Курс лекции по дисциплине YaPZ7302 Языки представления знаний

Лекция 4. Функции

Цель: Изучение функций

План:

1. Конструктор *deffunction*
2. Пример. Использование группового параметра
3. Пример . Использование рекурсии для вычисления факториала
4. Пример. Создание взаимно рекурсивных функций

Конструктор deffunction позволяет пользователю создавать новые функции непосредственно в среде CLIPS. Способ вызова функций, определенных пользователем, эквивалентен способу вызова внутренних функций CLIPS. Вызов функции осуществляется по имени, заданному пользователю. За именем функции следует список необходимых аргументов, отделенный одним или большим числом пробелов. Вызов функции вместе со списком аргументов должен заключаться в скобки. Последовательность действий определенной с помощью конструктора deffunction функции исполняется интерпретатором CLIPS (в отличие от функций, созданных на других языках программирования, которые должны иметь уже готовый исполнимый код).

Синтаксис конструктора deffunction включает в себя 5 элементов:

- имя функции;
- необязательные комментарии;
- список из нуля или более параметров;
- необязательный символ групповых параметров для указания того, что функция может иметь переменное число аргументов;
- последовательность действий или выражений, которые будут выполнены (вычислены) по порядку в момент вызова функции.

Определение. Синтаксис конструктора deffunction

(deffunction <имя-функции>

[<комментарии>]

<обязательные-параметры>

[<групповой-параметр>]

<действия>)

<обязательные-параметры> ::= <выражение-простое-поле>

<групповой-параметр> ::= <выражение-составное-поле>

Функция, создаваемая с помощью конструктора deffunction, должна иметь уникальное имя, не совпадающее с именами других внешних и внутренних функций. Функция, созданная с помощью deffunction, не может быть перегружена. Конструктор deffunction должен быть объявлен до первого использования создаваемой им функции. Исключения составляют только рекурсивные функции.

В зависимости от того, задан ли групповой параметр, функция, созданная конструктором, может принимать точное число параметров или число параметров не меньшее, чем некоторое заданное. Обязательные параметры определяют минимальное число аргументов, которое должно быть передано функции при ее вызове. В действиях функции можно ссылаться на каждый из этих параметров как на обычные переменные, содержащие простые значения. Если был задан групповой параметр, то функция может принимать любое количество аргументов большее или равное минимальному числу. Если групповой параметр не задан, то функция может принимать число аргументов точно равное числу обязательных параметров. Все аргументы функции, которые не соответствуют обязательным параметрам, группируются в одно значение составного поля. Ссылаться на это значение можно, используя символ группового параметра. Для работы с групповым параметром могут использоваться стандартные функции CLIPS, предназначенные для работы с составными полями , такие как length и nth. Определение функции может содержать только один групповой параметр.

Приведенный пример демонстрирует описанные выше возможности работы с групповыми параметрами.

Пример. Использование группового параметра

```
(deffunction   print-args (?a ?b $?c)
(printout t ?a “ “ ?b “ and “ (length ?c) “ extras: “ ?c
crlf))
(print-args 1 2)
(print-args a b c d)
(print-args a)
```

В данном примере с помощью конструктора deffunction определяется функция print-args, которая принимает два обязательных параметра: ?a и ?b, и имеет групповой параметр $?c. Функция выводит на экран свои обязательные параметры, а также число полей в составном параметре и его содержимое. Результат выполнения данного примера приведен на рис. 30.

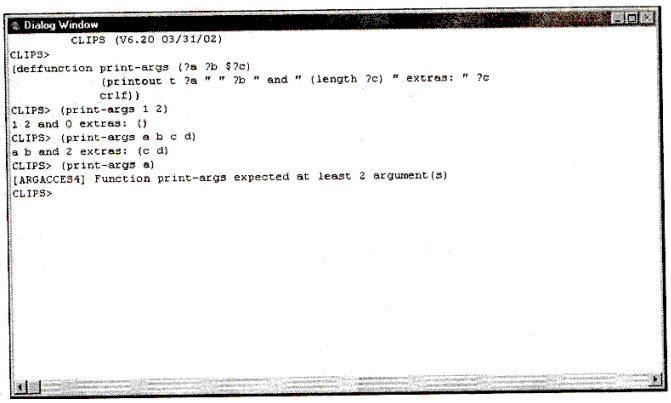

Рис. 30. Результат работы функции print-args

Обратите внимание, что вызов функции с числом параметров, меньшим минимального, приводит к сообщению об ошибке.

При вызове функции интерпретатор CLIPS последовательно выполняет действия в порядке, заданном конструктором. Функция возвращает значение, равное значению, которое вернуло последнее действие или вычисленное выражение. Если последнее действие не вернуло никакого результата, то выполняемая функция также не вернет результата (как в приведенном выше примере). Если функция не выполняет никаких действий, то возвращенное значение равно FALSE. В случае возникновения ошибки при выполнении очередного действия выполнение функции будет прервано и возвращенным значением также будет FALSE.

