**Проект по Изкуствен интелект,**

**специалност: „Информатика“, редовно обучение**

**Александър Владимиров Каранейчев, 2001261008,** [**stu2001261008@uni-plovdiv.bg**](mailto:stu2001261008@uni-plovdiv.bg)

*За точки от (5-10)*

*Родословни дървета:*

**Древногръцка митология:**

% Факти

parent(zeus, ares).

parent(zeus, athena).

parent(zeus, apollo).

parent(zeus, artemis).

parent(zeus, aphrodite).

parent(hera, ares).

parent(hera, athena).

parent(hera, apollo).

parent(hera, artemis).

parent(hera, aphrodite).

parent(cronus, zeus).

parent(cronus, hades).

parent(cronus, poseidon).

parent(rhea, zeus).

parent(rhea, hades).

parent(rhea, poseidon).

parent(gaia, uranus).

parent(gaia, cronus).

parent(uranus, gaia).

parent(uranus, pontus).

% Правила

ancestor(X, Y) :- parent(X, Y).

ancestor(X, Y) :- parent(X, Z), ancestor(Z, Y).

% Примери на заявки

?- ancestor(zeus, ares).

?- ancestor(gaia, zeus).

?- ancestor(rhea, aphrodite).

*В този пример използваме факти, които представляват родителските връзки между боговете. Правилото* ***"ancestor"*** *ни позволява да определим дали един бог е предшественик на друг бог чрез верига от родителски връзки. С тези факти и правила можем да изпълняваш заявки като* ***ancestor(X, Y)*** *, за да откриеш предшествениците на даден бог. Например, заявката* ***ancestor(zeus, ares)*** *ще върне* ***"Yes"****, защото* ***"ares"*** *е наследник на* ***"zeus"****. Заявката* ***ancestor(gaia, zeus)*** *ще върне* ***"No"****, защото* ***"gaia"*** *не е предшественик на* ***"zeus"****.*

**Библиотека:**

% Факти

author('George Orwell', '1984').

author('George Orwell', 'Animal Farm').

author('J.R.R. Tolkien', 'The Lord of the Rings').

author('J.R.R. Tolkien', 'The Hobbit').

author('Jane Austen', 'Pride and Prejudice').

author('Jane Austen', 'Sense and Sensibility').

author('William Shakespeare', 'Romeo and Juliet').

author('William Shakespeare', 'Hamlet').

% Правила

ancestor(X, Y) :- author(X, Y).

ancestor(X, Y) :- author(X, Z), ancestor(Z, Y).

% Примери на заявки

?- ancestor('George Orwell', '1984').

?- ancestor('William Shakespeare', Book).

*Отново в този пример използваме факти както в предния с връзки между авторите и техните книги. Отново ползваме същото правило* ***"ancestor"****. Дадени тези факти и правила можем да правим заявки като* ***ancestor('George Orwell', '1984')****, за да проверим дали* ***"George Orwell"*** *е автор на* ***"1984"****. Заявката* ***ancestor('William Shakespeare', Book)*** *ще върне всички книги, които са написани от* ***"William Shakespeare"***.

*За точки от (10-15)*

*I. Иван, Петър и Васил събирали шишарки в гората. Едно от момчетата носело*

*кошница, другото – кофа, а третото носело найлонова чанта. Петър не носел кошница,*

*нито чанта. Иван също не носел кошница. В какво е събирал шишарките всеки от тях?*

% Факти

item(ivan, \_).

item(peter, \_).

item(vasil, \_).

% Правила

:- use\_module(library(lists)).

valid\_items(Items) :-

Items = [\_, \_, \_],

permutation([basket, bucket, bag], Items).

% Изключения

:- not(item(peter, basket)).

:- not(item(peter, bag)).

:- not(item(ivan, basket)).

% Заявки

?- item(peter, Item1), item(ivan, Item2), item(vasil, Item3), valid\_items([Item1, Item2, Item3]).

*Тук използваме факти, които определят имената на момчетата* ***(ivan, peter, vasil)*** *и неизвестното им предмет* ***(\_)****. Правилото* ***valid\_items*** *проверява всички възможни комбинации от предмети* ***(basket, bucket, bag)*** *(пермутация).*

*Използваме изключения, които гарантират, че Петър* ***не носи*** *кошница и найлонова чанта, а Иван* ***не носи*** *кошница.*

*В последната заявка използваме* ***item(peter, Item1)****,* ***item(ivan, Item2)*** *и* ***item(vasil, Item3)****, за да определим предметите, които носят Петър, Иван и Васил.* ***valid\_items([Item1, Item2, Item3])*** *проверява дали комбинацията от предмети е валидна.*