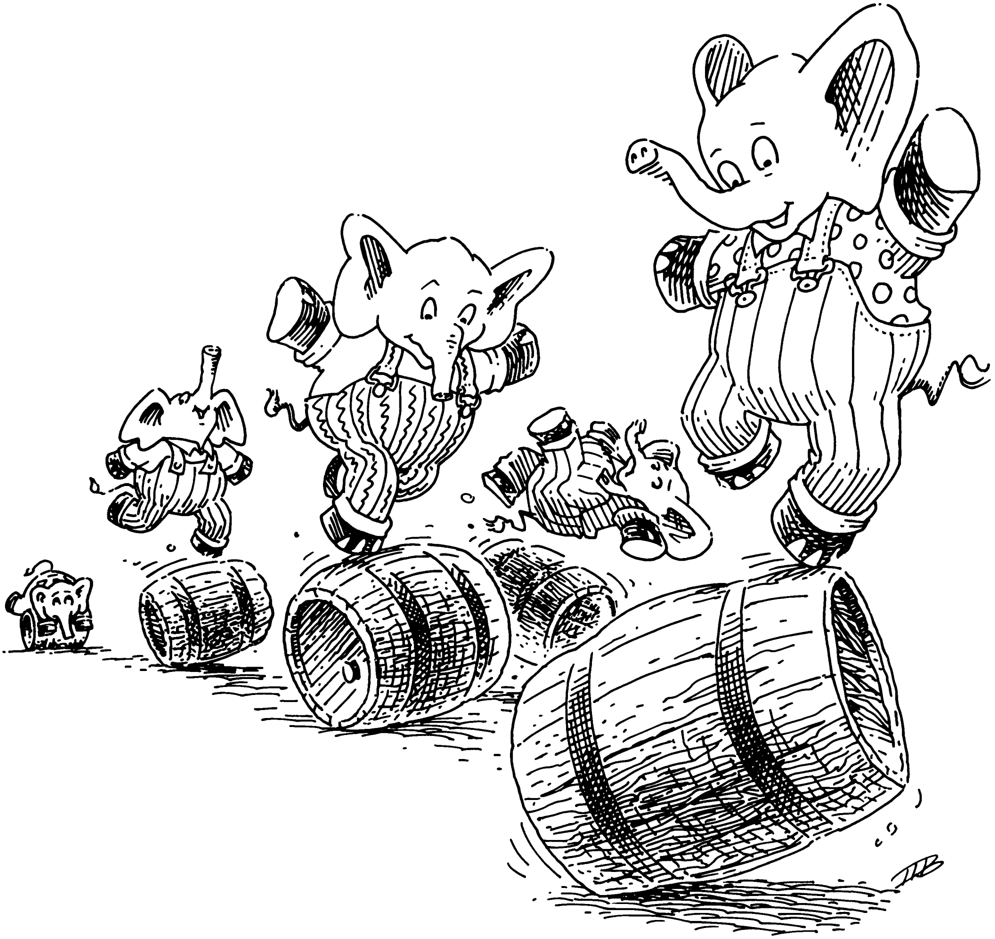
2.
Do It, Do It Again, and Again, and Again...



True or false: (*lat? l*) where *l* is (Jack Sprat could eat no chicken fat)	True, because each S-expression in *l* is an atom.
True or false: (*lat? l*) where *l* is ((Jack) Sprat could eat no chicken fat)	False, since (*car l*) is a list.
True or false: (*lat? l*) where *l* is (Jack (Sprat could) eat no chicken fat)	False, since one of the S-expressions in *l* is a list.
True or false: (*lat? l*) where *l* is ()	True, because it does not contain a list.
True or false: a lat is a list of atoms.	True! Every lat is a list of atoms!
Write the function *lat?* using some, but not necessarily all, of the following functions: *car cdr cons null? atom?* and *eq?*	You were not expected to be able to do this yet, because you are still missing some ingredients. Go on to the next question. Good luck.

Are you rested?

```
(define lat?[1]
  (lambda (l)
    (cond
      ((null? l) #t)
      ((atom? (car l)) (lat? (cdr l)))
      (else #f))))
```

What is the value of (*lat? l*)
where
  l is the argument (bacon and eggs)

#t.
  The application (*lat? l*)
  where
    l is (bacon and eggs)
  has the value #t —true—because *l* is a lat.

