```
   end;

functor Value(): VALUE =
   struct
      type 'a pair = 'a * 'a

      datatype Value = NUMBERvalue of int   |
                       BOOLvalue of bool   |
                       NILvalue   |
                       CONSvalue of Value pair

      exception Value

      val mkValueNumber = NUMBERvalue
      val mkValueBool = BOOLvalue

      val ValueNil = NILvalue
      val mkValueCons = CONSvalue

      fun unValueNumber(NUMBERvalue(i)) = i   |
          unValueNumber(_) = raise Value

      fun unValueBool(BOOLvalue(b)) = b   |
          unValueBool(_) = raise Value

      fun unValueHead(CONSvalue(c, _)) = c   |
          unValueHead(_) = raise Value

      fun unValueTail(CONSvalue(_, c)) = c   |
          unValueTail(_) = raise Value

      fun eqValue(c1, c2) = (c1 = c2)

(* Pretty-printing *)
      fun printValue(NUMBERvalue(i)) = makestring(i)   |
          printValue(BOOLvalue(true)) = "true"   |
          printValue(BOOLvalue(false)) = "false"   |
          printValue(NILvalue) = "[]"   |
          printValue(CONSvalue(cons)) = "[" ^
                 printValueList(cons) ^ "]"
          and printValueList(hd, NILvalue) = printValue(hd) |
              printValueList(hd, CONSvalue(tl)) =
                 printValue(hd) ^ ", " ^ printValueList(tl) |
```

```
            printValueList(_) = raise Value
end;
```

Appendix B: Files

The following files are available in the directory `/usr/cheops/mads/course`

- `interp1.sml` Version 1 (as included in Appendix A).
- `interp2.sml` ··· `interp4.sml` The other versions.
- `build1.sml` the structure declarations needed to build Version 1.
- `build2.sml` ··· `build4.sml` Similarly for the other versions.
- `parser.sml` The parser functor.

To build Version 3, say, you type the following (assuming you have copied the files to your directory):

```
use "interp3.sml";
use "parser.sml";
use "build3.sml";
```

Since the parser functor is completely closed, you dont have to include it more than once in every session, although you will probably want to build your system several times while you experiment with the extensions.