Clean Code

Chapter 1:

- Bad code can bring the company down.
- Leblanc’s law: Later equals never
- A mess of code decreases the productivity of the team since programmers need to understand the code whenever they want to update it.
- Complete rewriting can be considered when the code becomes too messy.
- Some features of the clean code: readable, elegant, detailed, undistracted
- Code needs to be kept clean over time.

Practical notes:

- Some appropriate variable names:
  1. It should reflect the intention of the programmer.
  2. It should not contain wrong information
  3. One letter variable names are not meaningful
  4. Meaningful names should be used
  5. Searchable names should be used
  6. Class names should be nouns not verbs.
  7. Function names should start with a verb like deleteUser
  8. To clarify the variable names, a meaningful context should be added

- Functions:
  1. Should be small
  2. Should do only one thing
  3. A function should abstract one level of the job (delete page if it is tested)
  4. Should have descriptive and consistent names
  5. Should have little number of arguments (if more than three, create a new class for them)
  6. Should contain exceptions instead of returning error codes

- Comments:
  1. Code should explain itself
  2. Comments can be used for legal reasons and informative reasons
  3. TODO comments can be useful
  4. Warning about the codes can be explained in the comments
  5. Redundant and misleading comments are bad examples

- Formatting:
  1. Formanting is important for communication with other programmers
  2. Lines between functions, statements,

   3. Small distance between related functions and variables
   4. Indentation

- Data structures
   1. Abstraction is crucial for changing the data without knowing the implementation details, as well as putting layers between functions and variables

- Error handling
   1. Use exceptions rather than returning errors
   2. Write try-catch- finally blocks
   3. Coding the necessary exceptions and throwing them is better than using third party exceptions
   4. Dont return null
   5. Dont pass null

- Boundaries
   1. Passing third party interfaces can lead people who have this interface to harm the code
   2. Writing our own interfaces for boundaries of code which doesnt exist prevents us form being blocked