How do we determine the answer #t for the application
  (*lat? l*)

You were not expected to know this one either. The answer is determined by answering the questions asked by *lat?*
  Hint: Write down the definition of the function *lat?* and refer to it for the next group of questions.

What is the first question asked by (*lat? l*)

(*null? l*)
  Note:
    (**cond** ...) asks questions;
    (**lambda** ...) creates a function; and
    (**define** ...) gives it a name.

What is the meaning of the **cond**-line
  ((*null? l*) #t)
where
  l is (bacon and eggs)

(*null? l*) asks if the argument *l* is the null list. If it is, the value of the application is true. If it is not, we ask the next question. In this case, *l* is not the null list, so we ask the next question.

What is the next question?

(*atom?* (*car l*)).

[1] L:
```
(defun lat? (l)
  (cond
    ((null l) t)
    ((atom? (car l)) (lat? (cdr l)))
    (t nil)))
```

What is the meaning of the line ((*atom?* (*car* *l*)) (*lat?* (*cdr* *l*))) where *l* is (bacon and eggs)	(*atom?* (*car* *l*)) asks if the first S-expression of the list *l* is an atom. If (*car* *l*) is an atom, we want to know if the rest of *l* is also composed only of atoms. If (*car* *l*) is not an atom, we ask the next question. In this case, (*car* *l*) is an atom, so the value of the function is the value of (*lat?* (*cdr* *l*)).
What is the meaning of (*lat?* (*cdr* *l*))	(*lat?* (*cdr* *l*)) finds out if the rest of the list *l* is composed only of atoms, by referring to the function with a new argument.
Now what is the argument *l* for *lat?*	Now the argument *l* is (*cdr* *l*), which is (and eggs).
What is the next question?	(*null?* *l*).
What is the meaning of the line ((*null?* *l*) #t) where *l* is now (and eggs)	(*null?* *l*) asks if the argument *l* is the null list. If it is, the value of the application is #t. If it is not, we ask the next question. In this case, *l* is not the null list, so we ask the next question.
What is the next question?	(*atom?* (*car* *l*)).
What is the meaning of the line ((*atom?* (*car* *l*)) (*lat?* (*cdr* *l*))) where *l* is (and eggs)	(*atom?* (*car* *l*)) asks if (*car* *l*) is an atom. If it is an atom, the value of the application is (*lat?* (*cdr* *l*)). If not, we ask the next question. In this case, (*car* *l*) is an atom, so we want to find out if the rest of the list *l* is composed only of atoms.
What is the meaning of (*lat?* (*cdr* *l*))	(*lat?* (*cdr* *l*)) finds out if the rest of *l* is composed only of atoms, by referring again to the function *lat?*, but this time, with the argument (*cdr* *l*), which is (eggs).

What is the next question?

(*null? l*).

What is the meaning of the line
  ((*null? l*) #t)
where
  l is now (eggs)

(*null? l*) asks if the argument *l* is the null list. If it is, the value of the application is #t —true. If it is not, move to the next question. In this case, *l* is not null, so we ask the next question.

What is the next question?

(*atom?* (*car l*)).

What is the meaning of the line
  ((*atom?* (*car l*)) (*lat?* (*cdr l*)))
where
  l is now (eggs)

(*atom?* (*car l*)) asks if (*car l*) is an atom. If it is, the value of the application is (*lat?* (*cdr l*)). If (*car l*) is not an atom, ask the next question. In this case, (*car l*) is an atom, so once again we look at (*lat?* (*cdr l*)).

What is the meaning of (*lat?* (*cdr l*))

(*lat?* (*cdr l*)) finds out if the rest of the list *l* is composed only of atoms, by referring to the function *lat?*, with *l* becoming the value of (*cdr l*).

Now, what is the argument for *lat?*

().

What is the meaning of the line
  ((*null? l*) #t)
where
  l is now ()

(*null? l*) asks if the argument *l* is the null list. If it is, the value of the application is the value of #t . If not, we ask the next question. In this case, () is the null list. So, the value of the application (*lat? l*)
where
  l is (bacon and eggs), is #t —true.

Do you remember the question about
  (*lat? l*)

Probably not. The application (*lat? l*) has the value #t if the list *l* is a list of atoms
where
  l is (bacon and eggs).

