##### Exercise 1

P1: Testing cannot show absence of defects.   
For example can you have tested a software as thoroughly as you can, and think it is correct, but it could show failure in the production environment.

P2: Exhaustive testing is impossible  
You cannot test everything.  
Imagine x input fields with y possible values. To test all combinations, you would need yx tests. That number becomes real big real fast. So a tester should take risks and priorities into consideration.

P3: Early testing: means to start testing as early as possible in the software development life cycle.  
Some benefits are that early-found defects are cheaper to fix and that testers will feel more comfortable with the software product because of early involvement.

P4: Defect clustering: defects are often found in a small number of modules.  
Reasons could be that certain modules are more complex than others or that less skilled programmers wrote that piece of software.

P5: Pesticide paradox  
Review of test cases is necessary, because the same kind of test can only detect a certain number of defects. New bugs will be ignored.

P6: Testing is context dependent   
means that you need different sets of tests for different kinds of software products. A high level software for a rocket going into space needs far more testing than let’s say a gaming app for a phone.

P7: Absence-of-errors-fallacy  
A system that is free of errors could still be unusable if the user’s needs are not met.

No objections, all sound like good things to keep in mind. But I can imagine that the practises in real-life look a bit different, with management pushing software teams further and further, trying to squeeze them to save yet another [insert currency].

##### Exercise 2

1. **The V-model is based on the waterfall model, but an improved version, because testers found that testing needed to be integrated in earlier stages. Every testing phase is associated with a development stage in the development cycle. A new phase can only start after the previous one has been completed. The testing phase which corresponds to the development phase is planned parallel. We have verification phases on one side of the V and validation phases on the other. At the bottom, or the tip, of the V we find the coding phase.**

**Integration into the software development life cycle happens as early as during the requirement analysis, where the testers can start designing acceptance tests. During system requirements, system tests can be prepared, during the global design phase, integration tests are designed. The next phase is the detailed design and implementation. Now, component tests are executed, then the prepared tests in last-in-first-out order (integration testing, system testing, acceptance testing). After successful completion of all phases, the product should be operational.**

**(The test levels can be re-organized, depending on the project’s needs)**

**Benefits:**

* **Easy to use / implement**
* **Time and money saving because testing (planning, designing) is happening before actual coding** 
  + **Also defects detected earlier, are usually cheaper and easier to fix. That is a huge advantage to the waterfall model, where testing happens only at the last phase. Which means, defects undetected in, let’s say, the user requirements phase, “fall down” through every phase of the waterfall and result in an unusable product. The result: waste of time, money and resources. Not to speak of the morale of the involved workers, thinking that they had delivered a high quality piece of software, when in fact they would have to start over again**

1. **Basically, the sequential V-model and agile programming development projects aren’t hard to combine. Every sprint could consist of every sequential V-model phase. But it seems rather impractical to do so. It would result in more workload. In agile development the iterations are called sprints for a reason, things have to go fast. There is no time for designing tests at each iteration or code stubs for incomplete parts to get tests running. Also the software becomes more and more complex after each sprint, regression testing becomes more important. Rather, testers are expected to get involved on a more personal level and to work with less formal documentation, if they can find it at all.**
2. **The V-model ensure the 7 principles in this way……**
3. There are four main stages of testing which must be completed during the software development life cycle (testing): unit testing, integration testing, system testing, and acceptance testing. Software Test types clearly define the goal of a certain level for a program or project.  A test type is focused on a particular test objective, which could be the testing of the function to be performed by the component or system.

* **Test levels:**
  + **Unit or component testing: includes check for resource use (memory leaks), robustness. Single components or units are tested.**
  + **Integration testing: checks software and system design, architecture, use cases and workflows, performance can be included as well. Several components are combined and tested to expose defects in interaction between units.**
  + **System testing: based on requirement specs, use cases, functional specs and risk analysis reports. Can include black- and white-box-techniques. Run in controlled test environment. A complete, integrated system is tested.**
  + **Acceptance testing: checks of usability, operational user acceptance testing (maintenance tasks, migration tasks,..), alpha (performed at development site, but not by the team who designed the software) and beta (performed by customers in environment where the software is supposed to run) testing**
* **Test types (performed at all levels):**
  + **Functional testing (aka black-box-testing): The basis for a functional test is a functional requirement for the system.**
  + **Non-functional testing: Checks for reliability, usability, efficiency, maintainability and/or portability**
  + **Structural testing (aka white-box-testing): Happens at component level. Measures code coverage.**
  + **Testing related to changes: Includes confirmation and regression testing.**