BASIC EXPRESSIONS.

Write the result from evaluating each expression, or “ERROR” if the expression cannot be evaluated.

If there is a horizontal line across the page, that means that the environment is cleared out and anything that’s defined is gone.

Otherwise, definitions persist from problem to problem.

(+ 3 4)

(5)

(define square (lambda (n) (* n n)))
(square 3)

square

(square)
SOME BASIC DATA TYPES.

So far we know about numbers and booleans.

Symbols are also an essential datatype. A symbol is created with a single-quote; e.g.:

'foo

creates a symbol named foo.

PREDICATES.

A predicate is an expression that evaluates to true or false.

“true” is represented with #t and “false” is with #f.

The convention is that predicate functions are named with a trailing ?.

There is a set of predicates for querying the type of an object.

What is the result of evaluating the following?

(symbol? 'foo)

(number? 'foo)

(boolean? 'foo)

(< 3 4)

answer: #t

(< 3 'foo)

(<= 3 4 5)
THE “IF” EXPRESSION

The “if” expression consists of four things:

- the keyword if
- an expression that must evaluate to a boolean
- a “then expression” which is evaluated if the boolean is true
- an “else expression” which is evaluated if the boolean is false.

E.g.:

```scheme
(if (< 3 4) 'smaller 'larger) evaluates to 'smaller.
```

Questions.

Does “if” use special evaluation rules? Explain.

What is the value of this series of two expressions? Give a number or “ERROR”.

```scheme
(define foo 3)
(if (< 3 4) foo bar)
```

What will happen if you leave out the “else expression”?
BOOLEAN OPERATORS.

The “not” operator exists; \((\text{not} \ f)\) is \(\text{#t}\) and \((\text{not} \ \text{#t})\) is \(\text{#f}\).

“and” and “or” accept an arbitrary number of arguments (like + and *).

\text{and} \text{ and or} \text{ evaluate from left to right and short-circuit (stop executing) as soon as the result is determined; e.g. in the case of and, when encountering the first expression that evaluates to false.}

Do \text{and} \text{ and or} use special evaluation rules? Explain.

What is the value of the following expressions? (or “ERROR”)

\begin{verbatim}
(define foo 3)
(and #t (= foo 3))
(or #t (= bar 4)
(or #f (= foo 3))
(and)
(or)
\end{verbatim}