**Chapter 17 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)17.2.1

**What is a text file and what is a binary file? Can you view a text file or a binary file using a text editor?**

Although it is not technically precise, a text file consists of a sequence of characters and a binary file consists of a sequence of bits. You can use a text editor to view a text file, but not a binary file.

[▼](javascript:collapse('expandText1',%20'collapseText1');)17.2.2

**How do you read or write text data in Java? What is a stream?**

You have to use Java I/O classes to create objects and use the methods in the objects to perform I/O. A Java I/O object is called a stream. An object for reading data is called an input stream and an object for writing data is called an output stream.

Section 17.3

[▼](javascript:collapse('expandText2',%20'collapseText2');)17.3.1

**What are the differences between text I/O and binary I/O?**

Binary I/O reads a byte from a file and copies it directly to the memory with any conversion, vice versa. Text I/O requires encoding and decoding. The JVM converts a Unicode to a file specific encoding when writing a character and coverts a file specific encoding to a Unicode when reading a character.

[▼](javascript:collapse('expandText3',%20'collapseText3');)17.3.2

**How is a Java character represented in the memory, and how is a character represented in a text file?**

Characters are represented using Unicode in the memory and characters are represented in a file using a specified encoding scheme. If no encoding scheme is specified, the system's default encoding scheme is used.

[▼](javascript:collapse('expandText4',%20'collapseText4');)17.3.3

**If you write the string "ABC" to an ASCII text file, what values are stored in the file?**

The values stored in the text file are 0x41 0x42 0x43.

[▼](javascript:collapse('expandText5',%20'collapseText5');)17.3.4

**If you write the string "100" to an ASCII text file, what values are stored in the file? If you write a numeric byte-type value 100 using binary I/O, what values are stored in the file?**

If you write string "100" to an ASCII text file, the values stored are 0x31 0x30 0x30. If you write a numeric byte-type value 100 using binary I/O, the value stored in the file is 0x64.

[▼](javascript:collapse('expandText6',%20'collapseText6');)17.3.5

**What is the encoding scheme for representing a character in a Java program? By default, what is the encoding scheme for a text file on Windows?**

The encoding scheme for representing a character in a Java program is the Unicode. By default, a text file is encoded using ASCII.

Section 17.4

[▼](javascript:collapse('expandText7',%20'collapseText7');)17.4.1

**Why do you have to declare to throw IOException in the method or use a try-catch block to handle IOException for Java I/O programs?**

Almost all the methods and constructors in Java I/O classes, because there are always some unexpected situation may arise during I/O.

[▼](javascript:collapse('expandText8',%20'collapseText8');)17.4.2

**Why should you always close streams? How do you close streams?**

Two reasons: (1) closing a stream ensures that data will be written to the file. (2) closing a stream releases resource acquired by the stream object.  
Two ways to close a stream: 1. Invoking the close() method. 2. Using the try-catch-resource to automatically close the stream.

[▼](javascript:collapse('expandText9',%20'collapseText9');)17.4.3

**The read() method in InputStream reads a byte. Why does it return an int instead of a byte? Find the abstract methods in InputStream and OutputStream.**

The value of a byte is returned as an int in the range 0 to 255. If no byte is available because the end of the stream has been reached, the value -1 is returned. The only abstract method in InputStream is read() and the only abstract method in OutputStream is write(int).

[▼](javascript:collapse('expandText10',%20'collapseText10');)17.4.4

**Does FileInputStream/FileOutputStream introduce any new methods beyond the methods inherited from InputStream/OutputStream? How do you create a FileInputStream/FileOutputStream?**

All the methods in FileInputStream/FileOutputStream are inherited from InputStream/OutputStream. Use new FileInputStream(filename) or new FileInputStream(File) to create a new FileInputStream and use new FileOutputStream(filename), new FileOutputStream(File), new FileOutputStream(filename, true) or new FileOutputStream(File, true) to create a FileOutputStream.

