Macros Perform Code Transformations

A macro is an operation performed on the source code of a program before evaluation.

Macros exist in many languages, but are easiest to define correctly in a language like Lisp.

Scheme has a `define-macro` special form that defines a source code transformation:

```scheme
(define-macro (twice expr)
  (list 'begin expr expr))
```

```scheme
> (twice (print 2))
2
2
```

Evaluation procedure of a macro call expression:
- Evaluate the operator sub-expression, which evaluates to a macro.
- Call the macro procedure on the operand expressions without evaluating them first.
- Evaluate the expression returned from the macro procedure.

(Demo)

For Macro

Discussion Question

Define a macro that evaluates an expression for each value in a sequence:

```scheme
(define-macro (for sym vals expr)
  (list 'map
       (list 'lambda (list sym) expr)
       vals))
```

```scheme
(define-macro (for x (2 3 4 5) (+ x x))
  (list 'map
       (list 'lambda (list sym) expr)
       vals))
```

(Demo)