```
#!/afs/cats.ucsc.edu/courses/cmps112-wm/usr/racket/bin/mzscheme -qr
;; $Id: symbols.scm,v 1.2 2014-10-31 17:35:08-07 - - $

;;
;; NAME
;;    symbols - illustrate use of hash table for a symbol table
;;
;; DESCRIPTION
;;    Put some entries into the symbol table and then use them.
;;

;;
;; Create the symbol table and initialize it.
;;

(define *symbol-table* (make-hash))
(define (symbol-get key)
        (hash-ref *symbol-table* key))
(define (symbol-put! key value)
        (hash-set! *symbol-table* key value))

(for-each
    (lambda (pair)
            (symbol-put! (car pair) (cadr pair)))
    `(

        (log10_2 0.301029995663981195213738894724493026768189881)
        (sqrt_2  1.414213562373095048801688724209698078569671875)
        (e       2.718281828459045235360287471352662497757247093)
        (pi      3.141592653589793238462643383279502884197169399)
        (div     ,(lambda (x y) (floor (/ x y))))
        (log10   ,(lambda (x) (/ (log x) (log 10.0))))
        (mod     ,(lambda (x y) (- x (* (div x y) y))))
        (quot    ,(lambda (x y) (truncate (/ x y))))
        (rem     ,(lambda (x y) (- x (* (quot x y) y))))
        (+       ,+)
        (^       ,expt)
        (ceil    ,ceiling)
        (exp     ,exp)
        (floor   ,floor)
        (log     ,log)
        (sqrt    ,sqrt)

     ))

;;
;; What category of object is this?
;;

(define (what-kind value)
    (cond ((real? value) 'real)
          ((vector? value) 'vector)
          ((procedure? value) 'procedure)
          (else 'other)))

;;
;; Main function.
;;
```

```

(define (main argvlist)
    (symbol-put! 'n (expt 2.0 32.0))
    (symbol-put! 'a (make-vector 10 0.0))
    (vector-set! (symbol-get 'a) 3 (symbol-get 'pi))
    (printf "2 ^ 16 = ~s~n" ((symbol-get '^) 2.0 16.0))
    (printf "log 2 = ~s~n" ((symbol-get 'log) 2.0))
    (printf "log10 2 = ~s~n" ((symbol-get 'log10) 2.0))

    (newline)
    (printf "*symbol-table*:~n")
    (hash-for-each *symbol-table*
        (lambda (key value)
                (printf "~s : ~s = ~s~n" key (what-kind value) value))
    ))

(main '())

```

```
2 ^ 16 = 65536.0
log 2 = 0.6931471805599453
log10 2 = 0.30102999566398114

*symbol-table*:
exp : procedure = #<procedure:exp>
ceil : procedure = #<procedure:ceiling>
^ : procedure = #<procedure:expt>
rem : procedure = #<procedure:...es/./symbols.scm:35:18>
quot : procedure = #<procedure:...es/./symbols.scm:34:18>
log10 : procedure = #<procedure:...es/./symbols.scm:32:18>
div : procedure = #<procedure:...es/./symbols.scm:31:18>
e : real = 2.718281828459045
sqrt_2 : real = 1.4142135623730951
log10_2 : real = 0.3010299956639812
+ : procedure = #<procedure:+>
floor : procedure = #<procedure:floor>
n : real = 4294967296.0
a : vector = #(0.0 0.0 0.0 3.141592653589793 0.0 0.0 0.0 0.0 0.0 0.0)
pi : real = 3.141592653589793
mod : procedure = #<procedure:...es/./symbols.scm:33:18>
sqrt : procedure = #<procedure:sqrt>
log : procedure = #<procedure:log>
```