Object-Oriented Data Structures using Java, 4th Edition
Selected Answers to Chapter 1 Exercises
Dale/Joyce/Weems
February, 2016

This file contains answers to exercises from the end of Chapter 1 of the 4th Edition of Object Oriented Data Structures using Java. This file is intended for use only by instructors who have adopted the textbook. Please do not post these answers on the internet. All instructors should be aware that it is possible that these answers will be made public through carelessness or mischievous intent.

Note that we include answers for most, but not all of the exercises. We do not include answers if the question is just a matter of opinion or of personal experience. In such cases we simply state that "answers will vary". Sometimes an exercise is really a "drill" rather than a question, so we don't answer it. We do not provide answers for major coding projects, although we may provide guidelines. Sometimes we do not provide answers for medium sized coding projects, although we hope to provide all of these eventually (check back later; or even better, send us your suggested solution and we will include it and give you credit if it is appropriate). Finally, in cases where the answer to a question is found easily in the body of the text we sometimes indicate the page number where the answer can be found, rather than repeating the information here. The intent of such questions is for the students to describe the information in their own words.

Chapter One – Getting Organized

Many of the questions in this chapter's exercises are "thought questions." The answers given here are typical or suggested responses, but they are not the only possible answers.

Section 1 – Classes, Objects, and Applications

1. Many of the winners' contributions dealt with programming, starting with Perlis in 1966, Dijkstra in 1972, and Knuth in 1974. But the list goes on. The winners whose work dealt primarily with object-orientation are Wirth in 1984, Ole-Johan Dahl and Kristen Nygaard in 2001 and Kay in 2003.

2. See http://www.omg.org/gettingstarted/what_is_uml.htm#12DiagramTypes.

3. A class defines a structure or template for an object or a set of objects. An object is an instance of a class. An example is a blueprint of a building and the building itself. Another example, from the text, of a Java class/object is the *Date* class and the *myDate*, *yourDate*, *ourDate* objects.

4. See pages 5 and 6.

5. According to the program, the number of days between 1/1/1900 and 1/1/2000 is 36524.
   Clearly there are 24 leap years.

   According to the program, the number of days between 1/1/2000 and 1/1/2100 is 36525.
   So that represents 25 leap years.

   The difference is because 1900, being divisible by 100 is not a leap year, whereas 2000, even though it is divisible by 100, is still a leap year because it is divisible by 400.

6. a. Answers will vary.
   b. Answers will vary.
   c. The number of days between11/21/1783 and 7/20/1969 is 67811

9. There are many possible "correct" answers for each part of this question. The intent was not for the student to create code, but rather just identify a good set of variables and methods. Here are sample answers for parts a and b:

a. Time Counter

```
instance variables:   protected int  totalMinutes;
                      protected int counter;

methods:   public void addTime(int minutes, int seconds);
           public void addTime(int minutes);

           public int getTotalTime();
           public int getAvgTime();
```

b. Basketball Statistics

```
instance variables:   protected int score;
                      protected int fieldGoalsAttempted;
                      protected int fieldGoalsMade;
                      protected int freeThrowsAttempted;
                      protected int freeThrowsMade;

methods:   public void fieldGoalMissed();
           public void freeThrowMissed();
           public void fieldGoalMade(int value);      // value indicates 2 or 3 point shot
           public void freeThrowMade();

           public int getScore();
           public float fieldGoalPercentage();
           public float freeThrowPercentage();
```

Section 2 – Organizing Classes

14. See pages 12 – 16.

15. a. *AbstractList*
 b. 3
 c. 31
 d. 16
 e. AbstractCollection

16.
 a. Legal - *getDay* is a public method that returns an *int*
 b. Legal- *getYear* is a public method that returns an *int*
 c. Illegal - *increment* is not defined for *Date* objects
 d. Legal - *increment* is defined for *IncDate* objects
 e. Legal - object variables can be assigned to objects of the same class
 f. Legal - subclasses are assignment compatible with the superclasses above them in the class hierarchy
 g. Illegal - superclasses are not assignment compatible with the subclasses below them in the class hierarchy

19. See pages19 – 21.

20. a. 15 plus 2 in support.cards
 b. 5
 c. i. java.io, java.util, ch05.collections, support
 ii. It uses the wild card character to import all needed classes from the first three packages but it

explicitly imports the *FamousPerson* class from the *support* package

d. i. It depends on the compiler whether a syntax error is generated but in any case other classes will not be able to import and use the class.

ii. Most compliers will simply compile the class although depending on settings you may receive a warning.

iii. You get a syntax error since the compiler cannot locate the needed *Date* class.

