# Теория

### Анонимни променливи

В ПРОЛОГ една от променливите има специално значение. Тя се означава със знак за подчертаване (\_). Всяка употреба на обикновените променливи се замества с един и същи обект. Например в правилото

loves(john,X) :- loves(X,wine).

променливата X означава един и същи обект - този, който ако обича вино е обичан от Джон. Всяко срещане на анонимната променлива може да се замести с различен обект. Ако в това правило използваме анонимна променлива:

loves(john,\_) :- loves(\_,wine).

то придобива следния смисъл: "Ако някой обича вино, то Джон обича някого (не непременно този, който обича вино)". Ефектът би бил същият, ако използваме различни променливи:

loves(john,Y) :- loves(Z,wine).

Добрият стил на програмиране на ПРОЛОГ изисква когато една променлива се използва само един път в правило, тя да се заменя с анонимна променлива.

Когато анонимна променлива се използва във въпрос, това означава, че ние не се интересуваме от конкретната стойност на променливата, за която отговорът е положителен. Например въпросът:

?-loves(john,X).
**X = jane**

се превежда като: "Кого обича Джон?", докато въпросът:

?-loves(john,\_).
**Yes**

има смисъл на: "Обича ли Джон някого?", на което ПРОЛОГ логично отговаря с **Yes**, без да има възможност за преудовлетворяване.

### Структури

Освен константи и променливи, обектите с които работи ПРОЛОГ могат да бъдат по-сложни и да съдържат повече информация, например:

book(war\_and\_peace,russian,author(leo,tolstoy))

Такива сложни обекти наричаме структури (термове). "Събирателните" названия (book, author), които свързват различните части на обекта в едно се наричат функтори (функционални символи). Обектът може да бъде произволно сложен, т.е. можи да има произволен брой функтори на произволна дълбочина. Тази възможност на ПРОЛОГ позволява да се строят по-сложни структури от данни - списък, стек, опашка, дърво и граф.

Всъщност този механизъм е толкова мощен, че може да се използва за дефиниция на естествените числа. Да се условим, че кодираме числото 0 с обекта o, а операцията "добавяне на единица" с s. Тогава числата могат да се представят като структури по следния начин:

0 -> o
1 -> s(o)
2 -> s(s(o))
3 -> s(s(s(o)))
...

Можем да дефиниране предикат, който разпознава кои структури са кодове на естествени числа:

nat(o).
nat(s(X)) :- nat(X).

Дефиницията на предакта е рекурсивна, понеже правилото има предикатния символ nat както в главата, така и в тялото.

Такъв предикат се нарича разпознавател, понеже при подадена структура разпознава дали тя е код на естествено число или не. Казваме, че предикатът разпознава множеството от структури - кодове на естествени числа.

Ако обаче зададем въпроса:

?-nat(X).
**X = o;
X = s(o);
X = s(s(o));**...

Ако преудовлетворяваме този въпрос достатъчно дълго можем да получим код на произволно голямо отнапред зададено число. В такъв случай казваме, че предикатът е генератор. Не всеки разпознавател е генератор, а някои генератори са само полу-разпознаватели - дават отговор **Yes**, ако структурата е разпозната, но забиват, ако не е.

Можем да дефинираме и предикати, които "смятат" с кодове на естествени числа:

sum(0,X,X) :- nat(X).
sum(s(X),Y,s(Z)) :- sum(X,Y,Z).

Този предикат може да смята по следния начин:

?-sum(s(o),s(s(o)),X).
**X = s(s(s(o)))**

т.е. задаваме въпроса "кое е числото, което е сума на 1 и 2"? Предикати, които по дадени аргументи могат да пресмятат резултат, подобно на функции се наричат изчислители.

# Задачи

1. Нека е зададена база от данни за оценките на студентите във ФМИ по следния начин с факти:

subject(<предмет>,<преподавател>).
student(<ф№>,<име>).
mark(<ф№>,<предмет>,<оценка>).

Да се напишат въпроси на ПРОЛОГ за следните задачи:
а) как се казват студентите, които са скъсани по предмета dis1?
б) какви са оценките на ф№ 43210 при преподавателя ivanov?
в) кой преподавател е писал и шестици и двойки по някой предмет?
г) как се казва студентът, който има 6 и 2 по два различни предмета, водени от един и същ преподавател?
д) какво е името на преподавателя, който е писал 6 на студент със същото име като неговото?
е) кои са предметите, по които няма скъсани студенти? а преподавателите, които не са късали?
Оформете някои избрани от вас въпроси като правила. Кога могат да се използват анонимни променливи?
2. Напишете предикат leq(X,Y), който разпознава тези двойки кодове на числа X и Y, за които X<=Y. Можете ли да напишете предиката така, че той да може да работи като генератор? Покажете при какви въпроси това е възможно.
3. Може ли предикатът sum да работи като разпознавател? а произволен предикат-изчислител?
4. Защо според вас е нужно тялото nat(X) в първия факт на дефиницията на sum? Покажете с пример какво ще се промени, ако то се премахне.
5. Работи ли sum и като генератор?
6. Напишете предикати за пресмятане на:
а) разлика (напишете програмата с една клауза, като използвате sum);
б) умножение (2 клаузи);
в) деление без остатък (1 клауза). Какво ще даде предикатът при опит за деление на 0?
г) частно и остатък при целочислено деление (по 1 клауза). Можете ли да напишете общ предикат за двете?
д) най-голям общ делител (3 клаузи).
7. Напишете предикати за генериране на:
а) четните числа;
б) нечетните числа;
в)\* точните квадрати.
Работят ли тези предикати като разпознаватели?