Notes part 1

Professional C++. Ch. 1. A Crash Course in C++ and the Standard Library

• Access `std:byte` in `<cstddef>` differentiate semantically from `char`.
• C++ has no basic `string` type – use `std::string` in `<string>`
• `std::cin << format(…)` og `std::cout << format(…)` for *formating of I/O* to *console/terminal*
• Use `std::numeric_limits` in `<limits>` in stead of e.g. `#defines INT_MAX`.
• Follow up later: *zero initialization* and *type casting*.
• `std::isnan()` and `std::isinf()` in `<cmath>` (p. 16).
• Operator table (p.16-18).
• Use *strongly-typed* `enum class` instead of *type-unsafe* `enum`.
• Modularity: `export module` in .cppm *modue linterface file* instead of a .h *header file* and .cpp *source code file*.
• Conditional Operator e.g. `cout << (i>2?"yes":"no")`.
• 3-way comparison: `strong_ordering::less`, `strong_ordering::greater` og `strong_ordering::equal`.
– Hvor 3-vejs sammenligningen muligvis ikke har et entydigt resultat, men hvor sammenligningerne stadig har en slags "ordning": `std::weak_ordering::less`, `std::weak_ordering::equivalent` eller `std::weak_ordering::greater`.
– Eller for float: `partial_ordering::less`, `partial_ordering::greater`, `partial_ordering::equal` og `partial_ordering::unordered` (ved `std::isnan()`).
– Follow up later: `<compare>`.
• `auto` instead of `var`
• Attributtes for functions/subrutines (p. 30-33).
• `array`-type in `<array>` requires two parametres e.g. `array<int,3>`, that signify an `int` array with a size of 3.
– iteration is simpler that with older C array types as `int[3]`.
• `std::vector` in `<vector>` a variable array.
• `std::pair` in `<utility>`. (p. 36)
• `std::optional` in `<optional>`. (p. 37)
• Ranged-based for loop (p. 39-40)
• Initializer Lists (p.40)
• Classes (s. 41-44)
• Pointers – avoid where possible (s. 50-52)
• use `nullptr` instead of `NULL`
• const (p. 53-58; 68-69)
• References [-&] (p. 58-68)
• Exceptions (p. 69-70)
• Type (p.70-73)