Book Questions
Chapters 1 and 2
Devayan Mandal

1. 1.2:
   What is the most important difference between generic software product development and custom software development? What might this mean in practice for users of generic software products?

   Difference between Generic and Custom Software Development:
   There are several differences in the software development process between generic and custom software. But, one of the more important differences lies in who decides what goes into the software's end product features. As the name implies, in custom software, the end user's requirements are primarily taken into consideration during all steps of software development. For example, Spotify, which is a highly customizable music player has been designed specifically targeting a younger audience. The user interface offers vibrant, energetic colors and the music software offer songs, albums and playlists which are constantly evolving according to user's music choices.

   On the other hand, software developers predominantly decide the software's vision, product features and delivery guidelines in generic software. Hence, we can deduce that software evolution costs are more than development costs in custom software.

   Implication on generic software users:
   A certain degree of flexibility would be required from generic software users. As the modifications to software can be made with or without prior notice, the end user's would be required to designate a calculated amount towards their budget in lieu with training the staff about the new software's features and functionality. Hence, the end users may be required to allocate work time towards training which may affect the business end goals.

2. 1.3:
   What are the four important attributes that all professional software should possess? Suggest four other attributes that may sometimes be significant.

   Four important attributes are:
   - Attribute 1: The software should attempt to fully meet the end user's requirements.
   - Attribute 2: The software should provide a secure work environment. Adequate security features should be built into the software ensuring that the end user(s) do not need to invest in external security measures.
   - Attribute 3: The software should provide capability to be used multiple times by multiple end users.
   - Attribute 4: Distribution of the software should be possible through practical and widely acceptable means.

Other important attributes:

- Attribute 5: Consumption of memory should be minimal during processes such as installation, startup, run time and shutdown of the software.

- Attribute 6: The software should be obtainable using widely-accepted, conventional modalities. For example, physical installation storage (compact disk, DVD, pendrive) or via online, cloud storage accessibility for immediate download and installation.

- Attribute 7: The software should offer complete functionality on different operating systems like Windows, Mac, Linux and others, unrestricted to the end user(s) operating system platform(s) such as personal computer, mobile, tablets and others.

- Attribute 8: The cost and time required for software maintenance should match or be lesser than similar software products available commercially.

3. 2.3:
   Consider the integration and configuration process model shown in Figure 2.3. Explain why it is essential to repeat the requirements engineering activity in the process.

   Software requirements gathering activity:
   The basis of the software requirements gathering activity is to ensure that all system(s) and component(s) requirements are met before proceeding onto the software development stage.

   The initial requirement activity attempts to understand the system functionality using a broad perspective. Yet, sufficient detail is observed while understanding the requirements to establish available software system or component matches with the user's requirements. The second requirement activity aims to further refine the system and component selection criteria by revisiting available software repositories to optimize on time and cost. By reduction of unnecessary software development, product evolution, design and risk are minimized resulting in faster delivery and deployment.

4. 2.4:
   Suggest why it is important to make a distinction between developing the user requirements and developing system requirements in the requirements engineering process.

   Difference between development of user requirements and system requirements
   User requirements are usually obtained using conventionally understood languages, such as English or the language spoken or written in that region or country. System requirements are expressed in more technical languages as specific systems and components are represented to the software development team. The user's needs, perspective and environment are accounted for and studied while gathering user requirements. Hence, the characteristic of flexibility is usually valued in this stage. The system requirements stage follows and system and component analysis is afforded the possibility to be done on-site or remotely. System requirements tend to be more specific in nature and great attention to detail is valued to avoid or reduce future re-visitation of the requirements gathering procedure.

5. 2.6:
   Explain why software testing should always be an incremental, staged activity. Are programmers the best people to test the programs that they have developed?

   Software testing as an incremental, staged activity:
   In incremental testing each component, module or system of the developed software version is tested individually. Hence, it offers the advantage of quicker error detection as the error can be pin-pointed inside a given unit. This also affords the possibility to modify or upgrade the software without repeating the development process from the beginning. This is highly favorable as the evolution of the present user's market is ever changing.

   Programmers best to test their own software?
   In most cases, especially in large scale software development projects the answer would be no. The possibility of programmers approaching the deliverable product from their own perspective is high. This may or may not match the end user's requirements. Hence, post software development quality assurance analysts engage in testing the software. This ensures that the business requirements, expected functionality, and project goals are met. Usually, the later factors do not directly fall into the scope of the programmer's responsibilities are they are more focused on meeting supplied requirement specifications.