Homework #3: It's Not My (De)Fault

Issued: Thursday, October 1
Due: Thursday, October 15

Purpose

This assignment allows you to learn more about Unix/C development, by improving your solution to the previous assignments.

Rather than hardcoding *all* of the character categories, some common ones will be hardcoded, while others will be passed into the executing program, as command-line arguments. Thus, the number of categories will vary dynamically, at run time, without recompilation. At this stage, there will still be a compile-time maximum number of categories, but it can be changed without editing source code. Make and the C preprocessor will do the work.

We will also allow categories to contain a couple of simple abbreviations: capitalization folding and character ranges.

Finally, we'll see a clever way to produce better error messages.

Assignment

Build upon your previous character-category counting program. As before, allocate memory for the `chrcats` array statically, but now according to the definition of a preprocessor macro `MAXCATS`. Do *not* initialize the array with categories, as we did before. The array starts out empty, with just a `{0}` entry:

```
typedef struct {...} ChrCat;
typedef ChrCat ChrCats[...];
static ChrCats chrcats={{0}};
```

Have `main` add three "builtin" categories: lowercase vowels, lowercase consonants, and letters (uppercase and lowercase). Then, have `main` process an even number of command-line arguments: a sequence of zero or more name/category pairs. After that, `main` behaves pretty much as before.

Other Requirements

- Employ good modularity, formatting, and documentation.
- Detect and report errors, with:

  ```
  pub/List/error.h
  ```

  For now, you can ignore the other files in that directory. Test and demonstrate that errors are detected.
- Do *not* use `<strings.h>` or its cousins. Write your own functions.
- Allow a category to specify capitalization folding, with the first character. For example, `"^aeiou"` represents the set of uppercase and lowercase vowels.
- Allow a category to specify character ranges. For example, `"a-z0-9"` represents the set of digits and lowercase letters.
- Write your own little makefile, which includes this makefile:

  ```
  pub/GNUmakefile
  ```

  If Make defines `MAXCATS` on the `gcc` command line, use that value. Otherwise, have your source code use a reasonable default value, like ten (hint: use `#ifndef`). Whence this assignment's title.
- Run `valgrind` on your program.