Функции могут быть само- и взаимно рекурсивными. Саморекурсивная функция просто вызывает сама себя из списка своих собственных действий. В качестве примера можно привести функцию, вычисляющую факториал.

Пример . Использование рекурсии для вычисления факториала

```
(deffunction factorial (?a)
(if (or (not (integerp ?a)) (< ?a 0)) then
(printout t "Factorial Error!" crlf)
  else
(if (= ?a 0) then
    1
else
(  * ?a   (factorial   (-   ?a  1))))))
```

Взаимная рекурсия между двумя функциями требует предварительного объявления одной из этих функций. Для предварительного объявления функции в CLIPS используется конструктор deffunction с пустым списком действий. В следующем примере функция foo предварительно объявлена и таким образом может быть вызвана из функции bar. Окончательная реализация функции foo выполнена конструктором после объявления функции bar.

Пример. Создание взаимно рекурсивных функций

```
(deffunction foo ())
(deffunction bar ()
(foo))
(deffunction foo ()
(bar) )
```

Внимательно следите за рекурсивными вызовами функций, слишком большой уровень рекурсии может привести к переполнению стека памяти. Например, приведенный выше пример с функциями bar и foo приводит к результату, представленному на рис. 31, и аварийному завершению CLIPS.

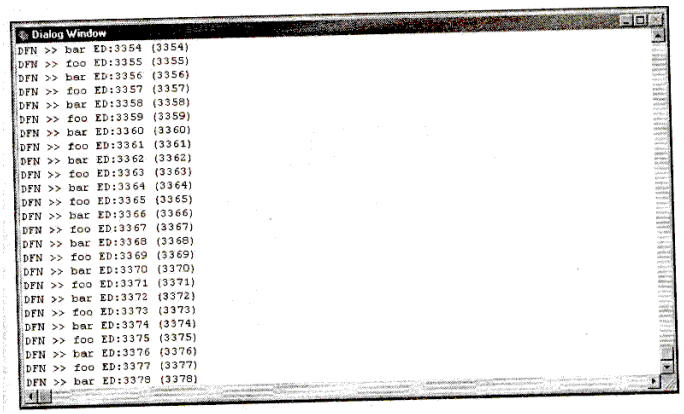

Рис. 31. Переполнение стека

Обратите внимание, что на рис. 31 в главном окне CLIPS выводится информация о запуске каждой функции. Для установки этого режима воспользуйтесь диалоговым окном **Watch Options.** Для этого откройте диалоговое окно, выбрав пункт **Watch** из меню **Execution,** и установите флажок **Deffunctions,** как показано на рис. 32. Этот режим также позволяет просматривать аргументы, которые использовались при каждом конкретном вызове функции.

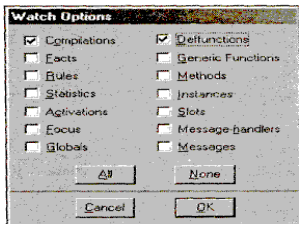

Рис. 32. Установка режима просмотра вызова функций

Так же как и для правил, предопределенных фактов, глобальных переменных Windows-версия CLIPS предоставляет специальный инструмент для работы с функциями — **Deffunction Manager** (Менеджер функций). Для запуска этого инструмента воспользуйтесь пунктом **Deffunction Manager** из меню **Browse. В** случае если в CLIPS не определена ни одна функция, данный пункт меню недоступен. Менеджер функций, отображающий функции, созданные нами в этой главе, изображен на рис. 34.

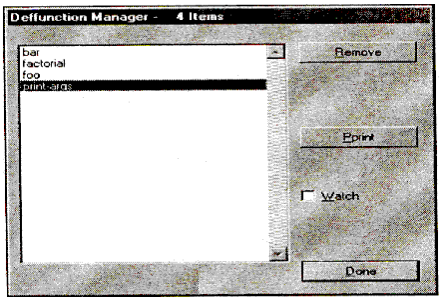

Рис. 34. Окно менеджера функций

Общее количество внешних функций отображается в заголовке окна менеджера — **Deffunction Manager — 4 Items.** С его помощью можно распечатать определение функции, удалить ее, а также установить режим просмотра вызова для отдельно выбранной функции.

Контрольные вопросы:

1. Синтаксис конструктора *deffunction.*
2. Использование группового параметра.
3. Как установить режим просмотра вызова функций

Литература:

1. А. П. Частиков Т. А. Гаврилова Д. Л.Белов. РАЗРАБОТКА ЭКСПЕРТНЫХ СИСТЕМ. СРЕДА CLIPS. СПб: БХВ-Петербург, 2003

Ключевые слова:

Функции, *deffunction*, , групповой параметр, взаимно рекурсивные функции