[▼](javascript:collapse('expandText11',%20'collapseText11');)17.4.5

**What will happen if you attempt to create an input stream on a nonexistent file? What will happen if you attempt to create an output stream on an existing file? Can you append data to an existing file?**

A FileNotFoundException would occur if you attempt to create an input stream for a nonexistent file. You can append data in an existent file if the output stream is created using new FileOutputStream(filename, true) or new FileOutputStream(File, true). Otherwise, the file is overridden if it already exists.

[▼](javascript:collapse('expandText12',%20'collapseText12');)17.4.6

**How do you append data to an existing text file using java.io.PrintWriter?**

An instance of FileInputStream can be used as an argument to construct a Scanner and an instance of FileOutputStream can be used as an argument to construct a Formatter. So you can create a Formatter to append text into a file using

new Formatter(new FileOutputStream("temp.txt", true));

If temp.txt does not exist, it is created. If temp.txt already exists, new data is appended to the file.

[▼](javascript:collapse('expandText13',%20'collapseText13');)17.4.7

**Suppose a file contains an unspecified number of double values that were written to the file using the writeDouble method using a DataOutputStream, how do you write a program to read all these values? How do you detect the end of a file?**

You can use the readDouble method in the DataInputSteam to read double values and use the EOFException to detect the end of the file.

[▼](javascript:collapse('expandText14',%20'collapseText14');)17.4.8

**What is written to a file using writeByte(91) on a FileOutputStream?**

writeByte(91) writes one byte for number 91 (0x5B in hex, 01011011 in binary) is written to a file using FileOutputStream.

[▼](javascript:collapse('expandText15',%20'collapseText15');)17.4.9

**How do you check the end of a file in an input stream (FileInputStream, DataInputStream)?**

If the read() method in InputStream returns -1, it is at the end of the file.

[▼](javascript:collapse('expandText16',%20'collapseText16');)17.4.10

**What is wrong in the following code?**

import java.io.\*;

public class Test {

public static void main(String[] args) {

try (

FileInputStream fis = new FileInputStream("test.dat"); ) {

}

catch (IOException ex) {

ex.printStackTrace();

}

catch (FileNotFoundException ex) {

ex.printStackTrace();

}

}

}

Since java.io.FileNotFoundException is a subclass of IOException, the catch clause for java.io.FileNotFoundException should be put before the catch clause for java.io.IOException.

[▼](javascript:collapse('expandText17',%20'collapseText17');)17.4.11

**Suppose you run the following program on Windows using the default ASCII encoding after the program is finished, how many bytes are there in the file t.txt? Show the contents of each byte.**

public class Test {

public static void main(String[] args)

throws java.io.IOException {

try (java.io.PrintWriter output =

new java.io.PrintWriter("t.txt"); ) {

output.printf("%s", "1234");

output.printf("%s", "5678");

output.close();

}

}

}

Java uses Unicode, but Windows uses ASCII. The Unicode is converted to ASCII code when writing a character. After the program is finished, the file will contain eight bytes, each represents an ASCII code. So, the values are

31 32 33 34 35 36 37 38

Note the ASCII code in hex for character 1 is 31.

[▼](javascript:collapse('expandText18',%20'collapseText18');)17.4.12

**After the following program is finished, how many bytes are there in the file t.dat? Show the contents of each byte.**

import java.io.\*;

public class Test {

public static void main(String[] args) throws IOException {

try (DataOutputStream output = new DataOutputStream(

new FileOutputStream("t.dat")); ) {

output.writeInt(1234);

output.writeInt(5678);

output.close();

}

}

}

Each int value takes four bytes. Since two int values are written into the file, the file contains eight bytes. The values are

00 00 04 D2 00 00 16 2E

The first four bytes are for 1234, which equals to 4D2 in hex, and the second byte is for 5678, which equals to 2246 in hex.

