Klokun

Опишіть вашу предметну область словами

Предметна область «Шрифти».

Об’єкти: шрифт, тип шрифта, словолитня, ціна.

Запити:

1. Знайти шрифти із засічками.
2. Знайти шрифти заданої словолитні.
3. Знайти шрифти, дорожчі за задану ціну.
4. Знайти дорогі моноширинні шрифти.

Напишіть мовою Пролог програму, яка описує вказані об’єкти та реалізує запити

```
domains
    name, type, foundry = symbol
    price = real

predicates
    typeface(name, type, foundry, price).
    find_serifs(name).
    find_by_foundry(name, foundry).
    find_more_expensive_than(name, price).
    find_expensive_monos(name).

clauses
    typeface(tiempos,serif,klimfonts,650).
    typeface(untitledsans,sansserif,klimfonts,300).
    typeface(untitledserif,serif,klimfonts,225).
    typeface(pitch,mono,klimfonts,300).
    typeface(lava,serif,typotheque,475).

    find_serifs(Name) :-
        typeface(Name, Type, _, _),
```

```
        Type = serif.

    find_by_foundry(Name, Foundry) :-
        typeface(Name, _, F, _),
        F = Foundry.

    find_more_expensive_than(Name, Lowerbound) :-
        typeface(Name, _, _, P),
        P > Lowerbound.

    find_expensive_monos(Name) :-
        typeface(Name, Type, _, Price),
        Type = mono,
        Price > 200.
```

Розгляньте розділи Пролог-програми: domains, predicates, clauses, goal. Для кожного з розділів поясніть словами його призначення та наведіть відповідний приклад зі своєї програми.

Розділ `domains`

У розділі `domains` описуються домени, тобто типи даних, визначені програмістом. У заданій програмі розділ `domains` виглядає так:

```
domains
    name, type, foundry = symbol % назва, тип, словолитня
    price = real
```

В ньому сказано, что типи даних `name`, `type`, `foundry` еквівалентні вбудованому типу даних `symbol`, а тип даних `price` еквівалентний вбудованому типу даних `real`.

Розділ `predicates`

У розділі `predicates` описуються прототипи предикатів, тобто скільки аргументів приймає предикат і якого вони типу. У заданій програмі розділ `predicates` виглядає так:

```
predicates
    typeface(name, type, foundry, price).
    find_serifs(name).
```

```
    find_by_foundry(name, foundry).
    ...
```

Тут об'явлені предикати `typeface` для об'явлення шрифта, `find_serifs` для пошуку шрифтів із засічками, `find_by_foundry` для пошуку шрифтів за назвою словолитні і так далі.

Розділ `goals`

У розділі `goals` розміщують внутрішню мету програми, тобто мету, якої Пролог спробує досягнути одразу ж після запуску програми.

У створеній програмі відсутній розділ `goals`, тобто при запуску користувачу запропонують ввести власну, зовнішню мету. Тим не менш, можна створити і внутрішню. Наприклад, щоб знайти перший шрифт із засічками:

```
goals
    find_serifs(X).
```

Розділ `clauses`

У розділі `clauses` описуються факти та правила. У розробленій програмі розділ `clauses` виглядає так:

```
clauses
    typeface(tiempos,serif,klimfonts,650).
    typeface(untitledsans,sansserif,klimfonts,300).
    ...
```

В приведеному прикладі описані шрифти `tiempos` і `untitledsans` з відповідними властивостями: типом, словолитньою, що його створила, та його ціною.

Поясніть словами мету виконання Пролог-програми. Наведіть пояснення мети виконання програми за запитами у своєму прикладі програми.

Мета Пролог-програми — це формулювання задачі, яку програма має вирішити, у формі запиту. Після формулювання цього запиту Пролог намагатиметься знайти усі записи (тобто факти або правила), які задовольняють умовам цього запиту.

Розглянемо мету виконання програми на прикладі зовнішньої мети `find_expensive_monos(X)`. Нагадаємо, що предикат виглядає так:

```
find_expensive_monos(Name) :-
    typeface(Name, Type, _, Price),
    Type = mono,
    Price > 200.
```

З такою метою Пролог шукатиме шрифти з типом `mono` та значенням `Price` більше 200.

