CS603 Programming Language Organization

Lecture 24

Spring 2004

Department of Computer Science

Overview

- Questions
- Answers from last time
- Programming

Welcome to the ML Realm

- Suppose we want to represent all inhabitants of a kingdom by their class:
  - King – simply himself
  - Peer – has a degree, territory and number
    - ex.:"the 7th Earl of Carlisle"
  - Knight or Peasant – has a name

How might we represent this in μ-Scheme?

The Realm in μ-Scheme

- `"king"`
- `("Earl" "Carlisle" 7) ("Duke" "Norfolk" 9)`
- `("Knight" "Gawain") ("Knight" "Galahad")`
- `("Peasant" "Jack Cade") ("Peasant" "Wat Tyler")`
- But this doesn't work in ML!

Why?

More About ML

- Strings are supported with "abc", concatenation is caret, ^
  - example: "abc" ^ "def" yields "abcdef"
- Pattern matching can be done on datatypes
  - ```
    datatype iTree = IntNode of int * iTree * iTree
                   | Empty;
    fun sum (IntNode(val,L,R)) = val + sum L + sum R
      | sum Empty = 0;
    ```
    ```

Programs

- Write a datatype for representing a person
  - `datatype person = King`
    `| Peer of ...`
    `| Knight of ...`
    `| Peasant of ... ;`

Programs (cont.)

- Write a function `title` that behaves as follows:
  `title King` yields `"His Majesty the King"`
  `title Peer("Earl", "Carlisle",7)` yields `"The Earl of Carlisle"`
  `title (Knight "Gawain")` yields `"Sir Gawain"`
  `title (Peasant "Joel")` yields `"Joel"`
- Write a function `sirs` that returns the name of all Knights in a list of `persons`, as a list of strings:
  `sirs [King, Peasant("Joel"), Knight("Gawain")]` yields `["Gawain"]`

Solution

- ```
  datatype person = King
                  | Peer of string * string * int
                  | Knight of string
                  | Peasant of string;

  fun title King                 = "His Majesty the King"
    | title (Peer(deg,terr,_)) = "The " ^ deg ^ " of " ^ terr
    | title (Knight name)        = "Sir " ^ name
    | title (Peasant name)       = name;

  fun sirs []                    = []
    | sirs ((Knight(s) :: ps)) = s :: (sirs ps)
    | sirs (_::ps)               = sirs ps;
  ```
  ```

Resources

- "Programming in Standard ML"
  - http://www-2.cs.cmu.edu/~rwh/smlbook/offline.pdf
- Source code from above book
  - http://www-2.cs.cmu.edu/~rwh/smlbook/examples/
- Moscow ML
  - http://www.dina.dk/~setsoft/mosml.html
- Web-based impcore, μ-scheme, Moscow-ML interpreters
  - http://www.unix.eng.ua.edu/~crutcher/class/cs603/wrapomatic/