

A Scheme Expression is a Scheme List

```
Scheme programs consist of expressions, which can be:
```

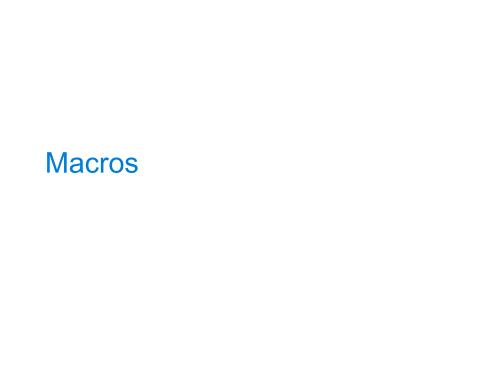
- Primitive expressions: 2 3.3 true + quotient
- Combinations: (quotient 10 2) (not true)

The built-in Scheme list data structure (which is a linked list) can represent combinations

```
scm> (list 'quotient 10 2)
(quotient 10 2)
scm> (eval (list 'quotient 10 2))
5
```

In such a language, it is straightforward to write a program that writes a program

(Demo)



Macros Perform Code Transformations

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

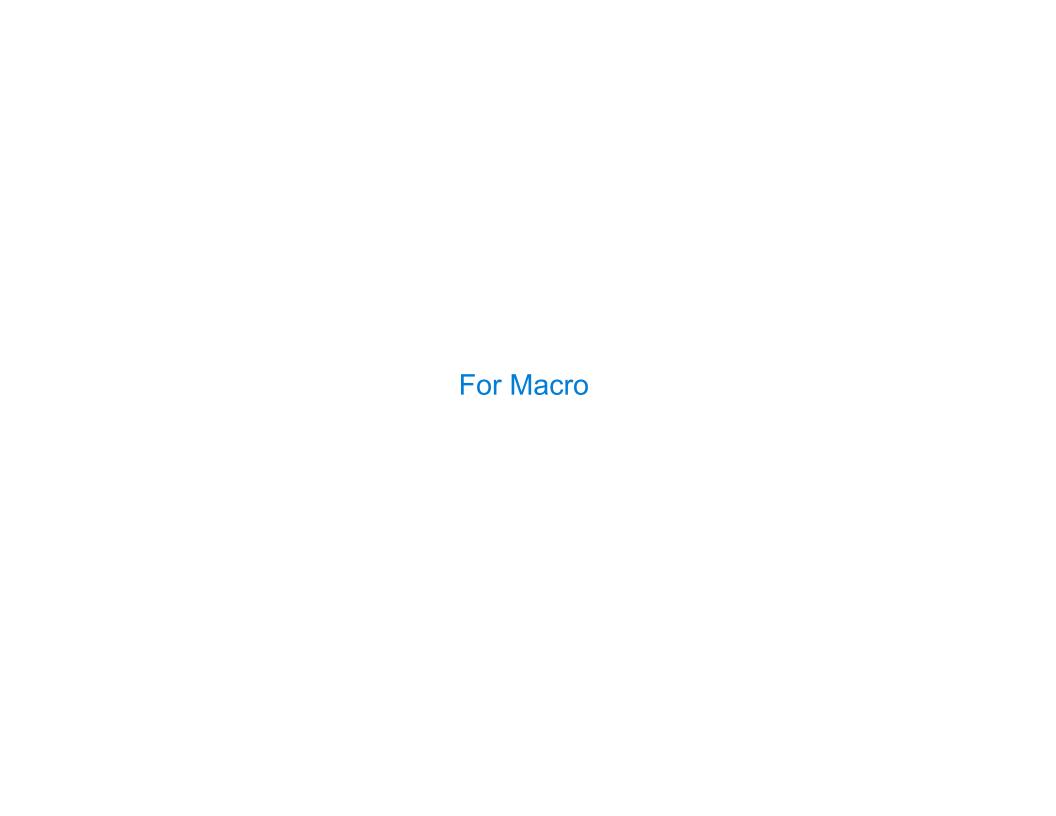
Macros exist in many languages, but are easiest to define correctly in a language like Lisp

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

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)



Discussion Question

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

U

Quasi-Quotation

(Demo)