[▼](javascript:collapse('expandText19',%20'collapseText19');)17.4.13

**For each of the following statements on a DataOutputStream output, how many bytes are sent to the output?**

output.writeChar('A');

output.writeChars("BC");

output.writeUTF("DEF");

output.writeChar('A'); => 2 bytes   
output.writeChars("BC"); => 4 bytes   
output.writeUTF("DEF"); => 2 + 3 bytes (the first two bytes store the number of characters in the string. Each ASCII character takes one byte in UTF)

[▼](javascript:collapse('expandText20',%20'collapseText20');)17.4.14

**What are the advantages of using buffered streams? Are the following statements correct?**

BufferedInputStream input1 =

new BufferedInputStream(new FileInputStream("t.dat"));

DataInputStream input2 = new DataInputStream(

new BufferedInputStream(new FileInputStream("t.dat")));

DataOutputStream output = new DataOutputStream(

new BufferedOutputStream(new FileOutnputStream("t.dat")));

Since physical input and output involving I/O devices are typically very slow compared with CPU processing speeds, buffered input/output streams can be used to improve performance. You can create a buffered input stream by wrapping a BufferedInputStream/BufferedReader on any instance of InputStream/Reader, and create a buffered output stream by wrapping a BufferedOutputStream/BufferedWriter on any instance of BufferedOutputStream/Writer.

Section 17.5

[▼](javascript:collapse('expandText21',%20'collapseText21');)17.5.1

**How does the program check if a file already exists?**

Create a File object and use its exists() method to test if a file exists.

[▼](javascript:collapse('expandText22',%20'collapseText22');)17.5.2

**How does the program detect the end of the file while reading data?**

If the read() method returns -1, it indicates the end of file.

[▼](javascript:collapse('expandText23',%20'collapseText23');)17.5.3

**How does the program count the number of bytes read from the file?**

Every time a byte is read, numberOfBytesCopied is incremented by 1.

Section 17.6

[▼](javascript:collapse('expandText24',%20'collapseText24');)17.6.1

**What types of objects can be stored using the ObjectOutputStream? What is the method for writing an object? What is the method for reading an object? What is the return type of the method that reads an object from ObjectInputStream?**

Any objects that are instance of Serializable may be stored using the object stream. You use the writeObject method to write an object to the object output stream and use readObject to read an object from the object input stream. The readObject method returns a value of the Object type.

[▼](javascript:collapse('expandText25',%20'collapseText25');)17.6.2

**If you serialize two objects of the same type, will they take the same amount of space? If not, give an example.**

No. For example, two ArrayList objects may have different size, so they are serialized into different sizes. Consider two JButton objects. If one button has an icon, and the other does not, the one with an icon will require more space in the file when it is serialized.

[▼](javascript:collapse('expandText26',%20'collapseText26');)17.6.3

**Is it true that any instance of java.io.Serializable can be successfully serialized? Are the static variables in an object serialized? How do you mark an instance variable not to be serialized?**

An object may not be serialized even though its class implements java.io.Serializable, because it may contain non-serializable instance variables. Implementing java.io.Serializable is a necessary requirement for serialization, but not sufficient. You still have to ensure that all the variables in the object are serializable. A static variable is not serialized. If you don't want a variable to be serialized, mark it transient.

[▼](javascript:collapse('expandText27',%20'collapseText27');)17.6.4

**Can you write an array to an ObjectOutputStream?**

Yes. An array can be serialized if all its elements can be serialized.

[▼](javascript:collapse('expandText28',%20'collapseText28');)17.6.5

**Is it true that DataInputStream/DataOutputStream can always be replaced by ObjectInputStream/ObjectOutputStream?**

Yes. Because ObjectInputStream/ObjectOutputStream contains all features and operations in DataInputStream/DataOutputStream.

[▼](javascript:collapse('expandText29',%20'collapseText29');)17.6.6

**What will happen when you attempt to run the following code?**

import java.io.\*;

public class Test {

public static void main(String[] args) throws IOException {

try ( ObjectOutputStream output =

new ObjectOutputStream(new FileOutputStream("object.dat")); ) {

output.writeObject(new A());

}

}

}

class A implements Serializable {

B b = new B();

}

class B {

}

A java.io.NotSerializableException would occur.

[▼](javascript:collapse('expandText30',%20'collapseText30');)17.7.1

**Can RandomAccessFile streams read and write a data file created by DataOutputStream? Can RandomAccessFile streams read and write objects?**

Yes, because they share the same interface for reading and writing data in the same format. No. Cannot write objects.

