# Теория

ПРОЛОГ е език за декларативно програмиране, основан на предикатно смятане от първи ред. Програмите на ПРОЛОГ представляват поредица от *правила* и *факти*, които общо наричаме *клаузи*.

Фактите дават информация за свойства на обектите и връзки между тях:

man(john).
woman(jane).
loves(jane,john).
gives(john,jane,ring).
has(ring,diamond).

Обектите (john,jane,ring,diamond), наричани още *константи*, се записват с латински букви и цифри, като задължително започват с малка латинска буква. Свойствата и връзките (man,woman,loves,gives,has), наричани още *предикати,* се записват с малка латинска буква в началото както константите, като след тях в скоби със запетаи се изброяват обектите, за които се отнасят. Целият запис се нарича *атом* (атомарна формула). Фактите представляват атом със знак за точка (.) накрая.

Правилата дават информация за условията, които са достатъчни за изпълняването на дадена връзка или свойство. Правилата имат *глава* (следствие), и *тяло* (опашка, предпоставки).

gives(john,X,ring) :- woman(X),loves(john,X).

В атомите освен константи могат да участват и *променливи* - започващи с главна латинска буква или знак за подчертаване (\_). Главата на правилото (gives(john,X,ring)) се състои от единствен атом и означава свойството или връзката, която ще е в сила, ако са изпълнени всички предпоставки на правилото. Тялото на правилото се състои от поредица от атоми, разделени със знак за запетая (,). Знакът, разделящ тялото и глава се чете " е вярно, ако". Правилата завършват с точка.

Правилото има логически смисъл на импликация, която се чете "ако **<тяло>** е вярно, то е вярно **<глава>**". Знакът :- представлява всъщност стрелка на импликация, сочеща наляво (<=). Фактите могат да се третират като правила без тяло, т.е. безусловни правила. Променливите, срещащи се в главата на правилото се четат "за всяко" и имат универсален квантор (), а променливите които се срещат само в тялото се четат "съществува" и имат екзистенциален квантор ().

С ПРОЛОГ се работи в диалогов режим, като към програмите на пролог се задават *въпроси (цели) -* поредица от атоми, разделени със запетая. Интерпретаторът се опитва да намери отговора на въпроса, като използва дадените клаузи.

?-man(john).
**Yes**

?-woman(X).
**X = jane**;
**No**

?-gives(X,jane,Y),has(Y,diamond).
**X = john
Y = ring**(Enter)
**Yes**

Променливите в целта се четат "има ли?" и пред тях има квантор ().

Вариантите за отговор са:

* Интерпретаторът забива - няма отговор
* No - въпросът не може да бъде доказан от дадените клаузи, т.е. отговорът е отрицателен
* Yes - отговорът на въпроса е положителен, т.е. той може да се изведе (докаже) от дадените правила и факти.
* X = ...
Y = ...
Отговорът на въпроса е положителен, изредени са стойностите на променливите във въпроса, за които въпросът може да се изведе от дадените клаузи. ПРОЛОГ очаква да въведете Enter за край, при което извежда Yes или да въведете точка и запетая (;), което води до търсене на нови стойности на променливите (преудовлетворяване), за които отговорът е положителен. Ако такива няма, извежда се No, което следва да се разбира като *няма повече*.

Процесът на отговор на въпрос от ПРОЛОГ се нарича *извод* (резолюция). Този процес не винаги завършва:

p(X) :- q(X).
q(X) :- p(X).
p(a).
?-q(a).
*(ПРОЛОГ забива)*

Причината е, че при отговор на въпросите, ПРОЛОГ претърсва базата от данни **отгоре-надолу** и проверява всяко правило **отляво-надясно**. На всяка итерация ПРОЛОГ сравнява първия (най-левия) атом на текущата цел с главата на първото (най-горното) правило (или с факт). Ако те могат да се "напаснат" (*унифицират*) за някоя стойност на променливите, това се прави. От текущата цел се премахва първия атом, а накрая се добавя тялото на правилото (или нищо, ако е факт). При достигане на *празна цел*, ПРОЛОГ отговаря с Yes, а ако това не е възможно при обхождане на всички клаузи в програмата - с No.

Процесът на извод се визуализира графично с т.нар. *дърво на извод*по следния начин:



За да се избягват двусмислици и грешки, при всяко използване на правило, променливите му се преименуват (X,X',X'',...).

Понякога просто пренареждане на клаузите в програмата може да доведе до това, един въпрос да получи отговор, вместо ПРОЛОГ да забие. Едно стилистично указание е клаузите за даден предикат да се пишат заедно, като първо се изреждат фактите, а после правилата:

p(a).
p(X) :- q(X).
q(X) :- p(X).
?-q(a).
**Yes**



За съжаление не всяка програма може да се нареди така, че ПРОЛОГ да може да отговаря на всички логически изводими въпроси. Това свойство на ПРОЛОГ се нарича *полуразрешимост*.

Накратко, ПРОЛОГ строи дърво на извод, което претърсва в дълбочина отляво надясно до достигане на празна цел (успех) или до изчерпване на дървото, ако е крайно (неуспех). В случай на преудовлетворяване, ПРОЛОГ извършва връщане назад (backtracking) и продължава търсенето от последната цел преди празната.

# Задачи

1. Нека е дадена програмата:

man(john).
woman(jane).
loves(john,wine).
loves(jane,wine).
loves(john,X) :- loves(X,wine).

а) Какво ще изведе ПРОЛОГ при въпроса:
?-loves(john,X).
б) Направете дърво на извод.
2. Дадена е база данни от факти от вида:

man(john).
man(tom).
woman(jane).
woman(mary).
parent(john,mary).
parent(jane,mary).
parent(mary,tom).

а) Напишете правила, описващи отношенията: майка, баща, баба, дядо, сестра, брат, леля, чичо, първи и втори братовчед. Тествайте програмата над собственото си родословно дърво, като задавайте въпроси от вида: кой е чичото на X? кой е брат на Y и баща на Z?

б) Нека разгледаме дефиницията на отношението "брат":

brother(X,Y) :- man(X),parent(Z,X),parent(Z,Y).

Какъв ще е отговорът на въпроса

?-brother(X,tom).

Как да избегнем проблема, който се получава? (Упътване: използвайте отношението \=). Защо при въпроси в родословни дървета при преудовлетворяване с ; братята на даден обект се дават от ПРОЛОГ два пъти като отговор? Начертайте дърво на извод.