This backwards-chaining rule-based expert system, Animal Consultant, is:

Not because it's so tempting, being a rudimentary first gasp, but I
might like to use some of the work sometime in a salable way.

In the second place, much of the basis is plagiarized through translation
out of the Winston and Horn Lisp book; though I've added quite a bit,
and changed some most obvious shortcomings in that engine's function.

It will, in addition to solving the simple problem, it will inform you of what it's
doing at any time, and explain how it reached a conclusion afterwards.

Be sure there's a great deal missing still.

You are completely welcome to play with it, copy it for friends, etc.;
just not sell it or incorporate it or parts of it in anything for sale.
In any case, Xlisp is restricted to non-profit use.

Getting ready to go:

To run this thing, double-click on xlisp, and when it comes up with the prompt,
type (load "animals") and a return. Don't forget the quotes.

The user front end should be self-explanatory (!). It accepts three
commands: yes, no, and why. It'll let you know when one of them is
inappropriate.

If you say you want the progress report, you'll be informed of the hypothesis
which is being attempted at each step.

If you say why, instead of yes or no, to answer a question, you'll be shown the
current rule under investigation.

After the run, you can keep asking for explanations of each of the hypothesis that
fired each rule.

When you say you don't want to try again, you'll be dumped into xlisp itself. To
get back to the finder, use (exit). To run the program again, type (consult).

To experiment, you can write code while in xlisp, though there's no way
to save what you've done without the file system. Or use an editor to create an
ascii (plain document) file, and (load "whatever you namedit").

Have fun!

Space Considerations:

In a 128k Mac, there isn't really enough room. With a rule set which takes up more room or causes deeper search recursion than this very tiny set, you'll get stack crashes (bomb 28).

Setting the workspace smaller with a lower expand will hurt severely by causing more gcs at a second or so each. That situation is already almost intolerable.

I haven't tried it, but with a 512k Mac, you should be able to use the extra space to great advantage. Many more rules, faster operation; though Xlisp, and this version -- without, for instance, the member primitive -- is certainly no speed demon. I'd experiment with (alloc and) expand, checking mem each time you run, until it can handle a full set of no answers without or with minimal gc's.

Xlisp:

For a tutorial on Xlisp, see March 85 Byte. However, this release, though it's self-named as 1.2, doesn't agree with the article. It uses the older C-like commands, such as foreach instead of mapcar, and is missing all but the most primitive Lisp commands. Also, I haven't doped out getting the file manager to work, if it does. The last hope is that this might be set up with the object Class system, but haven't bothered to try that route too far, since there's not room or commands (rplaca etc.) for much of an online editor anyway.