1 A Discipline of Software Design

Section 1.1 The first four questions in this section are designed to engage students in thinking about design and their experiences with it, and consequently have no right or wrong answers. The brief answers below are illustrative only.

1. Our environment, especially inside a building, is usually completely manufactured, and hence thoroughly designed. Even “natural” elements, such as plants, stones, or dirt, are selected by designers for their aesthetic or other qualities.
2. Several apparel manufacturers sell school backpacks and book bags. These containers are carriers for books, school supplies, computers, and other things needed for a day at school or work. They also need to be attractive and fashionable. The problem that their designers must solve thus has both practical and aesthetic or marketing components.
3. Much of the plumbing in eastern Europe features narrow drains, faucets that deliver either very hot or very cold water and nothing in between, and toilet fixtures that subject their users to unfortunate odors. Modern plumbing is designed to be safe and pleasant to use, and it makes life a lot nicer in many small ways.
4. Having used both Apple and Microsoft operating systems and utilities for many years, I can attest to the great pleasure and satisfaction I enjoy using the former and the enormous frustration and loss of productivity I suffer from the latter. These great differences are attributable to both the product design (particularly of the user interface) and the engineering design of these two product lines.
5. Details irrelevant to solving the Chicken Coop problem are that a chain link fence is being constructed, the yard is five acres, the enclosure will hold chickens, and it will enclose a chicken coop. The essential details are that there is 60 feet of material and the enclosure should be of maximum size.

   Details irrelevant to solving the Sheep and Wolf problem are that the shepherdess's name is Sheila, the animals are sheep and a wolf, and they must cross a river in a boat. The essential details are that three things must be conveyed two at a time from one point to another, and two of them cannot be left with the third.
6. Solving the Chicken Coop problem obviously involves considering different sizes and shapes of enclosure. It might also take into account whether one or more sides of the enclosure might use the chicken coop itself (though there is not any information about the size of the chicken coop given in the problem).

   Solving the Sheep and Wolf problem requires considering various sequences of boat trips with different combinations of animals.
7. Many students will use simple drawings to model this situation. Of course, the solution is best found using an equation to model the problem. Many students

will invent a notation to represent states of the situation, such as WB|SS to represent the wolf and boat on the left side of the river and the sheep on the right side of the river. Arrows can then represent boat trips to alter the state of affairs. Such a model is essentially a state diagram. This modeling notation is effective because a complete representation of all alternatives will easily reveal a solution in the form of a path from the initial state to the end state; however, the model is rather large.

8. The Chicken Coop problem is solved by taking advantage of the fact that a circle encloses a larger area for a given circumference than any other plane figure. The radius of such a circular enclosure is $C/2\pi$, which is 9.55 feet when $C = 60$ feet. The area enclosed is πr^2, which is 286.5 square feet when $r = 9.55$ feet.

   A solution to the Sheep and Wolf problem is given as follows using the notation discussed in the previous exercise: SSWB| → SS|WB → SSB|W → S|SWB → SWB|S → W|SSB → WB|SS → |SSWB.

9. A model is a simplification of reality because it necessarily abstracts, and hence simplifies, the target being modeled—if a model was not a simplification, it would be a copy of the target, and not a model at all. Simplification in the form of abstraction is the essence of modeling. On the other hand, there is more to modeling than merely simplification: modeling also requires relationships between the target and the model, as explained in the text. Thus this characterization is insightful but incomplete as a definition.

Section 1.2

10. In principle, any of these tasks could take place during either product or engineering design (see the discussion of the "what" versus "how" distinction). Usually, however, they will occur during the activities indicated below.
   (a) Product design
   (b) Engineering design
   (c) Product design
   (d)Product design
   (e) Engineering design or product design
   (f) Product design or engineering design
   (g) Engineering design
   (h) Product design
   (i) Engineering design or product design
   (j) Product design

11. Many automated versions of board games and card games are quite simple to implement but require very nice interface designs to be successful. For example, solitaire is very easy to program, but it won't be fun to use without an interface that makes it easy and fast to play.

12. Avionics software is notoriously difficult to write. Software that flies planes has strict real-time constraints, must make many difficult calculations, and cannot have flaws. The software product itself, however, is relatively simple to design because it must interface with the pilot and the aircraft in (relatively) simple ways.

