Tiny Lisp Interpreter

Brent Seidel
Phoenix, AZ

June 25, 2020

Contents

Chapter 1

Introduction

1.1 What is This?

This is a tiny Lisp interpreter written in Ada. It is designed to proved a language that can be embedded into other programs, including running on embedded systems without an operating system. As a result, effort has been made to remove dependencies on Ada packages that may not be available. A primary example is *Ada.Text_IO*. Another feature that may be missing is dynamic memory allocation.

1.2 Why is This?

Chapter 2

Code Examples

2.1 Lisp Code

Here is some sample Lisp code

```
;
;  Read an analog pin and print the value repeatedly.
;  Digital pin 10 is tied high to keep looping and tied low to exit the loop.
;
(defun monitor-analog (n)
  (pin-mode 10 0)
  (print "Connect digital pin 10 to high to continue looping or to gnd to exit")
  (new-line)
  (print "Connect analog pin " n " to the analog value to monitor")
  (new-line)
  (print "Press <return> to continue")
  (read-line)
  (dowhile (= (read-pin 10) (+ 0 1))
    (print "Analog value is " (read-analog n))
    (new-line))
  (print "Exiting")
  (new-line))
```

2.2 Ada Code

Here is some sample Ada code

```
   procedure first_value(e : element_type;
                         car : out element_type;
                         cdr : out element_type) is
      first : element_type;
      temp : element_type;
```

```
      s : cons_index;
   begin
      if e.kind = E_NIL then
         car := NIL_ELEM;
         cdr := NIL_ELEM;
      elsif e.kind /= E_CONS then
         car := indirect_elem(e);
         cdr := NIL_ELEM;
      else -- The only other option is E_CONS
         s := e.ps;
         first := cons_table(s).car;
         cdr :=  cons_table(s).cdr;
         if first.kind = E_NIL then
            car := NIL_ELEM;
         elsif first.kind /= E_CONS then
            car := indirect_elem(first);
         else -- The first item is a E_CONS
            temp := eval_dispatch(first.ps);
            if temp.kind = E_NIL then
               car := NIL_ELEM;
            elsif temp.kind /= E_CONS then
               car := temp;
            else
               car := cons_table(temp.ps).car;
               cdr := cons_table(temp.ps).cdr;
            end if;
         end if;
      end if;
   end;
```