Can you describe what the function *lat?* does in your own words?

Here are our words:
"*lat?* looks at each S-expression in a list, in turn, and asks if each S-expression is an atom, until it runs out of S-expressions. If it runs out without encountering a list, the value is #t. If it finds a list, the value is #f—false."
To see how we could arrive at a value of "false," consider the next few questions.

This is the function *lat?* again:

```
(define lat?
  (lambda (l)
    (cond
      ((null? l) #t)
      ((atom? (car l)) (lat? (cdr l)))
      (else #f))))
```

What is the value of (*lat?* *l*)
where
  l is now (bacon (and eggs))

#f,
  since the list *l* contains an S-expression that is a list.

What is the first question?

(*null?* *l*).

What is the meaning of the line
  ((*null?* *l*) #t)
where
  l is (bacon (and eggs))

(*null?* *l*) asks if *l* is the null list. If it is, the value is #t. If *l* is not null, move to the next question. In this case, it is not null, so we ask the next question.

What is the next question?

(*atom?* (*car* *l*)).

What is the meaning of the line
  ((*atom?* (*car* *l*)) (*lat?* (*cdr* *l*)))
where
  l is (bacon (and eggs))

(*atom?* (*car* *l*)) asks if (*car* *l*) is an atom. If it is, the value is (*lat?* (*cdr* *l*)). If it is not, we ask the next question. In this case, (*car* *l*) is an atom, so we want to check if the rest of the list *l* is composed only of atoms.

What is the meaning of
(*lat?* (*cdr l*))

(*lat?* (*cdr l*)) checks to see if the rest of the list *l* is composed only of atoms, by referring to *lat?* with *l* replaced by (*cdr l*).

What is the meaning of the line
((*null? l*) #t)
where
l is now ((and eggs))

(*null? l*) asks if *l* is the null list. If it is null, the value is #t. If it is not null, we ask the next question. In this case, *l* is not null, so move to the next question.

What is the next question?

(*atom?* (*car l*)).

What is the meaning of the line
((*atom?* (*car l*)) (*lat?* (*cdr l*)))
where
l is now ((and eggs))

(*atom?* (*car l*)) asks if (*car l*) is an atom. If it is, the value is (*lat?* (*cdr l*)). If it is not, we move to the next question. In this case, (*car l*) is not an atom, so we ask the next question.

What is the next question?

else.

What is the meaning of the question **else**

else asks if **else** is true.

Is **else** true?

Yes, because the question **else** is always true!

---

else

Of course.

Why is **else** the last question?

Because we do not need to ask any more questions.

Why do we not need to ask any more questions?

Because a list can be empty, can have an atom in the first position, or can have a list in the first position.

What is the meaning of the line
(**else** #f)

else asks if **else** is true. If **else** is true—as it always is—then the answer is #f—false.

What is
    )))

These are the closing or matching parentheses of (**cond** . . . , (**lambda** . . . , and (**define** . . . , which appear at the beginning of a function definition.

Can you describe how we determined the value #f for
  (*lat? l*)
where
  l is (bacon (and eggs))

Here is one way to say it:

> "(*lat? l*) looks at each item in its argument to see if it is an atom. If it runs out of items before it finds a list, the value of (*lat? l*) is #t. If it finds a list, as it did in the example (bacon (and eggs)), the value of (*lat? l*) is #f."

Is (**or** (*null? l1*) (*atom? l2*)) true or false
where *l1* is ()
and
  l2 is (d e f g)

True,
  because (*null? l1*) is true where *l1* is ().

Is (**or** (*null? l1*) (*null? l2*)) true or false
where
  l1 is (a b c)
and
  l2 is ()

True,
  because (*null? l2*) is true where *l2* is ().

Is (**or** (*null? l1*) (*null? l2*)) true or false
where
  l1 is (a b c)
and
  l2 is (atom)

False,
  because neither (*null? l1*) nor (*null? l2*) is true where
    l1 is (a b c)
  and
    l2 is (atom).

What does (**or** . . .) do?

(**or** . . .) asks two questions, one at a time. If the first one is true it stops and answers true. Otherwise it asks the second question and answers with whatever the second question answers.

Is it true or false that *a* is a member of *lat* where *a* is tea and *lat* is (coffee tea or milk)

True, because one of the atoms of the lat, (coffee tea or milk) is the same as the atom *a*—tea.