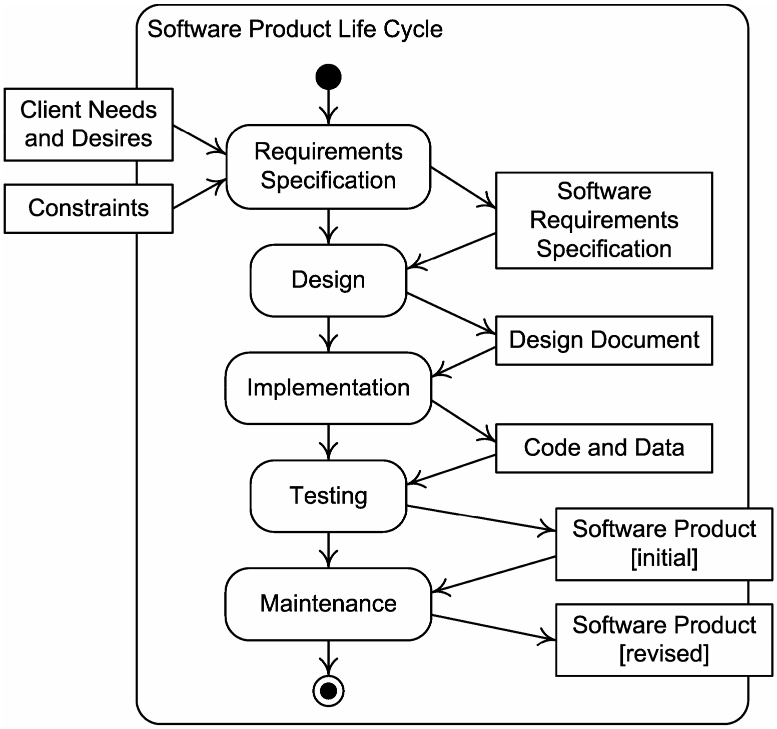
13. A suite of office tools, such as a word processor, spreadsheet, drawing program, database management system, and so forth, must have a sophisticated user interface and interact with one another in various ways, meaning they have challenging product designs. Technically, these products have many features that interact in complex ways, so they are difficult to implement.

Section 1.3

14. Yes, this is a defensible statement of the "what" versus "how" distinction because it focuses on the essential difference between requirements specification and design. This distinction is that a requirements specification states product characteristics in accord with client needs and desires, while a design states how to build the product described in a requirements specification. The traditional "what" versus "how" distinction focuses on the content of requirements and design specifications, which is accidental to the product and its clients.

15. In principle, any statement could appear in either a requirements specification or a design specification because, given a specification, a scenario can be constructed in which it expresses a client need or desire, or a scenario can be constructed in which it states an engineering design directive. In practice, of course, most statements about a product's features, capabilities, and user interactions are requirements specifications, and most statements about implementation are design specifications.

16. One reasonable solution is the following UML activity diagram.


Software Product Life Cycle
Client Needs and Desires
Constraints
Requirements Specification
Software Requirements Specification
Design
Design Document
Implementation
Code and Data
Testing
Software Product [initial]
Maintenance
Software Product [revised]

Section 1.4

17. An example heuristic: Work on the hardest parts of a design first.

18. An example notation: The symbolism of set theory, which uses capital letters for sets, small letters for set elements, and symbols for the set membership, set inclusion, and set identify relations, and the union, intersection, and complement operations.

19. One acceptable solution:

    Top level steps:

```
1. Open the text file and initialize counters
2. Repeat the following until the end of the file
   2a. Read a line and increment the line counter
   2b. Increment the word counter by the number of words in the line
   2c. Increment the character counter by the number of characters
3. Close the text file
4. Report the number of lines, words, and characters
```

    Steps to count words in a line:

```
1. Set the word counter to 0
2. Repeat the following until the end of the line
   2a. Scan past white space
   2b. If not at the end of the line, then
       2b1. Increment the word counter
       2b2. Scan past non-white space characters
3. Report the number of words in the line
```

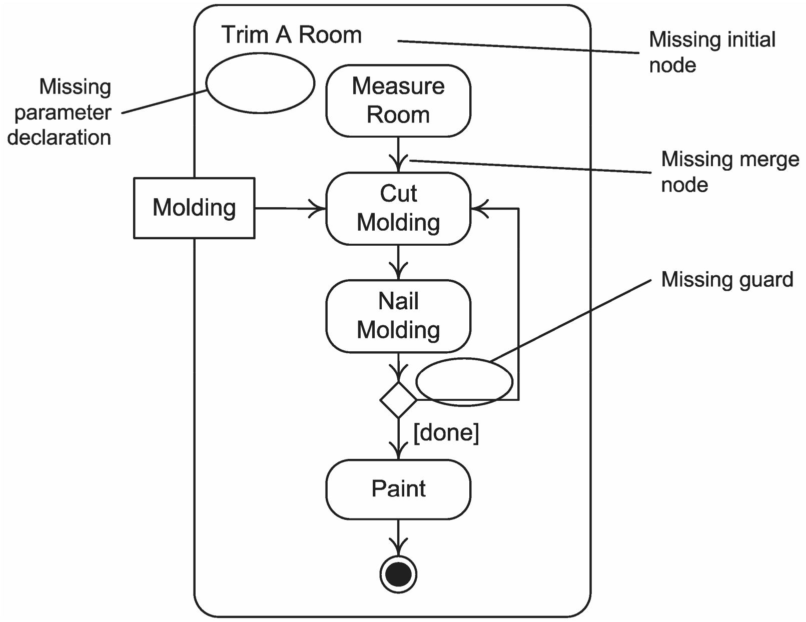
    Steps to count characters in a line:

```
1. Set the character counter to 0
2. Repeat the following until the end of the line
   2a. Scan a character
   2b. Increment the character counter
3. Report the number of characters in the line
```

