Dokumentace úlohy DKA:
Determinizace konečného automatu v Python 3 do IPP 2011/2012

Jméno a příjmení: Peter Lacko
Login: xlacko06

Program začíná již tradičně zpracováním parametrů příkazové řádky a to standardní utilitou `getopt()` přičemž zde nastavíme příznaky určující operace které se mají vykonat. Následně sa program pokusí otevřít vstupní a výstupní soubor a obsah vstupního souboru se načte do řetězce. Tento řetězec je dále zpracován konečným automatem kterého počáteční stav je nastaven na `INIT`.

Syntaktická kontrola

Automat čte řetězec znak po znaku a pokud narazí na znak indikující začátek popisu automatu - `'('`, konec popisu automatu - `')'`, začátek množiny - `'{'` nebo konec množiny - `'}'` přejde do příslušného stavu (určujícího množinu nebo pozici v množině) a pomocí regulárních výrazů zpracuje jednotlivé prvky. Pro čtení (a vypuštění) nevýznamových bílých znaků a komentářů jsou také vytvořeny speciální stavy. V případě syntaktické chyby automat přechází do stavu `FALSE` a program se ukončí s chybovým hlášením a návratovou hodnotou 40.

Reprezentace množin stavů a pravidel

Množiny stavů a vstupních symbolů jsou konstruovány jako "klasická" množina řetězců. Pro reprezentaci množiny pravidel byla vytvořena třída, jejíž instance jsou prvky této množiny. Třída má implementovány metody pro inicializaci obejktu (`__init__()`), reprezentaci objektu jako řetězce (`__repr__()`, `__str__()`), kvůli množinovým operacím také hash tohoto řetězce (`__hash__()`), pro možnosti porovnání a setřízení metody `__lt__()`, `__gt__()`, `__eq__()` ..., a napokon pro převod pravidla na malá písmena je tu metoda `lower()`.

Sémantická kontrola

Po úspěšném zpracování vstupního konečného automatu se provádí případná změna velikosti písmen na malá a kromě sémantických kontrol popsánych v zadaní také kontrola pravidel, tj. jestli se počáteční a koncový stav pravidla nachází v množině stavů a jestli se vstupní symbol nachází ve vstupní abecedě (nebo je to prázdný řetězec).

Odstránění ε-přechodů, determinizace a výpis konečného automatu

Díky vlastnostem jazyka Python a podkladem z kurzu IFJ nebyl problém tyto algoritmy jednoduše zapsat. Nové stavy jsou vytvářeny pomocí funkce `gen_state()`, která spojí řetězce jí prodané v množině parametrem do jednoho. Za zmínku zde stojí také nutnost vytvoření tzv. frozen setů z nově vytvořených stavů (Q''), kvůli vložení této množiny jako prvku do jiné množiny (Q_{new}). Po proběhnutí algoritmů následuje už jenom výpis finální podoby automatu a ukončení skriptu.