CS61A Project 4, Part I: Learn Logo by Doing

To acquaint yourself with Logo, write a Logo program `numspell` that takes an integer in the range $[0, 999]$ and returns a sentence that spells it out:

```
? print numspell 999
nine hundred ninety nine
? print numspell 1
one
? print numspell 11
eleven
? print numspell 23
twenty three
? print numspell 0
zero
? print numspell 514
five hundred fourteen
? print numspell 40
forty
? print numspell 234
two hundred thirty four
? print numspell 500
five hundred
```

Here are some Logo primitives that may be of help: `first`, `equalp`, `lessp`, `butfirst`, `remainder`, `sentence`. You may also find the following functions useful:

```
to dec :n
output difference :n 1
end

to list.ref :n :stuff
if equalp :n 0 [output first :stuff]
output list.ref dec :n butfirst :stuff
end

to digit.spell :n
output list.ref :n [zero one two three four five six seven eight nine]
end
```

You should write additional helper functions—don't do it all in one hideous program. Please put your code into a file called `numspell.lg` and submit it along with your project. As in Scheme, a semicolon denotes the start of a one-line comment in Logo.

Do not type your procedure definitions at the Logo prompt. Compose your source in Emacs, then use Logo's `load` command to input the file:

```
? load "numspell.lg
```

Oh yeah, to run Berkeley Logo, just type `logo` at the shell prompt. To quit Logo, say `bye`.