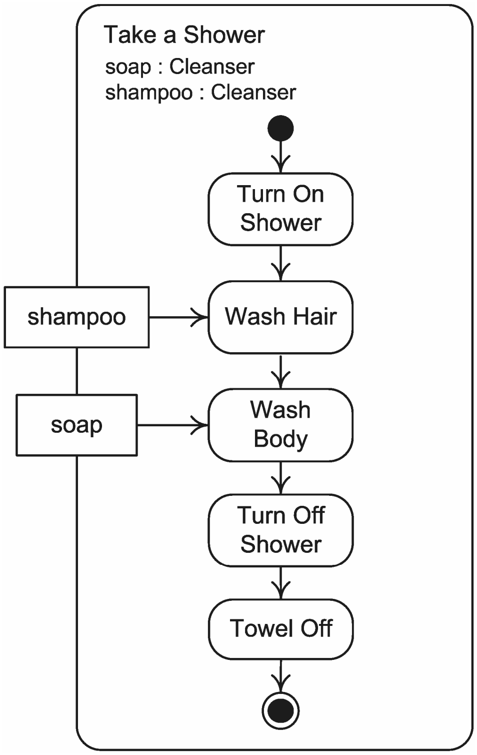
20. The statement conflates structured design and object-oriented design. Structured design indeed relies on top-down functional decomposition, but not (necessarily) to develop the methods in a class (though it could be used for this purpose). Structured design was invented before the object-oriented paradigm was popular, and it is intended for procedural paradigm systems, not object-oriented systems.

2 Software Design Processes and Management

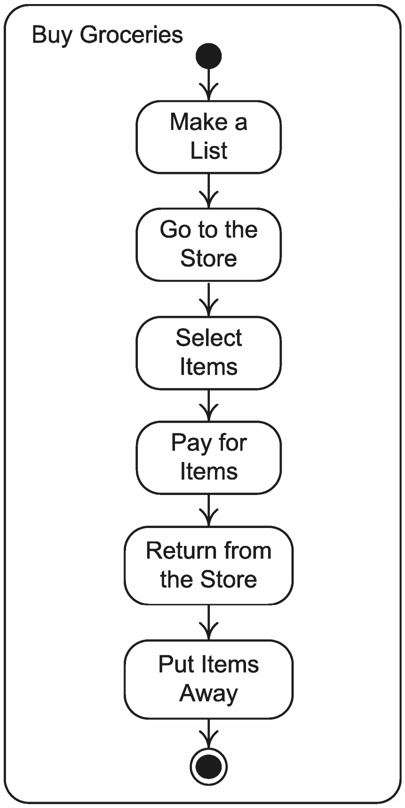
Section 2.1 1. The figure below indicates the errors in the diagram.



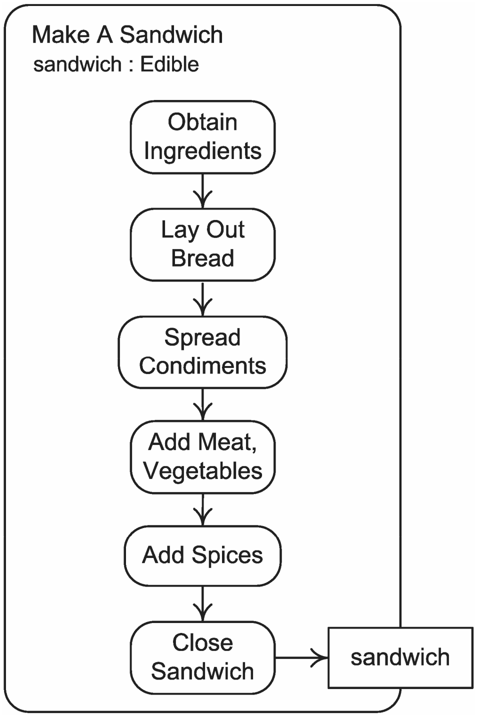
2. There are many ways to model this problem. The following diagram is an acceptable solution.



