoals (a(b)\*)\* |
| 2a. woorden genereren aan de hand van de ingevoerde reguliere expressie  Woorden genereren die niet in de taal zit | Hebben we niet |
| 2b. Voor DFA: testen of een woord wel/niet geaccepteerd wordt | Hebben we niet |
| 3 a. Constructor operaties DFA om snel test dfa's te maken:   * begint met xyz * eindigt op xyz * bevat xyz | Hebben we niet |
| 3 b. Operaties op DFA (tupel-constructie):   * en * of * niet | Hebben we niet |
| 4. Thompson constructie | Zit er impleciet in |
| 5. en 6.  NDFA->DFA | Werkt, op epsilonovergangen na |
| 6. Minimalisatie  (via reverse)  Bonus: ook algoritme uit het dictaat (via opsplitsen tabel) | Het doet wel wat maar het wordt niet korter.  De reverse werkt |
| 8. TestApplicatie:   * IO Inlezen en wegschrijven van automaten, grammatica en reguliere expressies * GUI | Er zit fileIO in, er wordt steeds in een bestand een dfa gezet dat getekend kan worden met graphviz |
| Totaalscore  Eindcijfer: aantal punten / 12 |  |

Verder is het mogelijk om reguliere expressies in te voeren en die om te zetten in reguliere grammatica’s