9. РАБОТА С ДИНАМИЧНИ ФАКТИ

Въведение. Пролог предоставя предикати за работа с динамични факти. По време на изпълнение на програмата вие можете да добавяте и изтривате факти за предикати, които са декларирани в домейна на секцията факти. Тъй като при работа с Prolog Interface Engine няма секции, възможно е да се добавят и отстраняват факти относно всички предикати.

Предикатът assertz(fact) добавя факт накрая след всички добавени вече факти за съответния предикат.

Предикатът retract(fact) отстранява първия факт, който съвпадне с посочения в скобите.

Предикатът save("име на текстов файл") запомня във файл динамично променените или добавени факти. При работа с Prolog Interface Engine се запомня програмата заедно с динамично променените или добавени факти.

Постановка. Работата с динамични факти ще покажем с примерната програма "Shop". Магазинът е за спортни стоки и нека са известни наличните артикули и тяхното количество. Програмата за обслужване на магазин трябва да има следните минимални възможности: да се отбелязват промените на количеството на наличните стоки при продажба и при зареждане; да се добавят нови артикули; да се получава разнообразна информация - за всички продавани артикули, за най-добре продаваните стоки, за стоките, които са се изчерпали, и т.н.

Упътване. Артикулите, наличните и продадените количества за всеки артикул са номерирани и се съхраняват като факти за предиката article(number, name, available, sold). Това дава възможност динамично да се изтриват, редактират и допълват артикулите, продадените и наличните бройки.

Заявката за закупуване се представя с правилото:

```
selling_request(Article,BrRequest):-
article(NArticle,Article,BrAvailable,BrSold),
available(BrRequest,BrAvailable),
BrSold1 is BrSold + BrRequest,
        BrAvailable1 is BrAvailable - BrRequest,
  retract(article(NArticle,Article,_,_)),
        assertz(article(NArticle,Article,BrAvailable1,BrSold1)),!.
```

Предикатът selling_request(Article,BrRequest) получава като аргументи името и количеството на стоката, която клиентът иска да закупи. Прочита от базата с факти има ли такава стока и какво количество от нея е налично; проверява дали наличното количество е достатъчно, за да се изпълни заявката за закупуване чрез извикването на предиката

available(BrRequest,BrAvailable), който сравнява заявеното и наличното количество от търсената стока. Ако се окаже, че има достатъчно количество от заявената стока, наличното и продаденото количество на стоката се редуцират; изтрива се съществуващият факт за тази стока и се записва нов факт с коригираните стойности за налично и продадено количество.

Добавянето на нова стока се реализира с клаузите на предиката new_article_quantity(Article, Br). Този предикат получава два аргумента: името и количеството на стоката. Първата клауза проверява дали е налична такава стока в базата от факти. Ако такава стока има, изтрива се намереният факт за количеството на стоката, увеличава се количеството с новодоставеното, добавя се нов факт за тази стока, като се запазва нейния номер и продаденото количество, и накрая се забранява изпълнението на втората клауза. Ако такава стока не съществува в базата от факти, унифицира се втората клауза, която добавя новата стока и нейното количество в базата от факти, като номерира новата стока с първия намерен свободен номер.

За да се намери броят на артикулите в базата от факти и съответно първият свободен номер за новата стока, се използват клаузите на предиката free_number(N,L). При неговото извикване се задава стойност 1 за първия аргумент N, а номерът на първия свободен номер за стока се получава като стойност на втория аргумент му L.

```
new_article_quantity(Article, Br) :-
        article(NArticle,Article,X,S),
        retract(article(_,Article,X,_)),
        Br1 is X + Br,
        assertz(article(NArticle,Article,Br1,S)),!.

new_article_quantity(Article, Br) :-
        free_number(1,LastN),
        assertz(article(LastN,Article,Br,0)).
```

Програма.

```
article(1,"топки",20,0).
article(2,"хилки",30,0).
article(3,"ски",50,0).
article(4,"раници",44,0).
article(5,"кънки",23,0).
article(6,"шапки",33,0).
```

```
available(BrRequest,BrAvailable):-
        BrRequest<BrAvailable,
        write("Заявката е приета!"),
        write("Благодарим ви, че пазарувахте при нас!"),!.

available(BrRequest,BrAvailable) :-
        BrRequest > BrAvailable,
        BrAvailable > 0,
        write("Останали са само ", BrAvailable, "броя."), !, fail.

available(_,_):-
       write("Съжаляваме, но стоката е изчерпана. Очакваме ново
       зареждане."),
       fail.

selling_request(Article,BrRequest):-
        article(NArticle,Article,BrAvailable,BrSold),
        available(BrRequest,BrAvailable),
        BrSold1 is BrSold + BrRequest,
        BrAvailable1 is BrAvailable - BrRequest,
        retract(article(NArticle,Article,_,_)),
        assertz(article(NArticle,Article,BrAvailable1,BrSold1)),!.

new_article_quantity(Article, Br) :-
        article(NArticle,Article,X,S),
        retract(article(_,Article,X,_)),
        Br1 is X + Br,
        assertz(article(NArticle,Article,Br1,S)),!.

new_article_quantity(Article, Br) :-
        free_number(1,LastN),
        assertz(article(LastN,Article,Br,0)).

free_number(N,L):- article(N,_,_,_), N1 is N+1, free_number(N1,L),!.
free_number(Y,Y):- write("Free number is ", Y).

all_articles :-
        article(_,X,Y,Z),not(Y=0),write("Наличните ",X ," са ",Y),nl,fail.

sold_articles :-
        article(_,X,Y,Z),Y=0,write(X, "те са изчерпани."),nl,fail.
```

