PROGRAMMIERABGABE
Abgabefrist: Montag, 04.06.2018, 10 Uhr

Einführung in die theoretische Informatik

Sommersemester 2018 – Hausaufgabenblatt Lösungsskizze 7

Programmierabgabe

Für die Korrektur Ihrer Programmierabgaben wird TUMjudge(`https://judge.in.tum.de/theo/public/`) verwendet. Auf dem ersten Hausaufgabenblatt werden die für Sie relevanten Funktionen des Webinterfaces erklärt. Weitere Details finden Sie auf den Folien des Praktikums Algorithms for Programming Contests von 2017(`https://www7.in.tum.de/um/courses/theo/ss2018/materials/judge.pdf`).

Laden Sie das Codegerüst für die siebte Hausaufgabe auf `https://www7.in.tum.de/um/courses/theo/ss2018/homework/` herunter.
Die vorimplementierten Klassen des Codegerüstes kennen Sie bereits aus den vorigen Progarmmieraufgaben.
Hinweis: Sie dürfen den Code des templates verändern. Allerdings sollten Sie main-Methode und parser so lassen, wie sie sind.

AUFGABE 7.1. (*CanProduce*) 2 Punkte

- Implementieren Sie die Methode canProduce(Grammar g, String w) in der Klasse CanProduce. Die Methode soll true zurück geben, wenn die Grammatik g das Wort w erzeugen kann, ansonsten false.
- Sie dürfen davon ausgehen, dass die Grammatik eine kontextfreie Sprache erzeugt und dass auf der linken Seite der Produktionen immer genau ein Nichtterminal steht.
- Laden Sie dann für Problem A (CanProduce) *alle* Dateien hoch, die für diese Aufgabe relevant sind, also Production, Grammar und CanProduce.

AUFGABE 7.2. (*Minimize*) 2 Punkte

- Implementieren Sie in der Klasse Minimize die Methode minimize(DFA d). Die Methode soll für einen DFA d den minimierten DFA zurück geben, der dieselbe Sprache wie d akzeptiert.
- Laden Sie dann für Problem B (minimize) *alle* Dateien hoch, die für diese Aufgabe relevant sind, also State, Transition, DFA, NFA, EpsilonNFA, Parser und Minimize.
- Da Ihre Ergebnis auf Sprachgleichheit getestet wird, übernimmt die main-Methode hier wieder das Einlesen der Zeilenzahl und das Ausgeben von Case #1: vor dem Ergebnis-DFA.

AUFGABE 7.3. (*CanonicDFA*) 1 Punkt

- Implementieren Sie in der Klasse CanonicDFA die Methode canonicDFA(DFA d). Die Methode soll für einen DFA d den kanonischen DFA zurück geben, der dieselbe Sprache wie d akzeptiert. Beschriften Sie hierzu jeden Zustand mit dem kürzesten, lexikalisch kleinsten Element der korrespondierenden Äquivalenzklasse. Repräsentieren Sie beim Beschriften das leere Wort als “eps”
- Laden Sie dann für Problem C (canonicdfa) *alle* Dateien hoch, die für diese Aufgabe relevant sind, also State, Transition, DFA, NFA, EpsilonNFA, Parser und CanonicDFA.