```
 1: #!/afs/cats.ucsc.edu/courses/cmps112-wm/usr/racket/bin/mzscheme -qr
 2: ;; $Id: objtests.scm,v 1.4 2014-10-31 17:35:08-07 - - $
 3:
 4: (define e  (exp 1))
 5: (define pi (atan 0 -1))
 6: (define i  (sqrt -1))
 7:
 8: (define the-list
 9:      `(
10:
11:    (arg0     ,(find-system-path 'run-file))
12:    (e^ipi    ,(expt e (* i pi)))
13:    (hello    world)
14:    (integer  3)
15:    (list     (car cadr caddr))
16:    (log      ,log)
17:    (null     ())
18:    (number   3.14159265358979323846264338327950288419716939937)
19:    (pair     (1 . 2))
20:    (sqrt-1   ,(sqrt -1))
21:    (string   "string")
22:    (vector   ,(make-vector 10 0))
23:
24: ))
25:
26: (define tests `(
27:
28:    (,boolean?   boolean?)
29:    (,char?      char?)
30:    (,complex?   complex?)
31:    (,integer?   integer?)
32:    (,list?      list?)
33:    (,number?    number?)
34:    (,pair?      pair?)
35:    (,path?      path?)
36:    (,procedure? procedure?)
37:    (,rational?  rational?)
38:    (,real?      real?)
39:    (,string?    string?)
40:    (,symbol?    symbol?)
41:    (,vector?    vector?)
42:
43: ))
44:
45: (define (print-prop element)
46:         (let ((key (car element))
47:               (val (cadr element)))
48:              (printf "~s => ~s:~n" key val)
49:              (for-each (lambda (pair)
50:                        (when ((car pair) val)
51:                        (printf "   ~s~n" (cadr pair))))
52:                        tests)
53:              (newline)))
54:
55: (for-each print-prop the-list)
56:
```

```
arg0 => #<path:./objtests.scm>:
   path?

e^ipi => -1.0+1.2246063538223773e-16i:
   complex?
   number?

hello => world:
   symbol?

integer => 3:
   complex?
   integer?
   number?
   rational?
   real?

list => (car cadr caddr):
   list?
   pair?

log => #<procedure:log>:
   procedure?

null => ():
   list?

number => 3.141592653589793:
   complex?
   number?
   rational?
   real?

pair => (1 . 2):
   pair?

sqrt-1 => 0+1i:
   complex?
   number?

string => "string":
   string?

vector => #(0 0 0 0 0 0 0 0 0 0):
   vector?

```