[▼](javascript:collapse('expandText31',%20'collapseText31');)17.7.2

**Create a RandomAccessFile stream for the file address.dat to allow the updating of student information in the file. Create a DataOutputStream for the file address.dat. Explain the differences between these two statements.**

RandomAccessFile raf = new RandomAccessFile("address.dat", "rw");

DataOutputStream outfile = new DataOutputStream(

new FileWriter("address.dat"));

To create a RandomAccessFile stream, you simply use the RandomAccessFile constructor. To create a DataOutputStream, you use DataOutputStream wrapped on FileOutputStream.

[▼](javascript:collapse('expandText32',%20'collapseText32');)17.7.3

**What happens if the file test.dat does not exist when you attempt to compile and run the following code?**

import java.io.\*;

public class Test {

public static void main(String[] args) {

try ( RandomAccessFile raf =

new RandomAccessFile("test.dat", "r"); ) {

int i = raf.readInt();

}

catch (IOException ex) {

System.out.println("IO exception");

}

}

}

It will compile fine, but raises a run time exception on invoking readInt() because nothing is in the file.

**Chapter 18 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)18.2.1

**What is a recursive method? What is an infinite recursion?**

A recursive method is the one that calls itself. An infinite recursion is the one that never stops.

[▼](javascript:collapse('expandText1',%20'collapseText1');)18.2.2

**How many times is the factorial method in Listing 18.1 invoked for factorial(6)?**

six times. (base case factorial(0))

[▼](javascript:collapse('expandText2',%20'collapseText2');)18.2.3

**Show the output of the following programs and identify base cases and recursive calls.**

(a)

public class Test {

public static void main(String[] args) {

System.out.println(

"Sum is " + xMethod(5));

}

public static int xMethod(int n) {

if (n == 1)

return 1;

else

return n + xMethod(n - 1);

}

}

(b)

public class Test {

public static void main(String[] args) {

xMethod(1234567);

}

public static void xMethod(int n) {

if (n > 0) {

System.out.print(n % 10);

xMethod(n / 10);

}

}

}

(a) Sum is 15 (5 + 4 + 3 + 2 + 1 = 15)   
(b) 7654321

[▼](javascript:collapse('expandText3',%20'collapseText3');)18.2.4

**Write a recursive mathematical definition for computing 2 n for a positive integer n.**

f(n) = 2 if n = 1

f(n) = 2 \* 2^(n-1) for (n > 1)

[▼](javascript:collapse('expandText4',%20'collapseText4');)18.2.5

**Write a recursive mathematical definition for computing x n for a positive integer n and a real number x.**

f(n) = x if n = 1

f(n) = x \* x^(n-1) for (n > 1)

[▼](javascript:collapse('expandText5',%20'collapseText5');)18.2.6

**Write a recursive mathematical definition for computing 1 + 2 + 3 + ... + n for a positive integer n.**

f(n) = 1 if n = 1

f(n) = f(n-1) + n for (n > 1)

Section 18.3

[▼](javascript:collapse('expandText6',%20'collapseText6');)18.3.1

**Show the output of the following two programs:**

(a)

public class Test {

public static void main(String[] args) {

xMethod(5);

}

public static void xMethod(int n) {

if (n > 0) {

System.out.print(n + " ");

xMethod(n - 1);

}

}

}

(b)

public class Test {

public static void main(String[] args) {

xMethod(5);

}

public static void xMethod(int n) {

if (n > 0) {

xMethod(n - 1);

System.out.print(n + " ");

}

}

}

(a) The output is 5 4 3 2 1   
(b) The output is 1 2 3 4 5

[▼](javascript:collapse('expandText7',%20'collapseText7');)18.3.2

**What is wrong in the following method?**

(a)

public class Test {

public static void main(String[] args) {

xMethod(1234567);

}

public static void xMethod(double n) {

if (n != 0) {

System.out.print(n);

xMethod(n / 10);

}

}

}

(b)

public class Test {

public static void main(String[] args) {

Test test = new Test();

System.out.println(test.toString());

}

public Test() {

Test test = new Test();

}

}

(a) n is double. There is no guarantee that n != 0 will be eventually false.   
(b) Infinite recursion due to new Test() inside the constructor Test().

