|  |  |  |  |
| --- | --- | --- | --- |
|  | ***Лекция 1*** |  |  |
|  | ***Общий обзор языка Пролог.*** |  |  |

## Cодержание

[1.1 Пример программы: родственные отношения.](#Lec1.1)

[1.2 Факты.](#Lec1.2)

[1.3 Вопросы.](#Lec1.3)

[1.4 Переменные.](#Lec1.4)

[1.5 Конъюнкция целей.](#Lec1.5)

[1.6 Правила.](#Lec1.6)

[1.7 Конъюнкция в правилах.](#Lec1.7)

[1.8 Переменные в теле правила.](#Lec1.8)

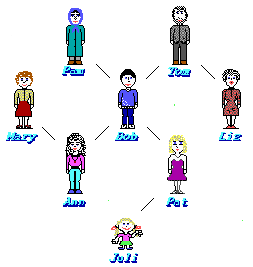
[1.9 Cтруктура пролог-программы.](#Lec1.9)

|  |  |
| --- | --- |
| ***Пролог -*** | ***это язык программирования, используемый для решения задач, в которых действуют объекты и отношения между этими объектами.*** |

*Программа на прологе состоит из предложений, которые могут быть фактами, правилами или вопросами.*

## 1.1 Пример программы: родственные отношения.

Рассмотрим **дерево родственных отношений :**



## 1.2 Факты.

Введем **отношение -родитель- (parent)** между объектами.  
**parent (tom, bob).**   
Это факт, определяющий , что **Том** является родителем **Боба**.

**parent** - имя отношения, **tom, bob** - его аргументы. Теперь можно записать программу, описывающую все дерево родственных отношений.

|  |  |  |
| --- | --- | --- |
|  | **parent (pam, bob). parent (tom, bob). parent (tom, liz). parent (bob, ann). parent (bob, pat). parent (mary, ann). parent (pat, juli).** | D:\777 Новая папка\SiteMorozov99\LECTION1\tree1.gif |

Эта программа состоит из семи **предложений (утверждений), clause(клоз).**  
Каждый клоз записан фактом в виде отношения **parent.**  
При записи фактов надо соблюдать следующие **правила**:

* Имена всех отношений и объектов с маленькой буквы.
* Сначала записывается имя отношения, затем в круглых скобках через запятую объекты.
* В конце ставится точка.

Еще пример факта:

|  |  |
| --- | --- |
|  | **like (bob, pam).** |

Совокупность фактов в прологе называют **базой даннах**.

## 1.3 Вопросы.

К составленной базе данных можно задать **вопросы.** .

Вопрос в обычном прологе начинается с **?-**   
Вопрос записывается также, как и факт.  
Например:

|  |  |  |
| --- | --- | --- |
|  | **? - parent (bob, pat).**  **yes** | D:\777 Новая папка\SiteMorozov99\LECTION1\comput.gif |

Когда пролог получает вопрос, он пытается сопоставить его с базой данных. Такой факт находится, ответ: **да (yes).**

На вопрос

|  |  |  |
| --- | --- | --- |
|  | **?-parent (bob,mary).**  **no** |  |

Ответ будет **нет (no)**, так как такого факта в базе данных нет.

## 1.4 Переменные.

Можно задать вопрос и узнать кто родитель **liz**:

|  |  |  |
| --- | --- | --- |
|  | **?-parent (X, liz).**  **X= tom** | D:\777 Новая папка\Презентация Пролога\LECTION1\TOM_LIZ.GIF |

Здесь **X** - **переменная**. Ее величина неизвестна и она может принимать значения. В данном случае ее значением будет объект, для которого это утверждение истинно.

Вопрос :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **?-parent (X, bob).**  **X=tom  X=pam** |  | D:\777 Новая папка\Презентация Пролога\LECTION1\BOBXX_2.GIF |

Можно задать вопрос, кто является чьим родителем.  
Или найти такие **X** и **Y**, что **X** является родителем **Y**.

|  |  |  |
| --- | --- | --- |
|  | **?-parent (X, Y).**  **X= pam  Y= bob  Х= tom  Y= bob  и т.д.** | tree |

## 1.5 Конъюнкция целей.

Можно задать более общий вопрос: Кто является родителем родителя **juli**. Так как нет отношения **grandparent**, то можно разбить на **два вопроса:**

1. кто родитель **juli**. Предположим- **Y**.
2. кто родитель **Y**. Предположим- **X**.

