

Języki formalne i metody kompilacji

Laboratorium nr 13

Stefan Sokołowski

Zadanie 1:

Proszę ściągnąć do swojego katalogu napisany w Bisonie parser i interpreter prostego języka¹. Skompilować go, uruchomić i zapoznać się z jego działaniem. Zapoznać się też z jego budową wewnętrzną.

Zadanie 2:

Na wzór programu z zad. 1 napisać i uruchomić własny parser wyrażeń **logicznych**. Powinien on dopuszczać zmienne zdaniowe oraz stałe zdaniowe (wartości prawdy i fałszu), spójniki logiczne (poniżej uporządkowane od najsilniej do najsłabiej łączących):

1. negacja $\sim$,
2. koniunkcja $\&$,
3. alternatywa $|$,
4. implikacja $>$,
5. równoważność $\#$,

oraz możliwość podstawiania wartości wyrażeń logicznych pod zmienne i wykorzystywania potem tych zmiennych.

Zadanie 3:

Plik `kompil.y`² zawiera gramatykę atrybutową potrzebną do stworzenia kompilatora dla bardzo prostego języka programowania zawierającą proste przypisania oraz ograniczoną postać pętli `while`. Proszę wypróbować ten program na jakichś **własnych** danych (różnych od `prog1` i `prog2` w tym samym katalogu); a następnie wzbogacić język przez dodanie

(a) mnożenia:

```
⟨komenda⟩ ::= ⟨zmienna⟩ = ⟨zmienna⟩ * ⟨zmienna⟩ ;  
           |   ⟨zmienna⟩ = ⟨zmienna⟩ * ⟨liczba⟩ ;
```

(b) mniejszości (do podanej zmiennej wpisuje 0 lub 1 zależnie od tego, czy mniejszość jest spełniona):

```
⟨komenda⟩ ::= ⟨zmienna⟩ = ⟨zmienna⟩ < ⟨zmienna⟩ ;  
           |   ⟨zmienna⟩ = ⟨zmienna⟩ < ⟨liczba⟩ ;
```

¹Patrz `iis.ans-elblag.pl/~stefan/Dydaktyka/JezForm/Wyklady/13-bizony/3-pjo.y`; gramatyka języka podana jest w `iis.ans-elblag.pl/~stefan/Dydaktyka/JezForm/Slajdy/13a.pdf` na str. 10.

²Patrz `iis.ans-elblag.pl/~stefan/Dydaktyka/JezForm/Wyklady/13-bizony/4-Kompilacja/kompil.y`.

(c) komendy **warunkowej** postaci

$$\langle \text{komenda} \rangle ::= [\langle \text{zmienna} \rangle] \{ \langle \text{komendy} \rangle \} \{ \langle \text{komendy} \rangle \}$$

w której wykonuje się tylko zawartość pierwszej klamry albo tylko drugiej klamry, zależnie od tego, czy zmienna w nawiasie prostokątnym jest niezerowa, czy zerowa.

Proszę przetestować działanie komend warunkowych **zagnieżdżonych** w innych komendach warunkowych, albo w komendach pętli, oraz na odwrót — komend pętli zagnieżdżonych w komendach warunkowych. Oczywiście będzie potrzebne wstawienie instrukcji skoków między przekłady zawartości obu klamer; a to będzie wymagało przesuwania już istniejącego kodu w pamięci. Proszę wzorować się na komendzie pętli

$$(x) \{ \dots \}$$

która tam już jest zdefiniowana.

Wskazówka:

Nie próbować uruchomić wszystkich rozszerzeń na raz. Lepiej wykonać tylko punkt (a) i starannie go przetestować, a dopiero potem przystąpić do punktu (b), itp.