A Basic Interpreter in Common Lisp

Paolo Boschini – Marcus Ihlar

University of Uppsala

December 12, 2011

Problem

- Design a basic interpreter in Common Lisp

Challenges

- Line numbers
- Gotos
- For-loops
- No block structures
- Sub-routines

Small basic program

```
10 FOR i=1 to 10
20 GOSUB 1000
30 NEXT i
40 END
1000 PRINT i
2000 RETURN
```

Same program in parsed form

```
(defvar *example*
    '(((:line 10 (:for i (:icon 1) (:icon 10) nil))
     (:line 20 (:gosub 1000))
     (:line 30 (:next i))
     (:line 40 (:end))
     (:line 1000 (:print (:var i)))
     (:line 2000 (:return)))))
```

Design

- Evaluate one line at a time
- A line can have several statements
- A statement can have several sub-expressions
- Evaluation of a line goes through three functions
  - eval-line, evaluates all statements in a line sequentially
  - b-eval, evaluates all subexpressions in a statement
  - call, executes lisp functions that correspond to their basic equivalents

Design

- GOTO, stores a line number in a global variable. When a line is to be executed we check if there is a value in the goto variable, and execution continues from that point
- FOR, stores the current line number, from, to, step in a table
- NEXT i, checks the table entry that corresponds to the symbol i, performs a check and either sets the goto variable or continues the execution from the current line
- GOSUB, sets the goto variable to the specified line and pushes its current line number on a stack which is then read by return
- RETURN, sets the goto variable to the top of the gosub stack

Design

- Library functions and custom functions (DEF) are stored in a separate table since they are not based on line numbers
- Variables are stored in a variable table
- Arrays are stored in an array table

Questionable design decisions

- Use of global variables

Example code

```
(defun b-eval (item acc lnum)
  (cond
   ((and (null item) (not (null acc)))
    (call (reverse acc) lnum))
   ((consp (car item))
    (b-eval (cdr item)
            (cons (b-eval (car item) nil lnum) acc)
            lnum))
   (t
    (b-eval (cdr item) (cons (car item) acc) lnum))))
```

Example

Example