Is (*member? a lat*) true or false where *a* is poached and *lat* is (fried eggs and scrambled eggs)

False, since *a* is not one of the atoms of the lat.

This is the function *member?*

```
(define member?
  (lambda (a lat)
    (cond
      ((null? lat) #f)
      (else (or (eq? (car lat) a)
                (member? a (cdr lat)))))))
```

What is the value of (*member? a lat*) where *a* is meat and *lat* is (mashed potatoes and meat gravy)

#t, because the atom meat is one of the atoms of *lat*, (mashed potatoes and meat gravy).

How do we determine the value #t for the above application?

The value is determined by asking the questions about (*member? a lat*). Hint: Write down the definition of the function *member?* and refer to it while you work on the next group of questions.

What is the first question asked by (*member? a lat*)

(*null? lat*). This is also the first question asked by *lat?*.

The First Commandment

(preliminary)

Always ask *null?* as the first question in expressing any function.

What is the meaning of the line

```
((null? lat) #f)
```

where

lat is (mashed potatoes and meat gravy)

(*null? lat*) asks if *lat* is the null list. If it is, the value is #f, since the atom meat was not found in *lat*. If not, we ask the next question. In this case, it is not null, so we ask the next question.

What is the next question?

else.

Why is **else** the next question?

Because we do not need to ask any more questions.

Is **else** really a question?

Yes, **else** is a question whose value is always true.

What is the meaning of the line

```
(else (or (eq? (car lat) a)
          (member? a (cdr lat))))
```

Now that we know that *lat* is not *null?*, we have to find out whether the *car* of *lat* is the same atom as *a*, or whether *a* is somewhere in the rest of *lat*. The answer

```
(or (eq? (car lat) a)
  (member? a (cdr lat)))
```

does this.

True or false:

```
(or (eq? (car lat) a)
  (member? a (cdr lat)))
```

where *a* is meat

and

lat is (mashed potatoes and meat gravy)

We will find out by looking at each question in turn.

Is (*eq?* (*car lat*) *a*) true or false
where *a* is meat
and
lat is (mashed potatoes and meat gravy)

False,
because meat is not *eq?* to mashed,
the *car* of
(mashed potatoes and meat gravy).

What is the second question of (**or** . . .)

(*member? a* (*cdr lat*)).
This refers to the function with the argument *lat* replaced by (*cdr lat*).

Now what are the arguments of *member?*

a is meat
and
lat is now (*cdr lat*), specifically
(potatoes and meat gravy).

What is the next question?

(*null? lat*).
Remember The First Commandment.

Is (*null? lat*) true or false
where
lat is (potatoes and meat gravy)

#f—false.

What do we do now?

Ask the next question.

What is the next question?

else.

What is the meaning of
(**or** (*eq?* (*car lat*) *a*)
(*member? a* (*cdr lat*)))

(**or** (*eq?* (*car lat*) *a*)
(*member? a* (*cdr lat*)))
finds out if *a* is *eq?* to the *car* of *lat* or if *a* is a member of the *cdr* of *lat* by referring to the function.

Is *a eq?* to the *car* of *lat*

No, because *a* is meat and the *car* of *lat* is potatoes.

So what do we do next?	We ask (*member? a* (*cdr lat*)).
Now, what are the arguments of *member?*	*a* is meat, and *lat* is (and meat gravy).
What is the next question?	(*null? lat*).
What do we do now?	Ask the next question, since (*null? lat*) is false.
What is the next question?	**else**.
What is the value of (**or** (*eq?* (*car lat*) *a*) (*member? a* (*cdr lat*)))	The value of (*member? a* (*cdr lat*)).
Why?	Because (*eq?* (*car lat*) *a*) is false.
What do we do now?	Recur—refer to the function with new arguments.
What are the new arguments?	*a* is meat, and *lat* is (meat gravy).
What is the next question?	(*null? lat*).
What do we do now?	Since (*null? lat*) is false, ask the next question.
What is the next question?	**else**.

What is the value of
(**or** (*eq?* (*car lat*) *a*)
(*member? a* (*cdr lat*)))

#t,
because (*car lat*), which is meat, and *a*, which is meat, are the same atom. Therefore, (**or** ...) answers with #t.

