CS 360 - Introduction to ML

Mark Boady

July 18, 2013

Homework Notes

- Abstract Data Types
  - The interface for a data type. The functions defining how the data type works.
  - Examples: lists, queues, stacks
- Data Structure
  - The impementation of a data type
  - Examples: arrays and linked lists using nodes
- $(0^n1^n2^n|n \geq 1)$ means the number of 0,1,2 must be the same.
- Written answers for lab questions can be short.
  - member: The member function takes an element and a list. It returns true if the element is in the list and false otherwise. Comparisons are made against the head of recursive lists.

ML

- sml/nj is installed on tux.
  - sml
- To quit
  - Ctrl-D (end of file)
  - OS.Process.exit(OS.Process.success);
- Load Files
  - You can load files from the current directory
  - use “filename.ml”;

Math

```
- 2 + 3;
val it = 5 : int
- 5 - 7;
val it = ~2 : int
- 5 < 7;
val it = true : bool
- 5 * 1.1;
stdIn:7.1-7.8 Error: operator and operand don't agree [
    operator domain: int * int
    operand:         int * real
    in expression:                5 * 1.1
```

List Functions

- Create a list
  - [1, 2, 3, 4, 5];
- Concat
  - 1 :: [2,3,4,5];
- Get the first element (head)
  - hd [1,2,3,4,5];
- Get all but the first element (tail)
  - tl [1,2,3,4,5];
- Check if a List is null
  - null [];
- We don't need most of these because of pattern matching.
- Assign the head to h and tail to t
  - val h::t = [1,2,3,4];

Define Functions

- Define Variables
  - val L = [1,2,3,4];
- Define functions
  - fun sqr x = x * x;
- Factorial Function
  - fun fact n = if n = 0 then 1 else n * fact(n-1);
  - fun fact2 0 = 1 | fact2 n = n * fact2(n-1);

```
- fact 10;
val it = 3628800 : int
- fact2 10;
val it = 3628800 : int
```

Defining Functions with Patterns

```
fun append nil L = L
    | append (h::t) L = h::(append t L);
fun append x y =
    case x of
        [] => y |
        (h::t) => h :: append t y;
append [4] [1,2,3];
val it = [4,1,2,3] : int list
- append;
val it = fn : 'a list -> 'a list -> 'a list
```

Lab 2

- Experiment with Scheme Code
- Test and Document 2 functions in ML
- Extra Credit: Test and Document additional ML function
- Complete Entire Assignment before showing code trace
- You can just show your executions, you do not have to redo for me.