1. List and explain in your own words three characteristics of a good software implementation. (pg 188)

* Readability: can be easily read and understood by other programmers
* Maintainability: can easily be modified and maintained
* Performance: implementation should produce code that performs as fast and efficient as possible

1. Briefly discuss the issues associated with naming variables and procedures in a program. (pg 189)

* Bad name requires extra effort to understand and/or comment to explain.
* It is recommended that long names should be used for global entities and short names for local entities.
* Inconsistent naming convention will also make it harder to understand
* Naming convention can be complicated working in a multicultural team

1. List the four phases of the debugging process. (pg 193)

* Stabilization (or reproduction): be able to reproduce the error on a particular configuration, and to find out the conditions that led to the error by constructing a minimal test case.
* Localization: finding/narrowing to the sections of the code that led to the error
* Correction: changing the code to fic the errors, it is critical that stabilizing and localizing have been completed before this step
* Verification: making sure the error is fixed, and no other errors were introduced after the changes in the code.

1. True or false: You should always optimize your code for performance. Why? (pg 195)

False, because optimizing for performance usually (but not always) affects maintainability and readability for the worse.

1. List three “bad smells” signaling that your code should probably be refactored. (pg 196)

* Duplicated code
* Long methods – methods that are large in contents and should be broken down to smaller methods
* Large class

1. List and briefly explain three of the refactorings mentioned in this chapter. (pg 196)

* Extract method: turns a code fragment into its own method, with an appropriate name, and calls the method
* Substitute algorithm: replaces the body of a method with a new algorithm that is clearer, and returns the same result
* Move method: a process that moves an algorithm from one class to another where it makes more sense
* Extract class: a process that divides into two

1. What is the main difference between a design pattern and code in a code library?

* Design pattern: assist the developer by solving already known problems
* Library: a piece of generic source code that be used by other applications

1. List the four broad ways in which we can take advantage of the cloud.

* Scalability
* Collaboration efficiency
* Flexibility of work practices
* Access to automatic updates

1. Briefly describe **infrastructure as a service** in the context of the cloud.

It’s a form of cloud computing that provides virtualized computing resources over the internet.

1. Briefly describe **platform as a service** in the context of the cloud.

It’s a cloud computing model in which a third-party provider delivers hardware and software tools to users over the internet.

1. List at least two of the application services provided over the cloud.

* Dropbox
* Virtual machine hosting

1. List at least two of the non-application services available in the cloud.

* Infrastructure as a service (IaaS): gives business access to vital web architecture, such as storage space, servers, and connections without the business need of purchasing and managing this internet infrastructure themselves
* Platform as a service (PaaS): clouds are created, many times inside IaaS Clouds by specialists to render the scalability and deployment of any application trivial and to help make expenses scalable and predictable

1. Read a program that you wrote a while ago (the older the better) and a recent program. Can you understand the older program? What changes do you notice in your programming style?

* Most of my oldest codes from Java1 about three years ago, I am able to understand the codes due to its simplicity but very unsatisfied with the ways I coded. Three main problems I found from this piece of code are:
  + Redundant code
  + Poor naming choice (mainly single letter name)
  + Enormous main method
* Some changes I’ve noticed since:
  + Strive for clear and concise code, reduce redundancy as best as I could
  + Codes is longer but methods are smaller
  + Naming for variables and methods are more descriptive and clear