Running CMUCL Lisp on Taz

1. Make sure you have an account on Taz. If not, see Dr. Kline.

2. Log on to Taz.

3. Start emacs.

4. Visit (open) a `.lisp` file to put emacs into the correct major mode. Make sure the following then appears in the mode line (the line at the bottom of emacs).

   ```
   (Lisp adoc Slime)
   ```

5. In that file, write a lisp expression, such as a `defun`. Here's a test case:

   ```
   (defun FF (L)
     "Return the first of a list"
     (first L))
   ```

6. Start Lisp: split the screen and start Lisp in the other window: on the machines in the lab, you can use the F9 key to do all this (itmay take a couple of seconds). Alternatively, start Lisp by executing the command `slime`; that is, enter `M-x slime`. (On most keyboards, the meta- key (M-) is the Alt key. If you have trouble with no Meta- key, which works like the Control or Shift keys, you can still execute commands using the ESC key. In this case you enter ESC followed by `x` followed by the name of the desired command.)

   When started, there will be an input prompt `CL-USER>` and `REPL Autodoc` (for "read-eval-print-loop") will appear on the mode line of the Lisp window. And the window, the one with the Lisp code or file, will now have on its mode line:

   ```
   Lisp Autodoc [COMMON-LISP-USER cmucl])
   ```

   thereby showing that CMUCL-lisp is running it that window.

7. Inform lisp of the function definition: you can either execute the command `slime-eval-defun` (using M-x or ESC x) which evaluates the function definition alone (and thus informs Lisp of the definition). When done, the name of the function, in our case `FF`, will appears at the beginning of the mode line.

   Alternatively, you can execute `slime-load-file` which loads the entire file (evaluates every s-expression in it and thus informs Lisp of every `defun` in the file). In this case, only `T` (for "true") appears in the mode line.

   For convenience, I have bound `slime-eval-defun` to F10, and `slime-load-file` to F11.

   It is also possible, but not recommended, to cut and paste any Lisp expression from the file directly into Lisp at the prompt. It is even possible to type the expression in directly at the prompt. It is strongly recommended not to do either.

8. Evaluate the desired expression at the REPL prompt. For example:

```
CL-USER> (FF '(q w e r))
Q
CL-USER>
```

9. In the Lisp file window, execute the emacs command `C-h m` (Control h, followed by m) to see `help` on the Lisp mode. Among the functions you will see is `slime-repl-previous-input` (it is bound to `ESC p` and also `M-p`) which "pops" the previous input expression. It can be used repeatedly to pop much earlier expressions. (It is also part of Lisp that *, **, and *** hold the values of the last three expressions evaluated.) There are many other useful Lisp-mode functions. Bind your favourites to keys: to see how, look in your ∼`/.emacs` file.

10. The `REPL` window is just a text buffer and can be saved and treated like any other text buffer: you can save it, print it, etc. This is how you should capture any runs.