COMP 4959: Lab 4

Submit a zip file `lab4.zip` containing your 2 source files. Maximum score: 10

The file `atomic-weights.txt` contains information about the atomic elements. Each row has 4 columns: atomic number, atomic symbol, name of element & approximate atomic weight. You are asked to implement 2 independent modules `Lab4a` and `Lab4b` (i.e., neither module is allowed to call functions from the other module) containing functions that are generated using the techniques of metaprogamming. Note that you don't explicitly write the specified functions; you write code that at compile time "writes" those functions.

1. Implement a module `Lab4a` that defines a set of functions whose names are the symbols (converted to all lowercase) of the elements in the atomic weights file. The defined functions take no arguments and return the atomic weight (a number) of the corresponding element. Effectively, we are creating functions like

```
def h() do
  1.008
end

def he() do
  4.003
end
```

2. Implement a module `Lab4b` that has a function `atomic_weight` that, when passed an atomic symbol (a string) from those listed in the given atomic weights file, returns the corresponding atomic weight (a number). This function must not explicitly use any data structure to look up the atomic weight of an element, e.g., the function must not use a map or association list to do so. The function returns `-1` if the passed symbol is not known. For example, `atomic_weight("He")` returns 4.003.

Put your code in 2 separate files: `lab4a.ex` and `lab4b.ex`. You must use metaprogramming techniques for this lab. Each module must read `atomic-weights.txt` at compile time (e.g., via `elixirc lab4a.ex`) to generate the specified functions.