Решенията за следните цели са:

all_articles
Наличните топки са 20
Наличните хилки са 30
Наличните ски са 50
Наличните раници са 44
Наличните кънки са 23
Наличните шапки са 33
No solutions

selling_request("хилки",10)
Заявката е приета!Благодарим ви, че пазарувахте при нас!True
1 Solution

selling_request("раници",22)
Заявката е приета!Благодарим ви, че пазарувахте при нас!True
1 Solution

best_seller
Free number is 7 Най-търсени са раниците.
От тях сме продали : 22 броя.
True
1 Solution

new_article_quantity("маратонки",8)
Free number is 7True
1 Solution

all_articles
Наличните топки са 20
Наличните ски са 50
Наличните кънки са 23
Наличните шапки са 33
Наличните хилки са 20
Наличните раници са 22
Наличните маратонки са 8
No solutions

free_number(1,L)
Free number is 8 L= 8
1 Solution

Задание

1. Разгледайте особеностите на правилата за добавяне на нова стока, за заявка за закупуване, за определяне на наличността на стоките, за намиране на пореден номер на стоката.

2. При тази реализация на предиката new_article_quantity(Article, Br) може ли една и съща стока да получи два различни номера в базата от факти?

3. Открийте предназначението на правилата, записани с предикатите all_articles и sold_articles. Обърнете внимание, че те завършват с No Solution след намирането на всички решения. Поправете това.

4. Реализирайте правила за намиране на най-продаваната стока.

Забележка. Съставете списък с продадените количества на стоките в базата от факти, т.е. съставете списък от последния аргумент на предиката article(_,_,_, soldnumber). Намерете броя на стоките.

Използвайте предикатът append, за да добавяте следващ елемент в списък. Формирането на списъка ще започне с добавянето на артикула с предпоследен номер към артикула с последен номер и ще завърши, когато се достигне до артикул с номер нула (такъв няма), т.е. редът е от последния към първия номер от артикули.

Предикатът max_selling(list_of_numbers, max) намира най-голямото число (втора позиция) от записаните в списъка (в първа позиция). Първоначално се допуска, че максималната стойност е 0. Ако стойността на главата на списъка е по-голяма от текущата максимална стойност, се извиква отново предикатът max_selling с намерената по-голяма стойност, която става текуща. Рекурсията завършва при достигане до празен списък.

Решение:

```
best_seller :-  free_number(1,FreeNumber),
                LastNumber is FreeNumber-1,
                article(LastNumber,X,Y,Z),
                BeforLastNumber is LastNumber-1,
                make_list([Z],BeforLastNumber).

make_list(L,0):-  max_selling(L,0),!.
make_list(L,M):-  article(M,X,Y,Z),
                  append([Z],L,L1),
                  M1 is M-1,
                  make_list(L1,M1).

max_selling([X|T],Max) :-  Max<X, Max1 is X, max_selling(T,Max1).
```

```
max_selling([_|T],Max)  :-  max_selling(T,Max).
max_selling([],Max)     :-  article(N,Article,Available,Max),
            write("Най-търсени са ",Article, "те."),nl,
            write("От тях сме продали : ",Max, " броя. "),nl,!.

append([],K,K).
append([X|K1],K2,[X|K3]):- append(K1,K2,K3).
```

5. Реализирайте правило can_buy(X,Y), което гласи: X може да си купи Y, ако X е човек, Y е стока, X харесва спорта P, а за спорта P е необходима стоката Y, като добавите следните факти:

```
person("Таня").
person("Павлина").
like("Таня","Баскетбол").
like("Павлина","Туризъм").
use("Баскетбол","топки").
use("Туризъм","раници").
use("Пързаляне","ски").
```

Решение:

```
can_buy(X,Y):-person(X),article(_,Y,_,_),like(X,P),use(P,Y).
```

6. Реализирайте програма, която проверява лексиката ви на английски език. Потребителят да отговаря на всяка зададена от програмата дума; да преброява верните му отговори и да оценява знанията му. Реализирайте възможност за динамично добавяне на нови думи.

Забележка. Думите са записани като факти за предиката word(number,engl_word,bulg_word).

7. Реализирайте предикат за генериране на псевдослучайни числа. Използвайте този предикат за извеждане на думи от наличните в базата от факти, реализирана в предната точка.

Забележка.

```
random(StartRnd, MaxNumber, 0):-!.
random(R,N,BrNum):-   R1 is (R+7) mod N, B1 is BrNum-1,
                      write(R1), nl, random(R1,N,B1).
Goal: random( 35, 20, 25 )
```