Recitation 02

Focus

- Structs, Vectors and Functions

Recitation Project

We have been given a file of chemical formulae, specifically hydrocarbons. Hydrocarbons consist of only hydrogen and carbon atoms.

Each line of the file contains the name of the chemical followed by its molecular formula which will be in the form**:**
C#H#
where the two #'s represent the number of carbon (C) and hydrogen (H) atoms, respectively..
(No name in the file contains blanks.)

Because hyrdocarbons with the same molecular formula can appear in many structural forms and each of the forms has a different name, the same molecular formula might appear more than once in the file, each time with different a name. (*e.g.*: Butane and 2-methylpropane both have the molecular formula C4H10.)

Our goal is to maintain one entry for each unique molecular formula along with all the names for that formula and then display all these entries in order by ascending number of carbon atoms. We will do this without implementing or using a full sorting algorithm, by simply making multiple passes over the vector, first printing all forumlas with one carbon, then all with two carbons, etc.

Each molecular formula will be kept in a struct with *three* fields**:**

- the name(s) for formula
- the number of carbon atoms
- the number of hydrogen atoms

We will be storing all these structs in a vector.

An example

Suppose this is the file**:**

```
n-Butane C4H10
```

```
Propyne C3H3
1,3-Butadiyne C4H2
Hexane C6H14
Butane C4H10
iso-Butane C4H10
Pentane C5H12
```
Your program should store these five entries and then display**:**

```
C3H3 Propyne
C4H10 n-Butane Butane iso-Butane
C4H2 1,3-Butadiyne
C5H12 Pentane
C6H14 Hexane
```
Notice that the output is "sorted" only by the number of carbon atoms. There is no attempt to sort by anything else.
Or, put another way**:** all other orderings show simply the order the items were taken from the file.

Discussion

As you have all studied data structures by now, you might have some good ideas of other ways to design a solution to this program. A map, for example, would save having to search the vector to see if an entry for the formula already exists. And if the map was a sorted map, perhaps implemented with a self-balancing binary tree, the issue of displaying the elements in some sorted order could have been simpler and more efficient.

Why aren't we doing this? Simply because we are avoiding throwing more of C++'s syntax and libraries at you. Our goal now is to be sure that you become adept at the basics of the language. We will introduce later things like the STL (Standard Template Library) map class along with the representation of iterators.

So while the design for this lab may seem a bit "clumsy", please understand that it is an exercise in the language features we have covered so far.

Notes

As always, write good clean code. What does that mean?

- Good use of functions to make code more readable.

- Good naming of everything.
- No long lines. What's long? Anything over 80 characters.

Submit

- Whether or not you have been "checked off" during the recitation, submit a single source file, `rec02.cpp`, on NYU Classes. If you do not, your grade may be penalized. Do *not* submit any other files, such as those generated for your development environment (IDE).