Notatki do Metod Programowania

Michał Bronikowski

24 lutego 2017

Spis treści

1 Wstęp do Prologa

1.1 Kompilowanie

Kompilator SWI-PROLOG.
W terminalu komenda:

- swipl lub prolog

?- //Działa
Kompilacja:
Pliki zapisuję z rozszerzeniem .pl. W otwartej "maszynie" prologa wpisuję:
?- [p1]. //p1 - nazwa pliku

1.2 Fakty,Zmienne,Koniunkcje

lubi(jan,maria).

- Fakt musi się kończyć kropką
- lubi(jan,_) - chodzi nam tylko o odpowiedź nie true oe false
- po uzyskaniu odpowiedźi jak klikniemy ';' to uzyskamy kolejną o ile istnieje kończymy Enterem
- koniunkcje oznaczamy ','

Przykład: Plik p1.pl

```
lubi(jan,reksio).
lubi(reksio,bartek).
lubi(jan,szklanka).
lubi(jan,beata).
```

Przykład: Działanie

```
?- [p1].
?- lubi(jan,beata), lubi(reksio,bartek).
true
```

1.3 Reguły

W prologu reguł używa się do zapisania, że fakt zależy od grupy innych faktów.(W języku polskim do stosowania reguł używa się "jeśli").

Przykład: Kot lubi każdego kto lubi mleko

```
czyli:
Kot lubi wszystko, jesli to lubi mleko,
Kot lubi X, jesli X lubi mleko.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
lubi(kot,X) :- lubi(X, mleko).
```

1.4 Struktury

Struktury w Prologu zapisujemy podając funktor oraz jego składniki. Nazwa funktor odpowiada typom z tradycyjnych języków programowania.Składniki ujęte są w nawiasach okrągłych i oddzielone od siebie przecinkami. Funktor umieszcza się przed nawiasem otwierającym.

Przykład: Strukturę można rozbudowywać

```
posiada(jan,rower(wigry(niebieski),1991).
```

Jan posiada rower marki wigry koloru niebieskiego z 1991 roku

1.5 Operatory

Operatory nie powodują wykonanai jakichkolwiek obliczeń 3+4 to nie 7 to term +(3,4).

- X =:= X - X i Y są tę samą liczbą
- X =\= X - X i Y są różnymi liczbami
- X < Y - X jest mniejsze od Y
- X > Y - X jest większe od Y
- X =< Y - X jest mniejsze równe Y
- X >= Y - X jest większe równe Y

Operator **is** operator infiksowy jego prawy argument jest termem, który ma być zinterpretowany jako wyrażenie arytmetyczne. Aby uzgodnic wyrażenie Prolog najpierw oblicza wyrażenie arytmetyczne, a wynik dopasowuje do lewego argumentu

Przykład: Operator "is"

```
?- X is 2+5.
X = 5
```

Po prawej stronie operatora is można używac takich wyrażeń jak:

- +

- -
- *
- / - iloraz
- // całkowity iloraz
- mod reszta z dzielenia

Przykład: Dodawanie

```
dodaj(X,Y,Z) :-Z is X + Y.
////
?- dodaj(2,3,A).
A=5.
```

1.6 Listy

Lista to struktura danych, która jest ciągiem uporządkowanych elementów(dowolne termy). Głowa listy to pierwszy element, ogon to całą reszta. Zapis `[X|Y]` utożsamaia X z głową listy a Y z ogonem.

Przykład: Utożsamianie głowy i ogona

```
a([1,2,3]).
~~~~~~~~~~~~~~~

?- p([X | Y]).
X=1   Y=[2,3].
```

1.7 Przeszukiwanie rekurencyjne

Sprawdzmy czy jakis obiekt X jest członkiem listy Y. Sprawdzamy dwa warunki:

- X jest elementem Y, gdy jest jego głową
- X jest elementem Y, gdy X nalezy do ogana tej listy

Przykład: Rekurencja

```
member(X,[X,_]). /*X nalezy jesli jest glowa*/
member(X,[_ | Y]) :- member(X,Y)./*X nalezy jesli jest w ogonie*/
```

Drugi warunek za każdym razem wywołuje krótszą listę. Więc albo uruchomiona zostanie pierwsza klauzula member albo jako drugi argument zostanie przekazana lista długości 0. Sa to warunki graniczne kończące wywoływanie celów member.