SW483, Summer 2017
CS131 Homework Assignment #8, "Looping Printer", in Lisp!

A Looping Printer Program

In this assignment, you will compose a program that uses a loop to print text. Your program will take an integer from the user and print the text "Lisp" that many times. It should print "Lisp" the number of times specified by the user, all on the same line and with no spaces or carriage returns in between. Your code only needs to work with positive integers (1 or greater).

For example, if the integer 3 is entered, the output from your program should be "LispLispLisp" and should look like this:

```
Enter an integer and press return.
3
LispLispLisp
```

If the integer 10 is entered, the output from your program should look like this:

```
Enter an integer and press return.
10
LispLispLispLispLispLispLispLispLispLisp
```

Your starter code for this assignment will be a file called looping_printer.lisp which takes an integer from the console and has a loop, but is otherwise incorrect. You can compile, run, and test the incomplete calculator program using the same steps as for previous assignments. There is a zip archive containing the code (looping_printer.lisp).

To compile and run the looping printer program:

1. Download LoopingPrinter.zip. It contains a Lisp source code file, looping_printer.lisp, a bash shell script, assignment_8_grader.sh, and a file with test cases: test_cases_loops.csv.

2. Use the bash shell and change directories to the folder created by unpacking the .zip file.

3. Then simply execute "clisp looping_printer.lisp" and follow the instructions in the console to run the program.

Extra Credit: Using Recursion

For extra practice with recursion, try implementing the looping printer with a recursive function instead of a loop. Below is a sketch of how you might do this by defining a recursive function called "recursive-print" and making a call to that function with firstInteger as an argument:

```
(format t "Enter an integer and press return.~%")
(setq firstInteger (read))
(defun recursive-print (x)
  (cond ((eq x 0))
        (T (format t "~a" "Lisp") (recursive-print … ))))
(recursive-print firstInteger)
```

The first s-expression in the "cond" represents the "base case" where the argument to recursive-print is 0 and nothing is printed. If the argument is anything other than 0, The second s-expression matches (since the test is T, which is always non-NIL). In this case "Lisp" is printed only once, and a recursive call to recursive-print is made. The key is to make it so that the recursive call has an argument that is less than "x".

Avoiding Infinite Loops

When working on your program and submitting your program to the autograder, you'll need to be careful not to implement a loop that executes infinitely many times. If when you run your program it seems to freeze and doesn't perform the inputs and outputs you expected, it may be stuck in an infinite loop. You can stop your program by pressing Ctrl-C.

Running the Autograder Script

Run the autograder script by executing "./assignment_8_grader.sh" from the bash shell in the zip directory. Your Lisp file must be named "looping_printer.lisp" with a lower-case "l" and all other letters lower-case.

The script will attempt to compile your looping_printer.lisp and will test your code by executing it using the clisp interpreter on a number of inputs. When it completes, the autograder script will print your final score (out of 100) to the console. Ignore any compiler warnings.

The autograder does run your program on with a timeout, so be careful. If your program loops infinitely it will freeze the autograder script and you will need to press Ctrl-C to exit it.