Тогда составной вопрос:

|  |  |  |
| --- | --- | --- |
|  | **?-parent (Y, juli), parent (X, Y).  X=bob  Y=pat** | D:\777 Новая папка\Презентация Пролога\LECTION1\JULI.GIF |

При поиске решения сначала находится **Y** , а затем по второму условию **Х**.  
**Вопрос:** Кто внуки тома?:

|  |  |  |
| --- | --- | --- |
|  | **?-parent (tom, Y), parent (Y, X).**  **Y=bob  X=ann  Y=bob  X=pat** | D:\777 Новая папка\Презентация Пролога\LECTION1\TOM_BOB.GIF |

И наконец, есть ли у **ann** и **pat** общий родитель?

|  |  |  |
| --- | --- | --- |
|  | **?-parent (Y, ann), parent(Y, pat).**  **Y=bob** |  |

## 1.6 Правила.

Введем отношение peбенок **child**, обратное к **parent** "родитель".   
Можно было бы определить аналогично:

|  |  |
| --- | --- |
|  | **child (liz, tom).** |

Но можно использовать, что отношение **child** обратно к **parent** и записать в виде утверждения- **правила** :

|  |  |
| --- | --- |
|  | **child(Y, X):-parent (X, Y).** |

Правило читается так:

**Для всех X и Y   
Y -child X, если  
X -parent Y.**

Правило отличается от факта тем, что факт всегда истина, а правило описывает утверждение, которое будет истинной, если выполнено некоторое условие. Поэтому в правиле выделяют: заключение условие

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **child(Y, X) :- parent (X, Y).**   |  |  |  | | --- | --- | --- | | **голова**    **head** |  | **тело**    **body** | |

Если условие **parent (X, Y)**. выполняется, то логическим следствием из него будет утверждение **child(Y, X)**.

**Как правило используется прологом :**

Зададим вопрос

|  |  |
| --- | --- |
| **?-child(liz, tom).** | D:\777 Новая папка\Презентация Пролога\LECTION1\TOM_LIZ.GIF |

В программе нет данных о **child**.  
Но есть правило, которое верно для всех **X Y**, в том числе для **liz** и **tom**.  
Мы должны применить правило для этих значений.  
Для этого надо подставить в правило вместо **X**- **tom**, a вместо **Y** - **liz**.

Говорят, что переменные будут связаны, а операция будет называться подстановкой.   
Получаем конкретный случай для правила

**child(liz, tom):-parent (tom, liz).**

Условная часть приняла вид   
**parent (tom, liz).**   
Теперь надо выяснить выполняется ли это условие. Исходная цель **child(liz,tom)** заменяется подцелью **parent (tom, liz).**, которая выполняется, поэтому пролог ответит **"yes".**

## 1.7 Конъюнкция в правилах.

Добавим еще одно отношение в базу данных, унарное, определяющее пол.

|  |  |
| --- | --- |
|  | **male(tom). male(bob). male(jim). female(liz). female(pam). female(pat). female(ann).** |

Теперь определим отношение **mother**. Оно описывается следующим образом:

**Для всех X Y   
X -mother Y, if   
X- parent Y и   
X -female.**

Таким образом правило будет

|  |  |  |  |
| --- | --- | --- | --- |
|  | **mother(X, Y):-parent(X, Y), female(X).** | | |
|  | |  |

Запятая между двумя условиями означает конъюнкцию целей. Это означает, что два условия должны быть выполнены одновремено.   
Как система ответит на вопрос?

|  |  |  |
| --- | --- | --- |
|  | **?-mother (pam, bob).**  **yes** | D:\777 Новая папка\Презентация Пролога\LECTION1\MOTHER.GIF |

Находится правило **mother**, производится подстановка

**X=pam   
Y=bob**

Получаем правило

**mother(pam, bob):-   
parent(pam, bob),   
female(pam).**

Сначала удовлетворяются **parent** , а затем **female**

Пролог отвечает: **yes**   
Вопрос :

|  |  |
| --- | --- |
| **?-mother (X, bob).**  **X=pam** |  |

## 1.8 Переменные в теле правила.

Определим отношение **sister**

**Для любых X и Y   
X sister Y, if   
у X и Y есть общий родитель,   
и X female**

Запишем правило на прологе

|  |  |  |  |
| --- | --- | --- | --- |
|  | **sister (X, Y):- parent(Z,X),**   |  |  | | --- | --- | |  | **parent(Z,Y),  female(X).** | |

Здесь **Z**-общий родитель. **Z**-некоторый, любой.   
Можно спросить

|  |  |
| --- | --- |
| **?-sister(ann, pat).**  **yes**  **?-sister(pat, pat).**  **yes** | D:\777 Новая папка\Презентация Пролога\LECTION1\SISTER.GIF |

Ответ будет **"yes"**. Так как мы не потребовали, чтобы **X** и **Y** были разные.

Добавим отношение **different (X, Y)**, которое указывает ,что **X** и **Y** разные.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **sister (X, Y):- parent(Z,X),**   |  |  | | --- | --- | |  | **parent(Z,X),  female(X),  different (X, Y).** | |

## 1.9 Cтруктура пролог-программы.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **предложение** | **-** | **факт,** | **правило,** | **вопрос** |
| **clause** |  | **fact,** | **rule,** | **goal** |
| **состав** |  | **head** | **head** |  |
|  |  | **.** | **..** | **?** |
|  |  |  | **|** | **|** |
|  |  |  | **body** | **body** |