What is the value of the application
(*member? a lat*)
where *a* is meat
and
lat is (meat gravy)

#t,
because we have found that meat is a member of
(meat gravy).

What is the value of the application
(*member? a lat*)
where *a* is meat
and
lat is (and meat gravy)

#t,
because meat is also a member of the *lat*
(and meat gravy).

What is the value of the application
(*member? a lat*)
where *a* is meat
and
lat is (potatoes and meat gravy)

#t,
because meat is also a member of the *lat*
(potatoes and meat gravy).

What is the value of the application
(*member? a lat*)
where *a* is meat
and
lat is (mashed potatoes and meat gravy)

#t,
because meat is also a member of the *lat*
(mashed potatoes and meat gravy).
Of course, this is our original *lat.*

Just to make sure you have it right, let's quickly run through it again. What is the value of (*member? a lat*)
where
a is meat
and
lat is (mashed potatoes and meat gravy)

#t.
Hint: Write down the definition of the function *member?* and its arguments and refer to them as you go through the next group of questions.

(*null? lat*)

No. Move to the next line.

---

else	Yes.
(**or** (*eq?* (*car lat*) *a*) (*member? a* (*cdr lat*)))	Perhaps.
(*eq?* (*car lat*) *a*)	No. Ask the next question.
What next?	Recur with *a* and (*cdr lat*) where *a* is meat and (*cdr lat*) is (potatoes and meat gravy).
(*null? lat*)	No. Move to the next line.
else	Yes, but (*eq?* (*car lat*) *a*) is false. Recur with *a* and (*cdr lat*) where *a* is meat and (*cdr lat*) is (and meat gravy).
(*null? lat*)	No. Move to the next line.
else	Yes, but (*eq?* (*car lat*) *a*) is false. Recur with *a* and (*cdr lat*) where *a* is meat and (*cdr lat*) is (meat gravy).
(*null? lat*)	No. Move to the next line.
(*eq?* (*car lat*) *a*)	Yes, the value is #t.

(**or** (*eq?* (*car lat*) *a*) (*member? a* (*cdr lat*)))	#t.
What is the value of (*member? a lat*) where *a* is meat and *lat* is (meat gravy)	#t.
What is the value of (*member? a lat*) where *a* is meat and *lat* is (and meat gravy)	#t.
What is the value of (*member? a lat*) where *a* is meat and *lat* is (potatoes and meat gravy)	#t.
What is the value of (*member? a lat*) where *a* is meat and *lat* is (mashed potatoes and meat gravy)	#t.
What is the value of (*member? a lat*) where *a* is liver and *lat* is (bagels and lox)	#f.
Let's work out why it is #f. What's the first question *member?* asks?	(*null? lat*).
(*null? lat*)	No. Move to the next line.

else	Yes, but (*eq?* (*car lat*) *a*) is false. Recur with *a* and (*cdr lat*) where *a* is liver and (*cdr lat*) is (and lox).
(*null? lat*)	No. Move to the next line.
else	Yes, but (*eq?* (*car lat*) *a*) is false. Recur with *a* and (*cdr lat*) where *a* is liver and (*cdr lat*) is (lox).
(*null? lat*)	No. Move to the next line.
else	Yes, but (*eq?* (*car lat*) *a*) is still false. Recur with *a* and (*cdr lat*) where *a* is liver and (*cdr lat*) is ().
(*null? lat*)	Yes.
What is the value of (*member? a lat*) where *a* is liver and *lat* is ()	#f.
What is the value of (**or** (*eq?* (*car lat*) *a*) (*member? a* (*cdr lat*))) where *a* is liver and *lat* is (lox)	#f.

What is the value of (*member? a lat*)
where *a* is liver
and
lat is (lox)

#f.

What is the value of
(**or** (*eq?* (*car lat*) *a*)
(*member? a* (*cdr lat*)))
where
a is liver
and
lat is (and lox)

#f.

What is the value of (*member? a lat*)
where *a* is liver
and
lat is (and lox)

#f.

What is the value of
(**or** (*eq?* (*car lat*) *a*)
(*member? a* (*cdr lat*)))
where
a is liver
and
lat is (bagels and lox)

#f.

What is the value of (*member? a lat*)
where *a* is liver
and
lat is (bagels and lox)

#f.