Поясніть, що таке послідовність цілей при виконанні Пролог-програми і яким чином вона змінюється. Наведіть приклад послідовності цілей та її зміни за своєю програмою

Послідовність цілей при виконанні Пролог-програми — це послідовність, в якій Пролог перетворює запит в ході пошуку розв'язкуу. Вона змінюється в залежності від того, як визначені запити, предикати (чи рекурсивні вони, чи викликають інші предикати) тощо.

Наприклад, ціль `find_expensive_monos(X)` буде замінена так:

```
find_expensive_monos(X)
```

↓

```
typeface(Name, Type, _, Price), Type = mono, Price > 200.
```

Якщо визначити програму так:

```
proportional(Name) :-
    serif(Name) ; sans(Name).

serif(untitledserif).
serif(lava).
mono(pitch).
sans(untitledsans).

expensive(untitledserif).
expensive(lava).
```

То для мети `proportional(Name), expensive(Name)` послідовність цілей буде такою:

1. `proportional(Name), expensive(Name)`.
2. `serif(Name), expensive(Name)` (залишиться `expensive(untitledserif)`).
3. `sans(Name), expensive(Name)` (залишиться `expensive(lava)`).

Поясніть словами, що таке дерево пошуку (чи дерево виконання) для Пролог-програми. Наведіть приклад дерева пошуку за своєю програмою

Дерево пошуку — це представлення того, як Пролог-програма шукає розв'язок поставленої задачі, у вигляді абстрактної структури даних — дерева.

Дерево пошуку для розробленої програми на прикладі мети `find_expensive_monos(X)`:

```
find_serifs()
|- typeface("tiempos", serif, _, _)
|  |- serif = serif
|     |- Yes
|- typeface("untitledsans", sansserif, _, _)
|  |- sansserif = serif
|     |- No
|- typeface("untitledserif", serif, _, _)
|  |- serif = serif
|     |- Yes
|- typeface(pitch, mono, _, _)
|  |- mono = serif
|     |- No
|- typeface(lava, serif, _, _)
   |- serif = serif
      |- Yes
```

Поясніть поняття «зіставлення». Наведіть приклад зіставлення за своєю програмою

Зіставлення — це процес, на вхід якого подаються два терми, і результатом якого є висновок, чи відповідають ці терми один одному. В розробленій програмі зіставлення можна зустріти у предикаті `find_serifs/1`:

```
find_serifs(Name) :-
    typeface(Name, Type, _, _),
    Type = serif. % Зіставлення
```

Поясніть поняття «пов'язування змінної», «звільнення змінної». Наведіть приклади пов'язування і звільнення змінної за своєю програмою

Пов'язування змінної — це надання змінній будь-якого значення, відмінного від неконкретизованої змінної. Звільнення змінної — це повернення змінної до неконкретизованого стану.

У розробленій програмі конкретизація та звільнення змінної відбувається на етапі пошуку розв'язку. Наприклад, у предикаті `find_serifs/1` на етапі `typeface(tiempos, serif, ...)` змінна Type конкретизується зі значенням `serif`, і після відкату звільнюється — повертаєтья до неконкретизованого стану.

```
typeface(tiempos,serif,klimfonts,650).
...
find_serifs(Name) :-
    typeface(Name, Type, _, _),
    Type = serif. % "serif" = "serif"
```

Поясніть поняття «рекурсія» та його застосування у логічній програмі. Наведіть приклад рекурсії у своїй предметній області. Поясніть словами, які дві гілки є у вашому прикладі рекурсії та яку роль виконує кожна з них

Рекурсія — це звернення об'єкта до себе самого. В контексті логічного програмування рекурсія часто використовується у визначенні предикатів.

Класичні шрифти часто перероблюють — відродждують, тому напишемо рекурсивний предикат, який перевіряє, чи є даний шрифт відродженням.

Helvetica → Neue Helvetica → Helvetica World → Helvetica W1G → Neue Haas Grotesk

```
revival(X,Y) :- updates(X,Y).

revival(X,Y) :- updates(X,Z),
                revival(Z,Y).

updates(neue_helvetica, helvetica).
updates(helvetica_world, neue_helvetica).
updates(helvetica_w1g, helvetica_world).
updates(neue_haas_grotesk, helvetica_w1g).
```

В цьому прикладі є дві гілки: рекурсивна, яка відповідає за поглиблення пошуку:

```
revival(X, Y) :- updates(X, Z), revival(Z, Y)
```

А також термінальна, яка відповідає за завершення рекурсії.

```
revival(X, Y) :- updates(X, Y)
```