Homework, Week 2, part C

lightprogramming.org

September 9, 2016

1 Analysis and design

Create a new file, call it `my-guess.lisp`, to allow the computer to pick a number for you to guess. Go through the analysis and design phases. A little thought before hand will save you a lot of work later on.

Before you write any code, think about the requirements of the new program, and how you would design the program. In other words, do the analysis and design phases. After you make the changes, test your new program to see if it meets the requirements.

2 Implementation and testing

Implement and test the `my-guess.lisp`. I have copied my version below. You are allowed to look at my answer, but before you look at my answer, try to write it yourself. When you have written it, test it.

Do an internet search for “common lisp function format”, “common lisp function incf”, and “common lisp function cond.” I do not expect you to understand these, yet, but you will a little later on in the course. You should already understand the functions `(=)`, `(<)`, and `(>)` from Week 0.

```lisp
;;; my-guess.lisp
(defun start-guess-number ()
  (defparameter number (+ 1 (random 100)))
  (defparameter tries 0)
  (format t "The computer has guessed a number between 1 and 100 ~%")
  (play-game))

(defun play-game ()
  (format t "Enter a number between 1 and 100:  ")
  (incf tries)
  (let ((guess (read)))
    (cond
      ((= number guess)
       (format t "You guessed the number! It took you ~a tries.~%" tries))
      ((> number guess)
       (format t "Your guess was too small.~%")
       (play-game))
      ((< number guess)
       (format t "Your guess was too big.~%")
       (play-game)))))
```