I/O in ML

10-22-2004

Opening Discussion

- What are patterns in ML? When do we use them? How do they help us program?
- Let's look at how we could write a function that returns the length of a list.

The print Function

- The print function prints a string to the screen.
- Non-strings can be printed with functions that convert another type to a string.
  - str: char -> string
  - Real.toString
  - Int.toString
  - Bool.toString
- Returns *unit* which is denoted as ().

Structures and Opening Them

- In the last slide we saw a '.' syntax in ML. What is on the left of the dot is called a “structure”. If you are using a structure often, you can open it with an expression like “open Int;”. This adds everything from Int into the global namespace.
- This is not generally a good practice so only do it when you really need to.

Statement Lists

- In Scheme when we were printing we often used begin to allow us to put multiple statements in sequence.
- In ML we use a statement list. This is simply a list of semicolon separated expressions inside of parentheses.
- (<exp1>; <exp2>; ... ; <expN>)
- Like begin, it takes the value of the last expression.
- Not the same as let. Let only allows declarations.

Reading from File

- To read from a file we need an instream that we get with TextIO.openIn(string_name).
- The predicate TextIO.endOfStream(<file>) tells us if we have reached the end of a file.
- TextIO.inputN(<file>,n) reads the next n characters from the file and returns them as a string.
- TextIO.inputLine(<file>) returns the string from the next unread line, including the newline at the end.
- TextIO.input(<file>) reads a whole file as a string.

Reading Single Characters

- We can also read a single character with TextIO.input1(<file>). What makes this interesting is the return type: char option.
- A char option is either NONE or SOME c, where c is a char. We can match which one it is with a pattern.
- In general anything can be made an option.
- There are also function, isSome(x) and valOf(x) that can be applied to options.

Lookahead on Input and Closing

- Sometimes it is nice to look at the next character without actually reading it. That can be done with TextIO.lookahead(<file>).
- We can also inquire if there are a certain number of characters left to read with TextIO.canInput(<file>,n).
- As in all languages, we also need to close files that we open and we do so with TextIO.closeIn(<file>).
- TextIO.stdIn can be used for standard input.

Ouput to Files

- We can open files with TextIO.openOut (<name>) or TextIO.openAppend(<name>).
- When done we close the file with TextIO.closeOut(<file>).
- We write output to the file with the command TextIO.output(<file>,<string>).
- We can flush a file with TextIO.flushOut (<file>).
- TextIO.stdOut is standard output and TextIO.stdErr is the standard error output.

Minute Essay

- Write code to read in a line of text and take all the spaces out of it.
- Remember that assignment #5 is due today. I should have a description of assignment #6 up soon.