98-172 : Great Practical Ideas for Computer Scientists

Time To Make A Choice (I Choose You, `emacs`!)

Getting Started

Welcome to Lab-itation 3! You seem to have chosen `emacs`! When you logged in, you should have gotten a message to update your `.bashrc` file. Type `gpi_update` on the command line to update it.

Part of the update was to add a new `gpi_install` function which will let you install important things that we give you.

If you run `gpi_install`, it will give you some potential things to install. You should install `.emacs` before continuing. Go ahead and open up `emacs`. You should have a copy of the `emacs` cheatsheet on-hand before proceeding.

Getting Your Source File

We have provided example source files for you to use for this lab-itation. The catch is that they are provided on our website, so you will have to copy them into `emacs`.

If you want to use `python`, you can copy the file using the command
`cp /afs/andrew/course/98/172/www/examples/example.py ./`

If you want to use `c0`, you copy the file using the command
`cp /afs/andrew/course/98/172/www/examples/example.c0 ./`

Now that you have the code, the sixth line should be a space for your Andrew ID. Move to that line (for example, by `C-u 6 C-n` to go down 6 lines, or just typing `C-n` six times).

Now, move to the end of line by typing `C-e`, and enter your andrewid!

Get Ready To Code

In this section, you'll finish up the header of the file. It teaches **searching**, **selecting**, and **deleting a line**.

(a) Search for `delete-this-text-and-replace-it-with-your-name` in the file by typing `C-s`, then entering the string to search for. Once you've found it, type `C-g` to exit interactive-search-mode.

(b) Make sure you're at the *beginning* of the text you want to delete, then hit `C-SPC` to *mark* the current position.

(c) Move to the end of the string you want to delete, for example, by repeating `M-f` to move forward a word. Since you *marked* the beginning, you should see the word being hilighted as you move the cursor.

(d) Type `C-w` to kill (delete) the selected text.

(e) The next line is too personal for you to leave it in the file; move to it, then go to the beginning of line using `C-a`, and kill the line by typing `C-k` once to delete the text, then `C-k` once again to remove the empty line.

Fix Some Stuff

In this section, we will cover **search** and **replace**.

(a) Search for the text `this_function_name_is_terrible` using `C-s`.

(b) To replace the text `X` with the text `Y`, you should type `M-% RET`, the enter the text to replace (in this case, `this_function_name_is_terrible`) and type `RET`, then the text to replace it with (in this case, `add`) and hit `RET`. It will then ask you for each occurrence of the string whether to replace it or not. You can type `y` each time, or hit `!` to replace all occurrences with no further prompting.

Note that `M-%` will only replace occurrences from the point (current location of the cursor) until the end of the buffer; to replace in the entire buffer, you can go to the beginning of the file using `M-<`, then use `M-%`.

(c) Unfortunately, the person who was writing the file "accidentally" mispelled terrible as turrible. To fix this, use `M-%` again to replace "`turrible`" with "`terrible`", then use `M-%` *again* as in the previous step (remember to jump to the top of the file using `M-<`). But this time, after typing `M-% RET`, instead of typing out the whole string "`this_function_name_is_terrible`", you can type `M-p` repeatedly to cycle through previous inputs you've given to the `M-%` command and which `emacs` has remembered, including "`this_function_name_is_terrible`". Once you find it, hit `RET`; you can now use `M-p` again to find the replacement text `add` (or just type it directly).

Remove Useless Code?

`do_nothing` is really a rather silly function. Delete it in any way you wish.

But wait! `do_nothing` is being used by `loop_forever`. We can't delete it. Use `C-/` to undo your changes (if `C-/` doesn't work, try `C-_` or `C-x u`). Type `C-/` repeatedly to undo changes until you have undeleted `do_nothing`.

If you undo too much, you'll need to redo. Emacs doesn't have redo functionality per se; instead, it records your undos as actions and lets you undo them. First, though, you have to do something other than an undo, like typing some nonsense text. Then start undoing until you've redone all the changes you wanted to make.

Fixing Indentation

To fix the indentation of a group of lines, you should hilight them by setting a mark using `C-SPC` and moving the cursor. You then have several options. The easiest is `C-M-\`, which indents the selected region according to emacs' auto-indentation rules. Alternatively, you can explicitly indent or dedent the region by n spaces by supplying n as a *prefix argument* to `C-x TAB`, as in `C-u` n `C-x TAB`. n can be positive, to indent, or negative, to dedent.

Save Your Code

Save your code by typing `C-x C-s`.

Now, switch back to your shell. If you are running emacs graphically, this should be no problem. Otherwise you may need to suspend emacs by typing `C-z`.

If you are using `C0`, compile your code on the command line using `cc0 example.c0`. There will be an error that looks like `examples/example.c0:23.49-23.49:error:Parameter missing type`.

If you are using `python`, run your code on the command line using `python example.py`. There will be an error about indentation.

In either case, go ahead and fix the error, then run the code again.

If you suspended emacs, you can resume it using the `fg` command.

Make it Prettier

If you are using `c0`, your program still won't compile, because the `do_nothing` function is being used before it is created. In general, this is bad style; so, even if you are using python, you should fix this.

Select all of the lines of the `do_nothing` function using `C-SPC` as before; then kill them using `C-w`. If you wanted to only copy them instead, you would use `M-w` instead. To paste them, move to the line above `loop_forever` and type `C-y` to "yank" them back.

Finish It Up

`do_nothing` and `loop_forever` are fairly useless functions; so, go ahead and delete them, then remove the call to `loop_forever` from `add`. Then re-compile/re-run. You should get sane output.