21.

a. Yes
b. Labels
c. Length
d. records
e. media

Section 3 – Exceptional Situations

22. See page 27.

Section 4 – Data Structures

26. Answers will vary.

27. In order of occurrence:

alphabetized list of buttons	sorted list
wall map	two dimensional array
connections of trains	linked list
ticket line	queue
direct access to car 4	direct access to an array location
walking through train cars	traversing a linked list
tray holder	stack
Pez dispenser	stack
candy machine button	accessing an array

28. Answers will vary.

29.

a. Nodes are airports; edges connect airports that have flights available between them
b. Nodes are countries; edges connect countries that border each other
c. Nodes are research articles; directed edges connect articles to the articles that they reference
d. Nodes are actors; edges connect actors to the actors they have appeared in the same movies as them
e. Nodes are the computers; edges show a direct network link between the two computers
f. Nodes are rooms; edges are between rooms which have a connecting tunnel
g. Nodes are web pages; directed edges represent a hyperlink from the first page to the second

Section 5 – Basic Structuring Mechanisms

30. a.

<table>
<tr><th>address</th><th>word</th></tr>
<tr><td>. . .</td><td></td></tr>
<tr><td>099</td><td></td></tr>
<tr><td>100</td><td>3</td></tr>
<tr><td>101</td><td>'c' | 'a' | 't' |</td></tr>
<tr><td>. . .</td><td>…</td></tr>
<tr><td>123</td><td>10</td></tr>
<tr><td>124</td><td>20</td></tr>
</table>

	...	
str1:	135	100
	136	

b.

	address	word
	...	
	099	
	100	3
	101	'c' | 'a' | 't' |
	...	…
	123	10
	124	10
	...	
str1:	135	100
str2:	136	100

31. See pages 35-36

32. The output of the code would be

```
5/5/2000
5/5/2000
5/6/2000
5/6/2000
```

33. See page 36.

34. The output of the code would be

```
not equal
equal
equal
```

35. A program that meets the specifications is

```
//----------------------------------------------------------------------
// Exer35.java             by Dale/Joyce/Weems                 Chapter 1
//
// Solves Chapter 1, Exercise 35
//----------------------------------------------------------------------

public class Exer35
{
  public static void main(String[] args)
  {
    int[] squares = new int[10];

    for (int i = 0; i < 10; i++)
      squares[i] = i * i;
    for (int i = 0; i < 10; i++)
      System.out.println(squares[i]);
  }
}
```

36. A program that meets the specifications is

```
//----------------------------------------------------------------------
// Exer36.java             by Dale/Joyce/Weems                 Chapter 1
//
// Solves Chapter 1, Exercise 36
//----------------------------------------------------------------------

public class Exer36
{
  public static void main(String[] args)
  {
    Date[] dates = new Date[10];
    int month, day, year;

    for (int i = 0; i < 10; i++)
      dates[i] = new Date(12, i+1, 2005);

    for (int i = 0; i < 10; i++)
      System.out.println(dates[i]);
  }
}
```

This solution assumes that the `Date` class is “visible” to the `Exer36` application.

Section 7 – Comparing Algorithms: Big-O Analysis

39. Best Case: all answers are 1 – just need to be very lucky

Worst Case:

	Sequential	Binary
a. 10	10	4
b. 1,000	1,000	10
c. 1,000,000	1,000,000	20
d. 1,000,000,000	1,000,000,000	30

40. Answers can vary. The important thing is that the student's answer is clear, and consistent.

41. The answer to the last part of the questions is:
a. approximately 1.37
b. approximately 8.87
c. 4
d. approximately 5.5

42.
a. O(N^2)
b. O(N2)
c. O(N^5)
d. O(N2)

e. $O(N^4)$
f. O(N2)

43.
a. O(N)
b. $O(N^2)$
c. $O(\log_2 N)$
d. O(1)
e. O(N)
f. O(N)

44. The order of growth is O(N). A O(1) approach can be created based on the formula for the sum of the integers between 1 … N being N (N + 1) / 2.

45.
a. O(1)
b. O(N)
c. O(N) ... even though on average takes N/2 steps, that is still O(N)

46.	27	15	83	12	104	28	57	30
	12	15	83	27	104	28	57	30
	12	15	83	27	104	28	57	30
	12	15	27	83	104	28	57	30
	12	15	27	28	104	83	57	30
	12	15	27	28	30	83	57	104
	12	15	27	28	30	57	83	104
	12	15	27	28	30	57	83	104

End of Selected Answers to Chapter 1 Exercises