Subject: zero-padding and emacs
From: Joseph Picone <joseph.picone@gmail.com>
Date: 8/26/23, 6:58 PM
To: nedc_research <nedc_research@googlegroups.com>
CC: ECE 1111 <temple_engineering_ece1111@googlegroups.com>, ENGR 2011 <temple_engineering_engr2011@googlegroups.com>

As many of you know, we use emacs macros extensively to clean up data. Let me demonstrate a typical example of this through a simple example involving zero-padding. Suppose your goal is to convert the following data:

```
 123
 1234
 12345
```

to something like this:

```
 000123
 001234
 012345
```

And what do you do if you have to do this for 10,000 lines?

Why would we want to do this? We might be creating a list of filenames from some spreadsheet data that contains numeric subject identifiers, and we might want to merge other information, like a sample identifier or date, into the filename. But in the end we want each filename to be the same length.

While this can be done any number of ways in Excel, Python, Shell, C/C++, etc., it can be done very quickly in emacs. Here are the steps:

(1) Pad each line with an arbitrarily large number of zeros:

```
 123   => 000000000123
 1234  => 0000000001234
 12345 => 00000000012345
```

using a macro or a global replace if you can get away with it.

(2) Write macro that does the following:

```
 jump to end of line (ctrl e)
 backward space 6 characters
 set the mark
 jump to beginning of line (ctrl a)
 cut (ctrl w)
 go to the end of line (ctrl e)
 go to the next line (ctrl f)
```

This will leave each line with exactly six characters whether the original line had 1, 2, 3 ..., 6 characters.

Once you know a bit of emacs, you can get quite creative at solving these kinds of problems using a sequence of macros. Often, I don't do everything in one step. I do things in several simple steps so each macro is fairly simple and easy to write and debug.

In fact, if I may borrow a concept from ENGR 2011, the problem is you have a set of basic capabilities, called basis functions, and you want to map a task (a vector) onto these basis functions. You want to represent the vector in terms of these basis functions.

A Fourier series is a good example of this - we map an arbitrary periodic signal into a weight sum of sines and cosines (the basis functions).

ChatGPT is an example of an algorithm that actually does a very nice job of these kinds of things - it figures out how to solve your specific problem using components it has learned from all the

information available in its training data set.

The more code you look at, the more you realize there are certain basic building blocks common to most programs. We rarely write a new program from scratch. We also adapt existing code.

The Unix operating system, of which Linux is a variant, was also built on this principle. Provide users a simple set of commands with common interfaces, and let the user build more complex functions out of these commands. We call this command line programming.

But the key thing here is that you have to a programmer's mentality - the ability to break a complex task down into a series of simple steps for which you already have basic emacs functions. This is what we call problem decomposition.

-Joe