CISC280 Spring 2007 Lab 2

Xenon wants to write a decimal-to-binary converter in Scheme. S/he knows about what columns in number systems mean about values, but cannot figure out how to use that information to convert a number in a program without writing something really inefficient.

Then Zorpox shows Xenon an algorithm for computing binary numbers that does not involve lots of exponentiation, but only uses division by two. It works like this:

1. If n is less than 2; write n.
2. Divide n by 2; if there is a remainder, write 1. If there is no remainder, write a 0.
3. Recurse on what is left of n after the division.

Xenon happily[1] implements this algorithm, and gets the following results:

```
> (convert 0)
0
> (convert 4)
001
> (convert 13)
1011
```

Write Xenon's recursive convert program using **only** the following scheme operators: lambda, =, <, quotient, remainder, display, newline.

Xenon can tell s/he is close to the solution, but that the output is reversed. Xenon wants to fix the problem, but knows that the printed results of **display** cannot be turned around. Then Xenon realizes that the placement of the calls in the program can be changed, and that will change the output. After a little fiddling Xenon gets:

```
> (decimal2binary 0)
0
> (decimal2binary 1)
1
> (decimal2binary 127)
1111111
> (decimal2binary 128)
10000000
>
```

Modify your program to produce the correct result without adding any new operations (but you may add a wrapper).

If you need to submit (see syllabus if you are unsure) submit your code file(s) and a script of several well-chosen test cases via MyCourses (due Thursday midnight) and on paper (to your TA's mailbox Friday by 1 p.m.) to receive full credit.

When you use MyCourses, remember that you can "upload" files multiple times, but you only click "submit" once.

[1] Well s/he wasn't happy at first; then s/he remembered two things: 1) you can get Scheme to "write" something with the **display** command, and 2) when you are using a **cond**, you can have more than one expression be evaluated for a single true condition.