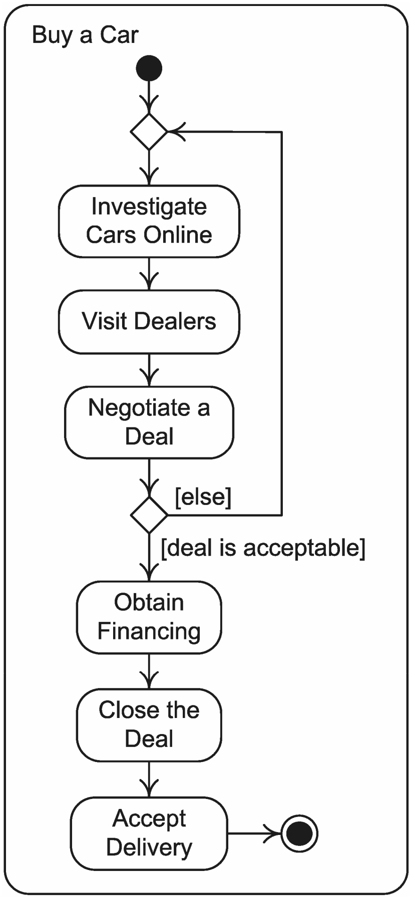
3. The following diagram is an acceptable solution.



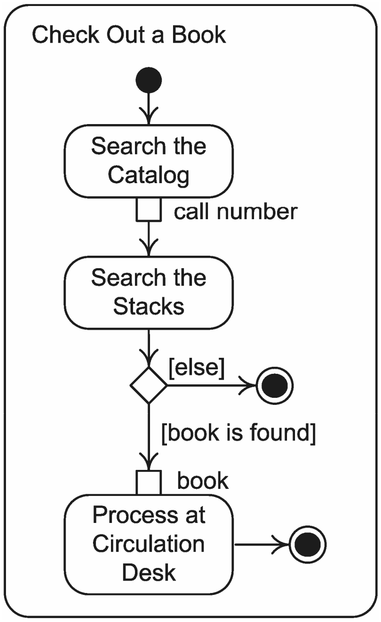
4. The following diagram is an acceptable solution.



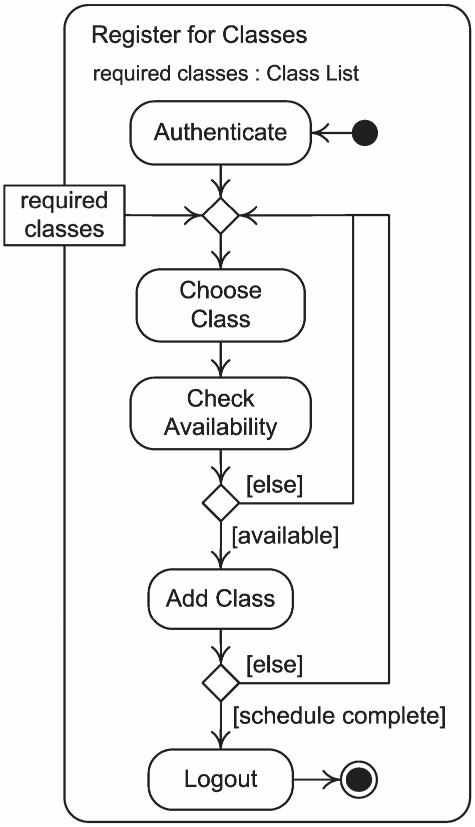
5. The following diagram is an acceptable solution.



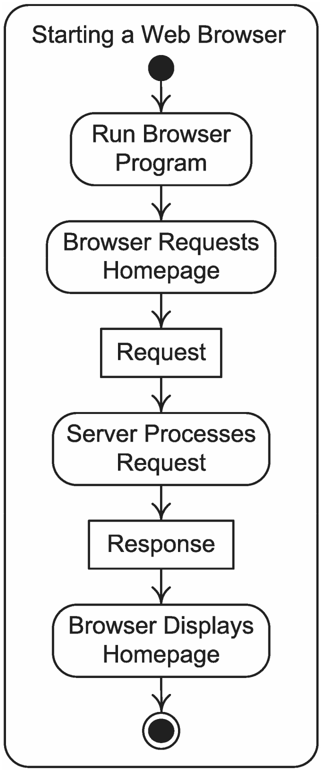
6. The following diagram is an acceptable solution.



7. The following diagram is an acceptable solution.



8. The following diagram is an acceptable solution.


Starting a Web Browser
Run Browser Program
Browser Requests Homepage
Request
Server Processes Request
Response
Browser Displays Homepage