

Kompilatory

Laboratorium 3

Piotr Błaszyński

3 marca 2025

Zadania (wyjaśnienie w dalszej części dokumentu):

- przygotować gramatykę dla pojedynczego wyrażenia składającego się ze zmiennych, liczb i operatorów (plik `def.y` - przykładowa gramatyka w katalogu `z5`),
- przekazać wartości semantyczne identyfikatorów i liczb (całkowitych i rzeczywistych) z analizatora leksykalnego do analizatora składniowego,
- zaimplementować funkcję `main` (tylko jedna wersja w `def.y`), z obsługą parametrów wywołania (`argc`, `argv`),
- zapisać do pliku wartości semantyczne poszczególnych identyfikatorów i liczb oraz operatory w kolejności dopasowywania,
- przetestować działanie dla wyrażeń składających się z kilku (8-10) elementów.

Reguły gramatyki języka przetwarzane przez generator bison składają się z:

- nazwy symbolu nieterminalnego,
- znaku dwukropka,
- definicji składającej się z symboli terminalnych i nieterminalnych.

Alternatywne definicje są od siebie oddzielone pionową kreską (`'|'`) a po ostatniej alternatywie (dla porządku) powinno się postawić średnik (`;`). Jako symbol startowy gramatyki jest wybierany symbol nieterminalny znajdujący się na początku, chyba, że wskażemy go wprost (dyrektywa `%start`). Kolejność pozostałych reguł nie jest istotna, jednak warto żeby reguły były zapisane w uporządkowany sposób. Do definiowania reguł jako symboli terminalnych używać należy uprzednio

zdefiniowanych tokenów zdefiniowanych w pierwszej sekcji oraz symboli jednoznacznych (w apostrofach). Po każdej definicji (również po każdym jej fragmencie) można zapisać akcję (semantyczną) w postaci kodu C/C++, która zostanie wywołana jeżeli reguła zostanie dopasowana.

Przykładowa definicja symboli nieterminalnych czynnik i składnik:

```
skladnik
    : skladnik '*' czynnik    {printf("_*_\n");}
  | skladnik '/' czynnik    {printf("_/_\n");}
  | czynnik
    ;
czynnik
    : ID                      {printf("zmienna\n");}
  | LC                        {printf("liczba\n");}
  | '(' wyr ')'              {printf("nawiasy\n");}
    ;
```

Do przekazania wartości semantycznych z analizatora leksykalnego do składniowego należy użyć unii *yylval* (jej domyślna nazwa w kodzie generowanym przez flex). Pola tej unii definiuje się w pliku dla bisona (np. *def.y*). Wartości leksemów dla liczb i identyfikatorów (dla pozostałych leksemów również) jest przechowywana w zmiennej *ytext*.

Przykładowa definicja unii do przekazywania wartości semantycznych, tokeny których wartość semantyczną będziemy przekazywać muszą mieć określony typ:

```
%union
{
    char *text;
    int ival;
};
%token <text> ID
%token <ival> LC
```

W analizatorze składniowym do wartości semantycznej odwołać się można przy użyciu symbolu \$, i liczby. Liczba oznacza miejsce leksemu, którego wartość semantyczną chcemy otrzymać, w regule (więc najczęściej jest to \$1, ale np. w przypadku deklaracji tablic może być to \$2 lub \$3)

Przykładowe odwołanie do wartości semantycznej identyfikatora.

```
czynnik
    : ID                      {printf("id:_%s\n", $1);}
    ;
```