Exercises:

1. Rewrite the struct *Fraction* as a class. All the free functions that make up the API of the struct *Fraction* should be coded as member functions of this new class.
   a) Review the naming of *Fraction*'s member functions!
   b) Add one or more meaningful ctors to *Fraction*.
   c) Review the comments of the API, so that users can understand and use *Fraction* and its API, w/o inspecting the code. – Are there any differences compared to the API with free functions?
   d) Update the formerly created menu-driven application to let the user play with the updated API.
2. Write the UDT *Date* and add a dctor initializing a new *Date* instance with the current date.

Remarks:

- Everything that was left unspecified can be solved as you prefer.
- In order to solve the exercises, only use known constructs, esp. the stuff you have learned in the lectures!
- **Please obey these rules for the time being:**
  - **The usage of goto, C++11 extensions, as well as #pragmas is not allowed.**
  - **The usage of global variables is not allowed.**
  - **You mustn't use the STL, esp. *std::string*, because we did not yet understood how it works!**
  - **But *std::cout*, *std::cin* and belonging to manipulators can be used.**
  - **You mustn't use new and delete!**
- Only use classes for your UDTs. The usage of public fields is not allowed! The definition of inline member functions is not allowed!
- Do not put class definitions and member function definitions into separate files (we have not yet discussed separated compilation of UDTs).
- The results of the programming exercises need to be runnable applications! All programs have to be implemented as console programs.
- The programs need to be robust, i.e. they should cope with erroneous input from the user.
- Stick to the agreed upon coding conventions.
- You should be able to describe your programs after implementation. Comments are mandatory.
- In documentations as well as in comments, strings or user interfaces make correct use of language (spelling and grammar)!
- Don't send binary files (e.g. the contents of debug/release folders) with your solutions! Do only send source and project files.
- Don't panic: In programming multiple solutions are possible.
- If you have problems use the Visual Studio help (F1) or the Xcode help, books and the internet primarily.
- Of course you can also ask colleagues; but it is of course always better, if you find a solution yourself.