1. What are statements that define and qualify what a program needs to do?

They’re called Program Requirements

1. What are statements that constrain the ways in which software can be designed and implemented?

They’re called Design Constraints

1. Which type of requirement statement defines what the program needs to do?

Functional Requirements

1. List three of the typical kinds of nonfunctional requirements.

* Performance
* Modifiability
* Usability

1. Which decisions are those taken by the software developer about the best ways (processes, techniques, and technologies) to achieve the requirements?

* Programming language
* Algorithms

1. What type of testing refers to testing done by the clients (or somebody on the client's behalf) to make sure the program runs as specified?

It’s called Acceptance Testing

1. What sequence of activities did you observe in considering the programming effort discussed in this chapter?

* Step 1: Estimate ideal time – assuming that there’s no interruption, how long (in minutes) do you think it will take you to finish coding the problem using your favorite language and technology.
* Step 2: Estimate calendar time – divide the task into several subtasks, and estimate how time it will take you to do each task.

1. Is the selection of the programming language for a program considered a requirement, a design constraint, or design decision? Discuss why you think so.

I think selecting a programming language for a program is considered a design constraint. Different languages have different capabilities, and depending on the problem given and/or if we have the choice of choosing language, it can definitely be a design constraint. For example, developing a large system that needs to run large amount of threads might be better in GoLang than Java.

1. Describe one way to simplify a complex problem.

Breaking a complex problem to smaller problems is one way to simplify a complex problem.

1. Identify two technical concerns in developing large systems.

* Size
* Complexity

1. What is the maximum number of communication paths for a team of twenty people?

The maximum number of communication paths for a team of twenty is 190

1. Identify four factors that should be considered in deciding how many post-release people will be needed to support the system.
2. What is meant by **integration** and why is it important to manage this effort for large systems?
3. What are the major tasks in developing and supporting a software system? (Hint: there are six major tasks in the book’s answer to this question.)

* Determine functional and nonfunctional requirements of the system
* Design the system
* Code and unit test the system
* Integration and functionality testing the system
* Release of the payroll system
* Support and maintenance

1. Discuss the three areas that need to be coordinated in a large software project. Is any one area more important than the others? Explain your conclusion.

* Upward scaling of the needed process
* Design structure and contents of the product
* Required personnel

I think all areas are equally important