[▼](javascript:collapse('expandText8',%20'collapseText8');)18.3.3

**How many times is the fib method in Listing 18.2 invoked for fib(6)?**

25 times (Why?   
number of time fib is invoked in fib(0) =   
1   
number of time fib is invoked in fib(1) =   
1   
number of time fib is invoked in fib(2) =   
1+ number of time fib is invoked in fib(1)+number of time fib is invoked in fib(2) =1+1+1=3   
number of time fib is invoked in fib(3) =   
1+ number of time fib is invoked in fib(1)+number of time fib is invoked in fib(2) = 1+1+3=5   
number of time fib is invoked in fib(4) =   
1+ number of time fib is invoked in fib(2)+number of time fib is invoked in fib(3) = 1+3+5=9   
number of time fib is invoked in fib(5) =   
1+ number of time fib is invoked in fib(3)+number of time fib is invoked in fib(4) = 1+5+9=15   
number of time fib is invoked in fib(6) =   
1+ number of time fib is invoked in fib(4)+number of time fib is invoked in fib(5) = 1+9+15=25

Section 18.4

[▼](javascript:collapse('expandText9',%20'collapseText9');)18.4.1

**Describe the characteristics of recursive methods.**

One or more base cases (the simplest case) are used to stop recursion. Every recursive call reduces the original problem, bringing it increasingly close to a base case until it becomes that case.

[▼](javascript:collapse('expandText10',%20'collapseText10');)18.4.2

**For the isPalindrome method in Listing 18.3, what are the base cases? How many times is this method called when invoking isPalindrome("abdxcxdba")?**

The base cases are (1) s.length() <= 1 and (2) s.charAt(0) != s.charAt(s.length - 1)   
When invoking isPalindrome("abdxcxdba"), the isPalindrome method is called 5 times.

[▼](javascript:collapse('expandText11',%20'collapseText11');)18.4.3

Show the call stack for isPalindrome("abcba") using the method defined in Listing 18.3.

Omitted

Section 18.5

[▼](javascript:collapse('expandText12',%20'collapseText12');)18.5.1

Show the call stack for isPalindrome("abcba") using the method defined in Listing 18.4.

Omitted

[▼](javascript:collapse('expandText13',%20'collapseText13');)18.5.2

Show the call stack for selectionSort(new double[]{2, 3, 5, 1}) using the method defined in Listing 18.5.

Omitted

[▼](javascript:collapse('expandText14',%20'collapseText14');)18.5.3

**What is a recursive helper method?**

An overloaded method with additional parameters.

[▼](javascript:collapse('expandText15',%20'collapseText15');)18.6.1

**What is the base case for the getSize method?**

The base case for the getSize(File d) method is that d is a file.

[▼](javascript:collapse('expandText16',%20'collapseText16');)18.6.2

**How does the program get all files and directories under a given directory?**

The program gets all files and directories under the directory d using d.listFiles(), which returns an array of File objects under the directory.

[▼](javascript:collapse('expandText17',%20'collapseText17');)18.6.3

**How many times will the getSize method be invoked for a directory if the directory has three subdirectories and each subdirectory has four files?**

4 times for the directories and 4 \* 4 time for all the files. So, the total is 20.

[▼](javascript:collapse('expandText18',%20'collapseText18');)18.6.4

**Will the program work if the directory is empty (i.e., it does not contain any files)?**

Yes.

[▼](javascript:collapse('expandText19',%20'collapseText19');)18.6.5

**Will the program work if line 20 is replaced by the following code?**

for (int i = 0; i < files.length; i++)

No. The directory may be empty.

[▼](javascript:collapse('expandText20',%20'collapseText20');)18.6.6

**Will the program work if lines 20-21 is replaced by the following code?**

for (File file: files)

size += getSize(file); // Recursive call

No. files may be null.

[▼](javascript:collapse('expandText21',%20'collapseText21');)18.7.1

**How many times is the moveDisks method in Listing 18.8 invoked for moveDisks(5, 'A', 'B', 'C')?**

2^5 - 1

Section 18.8

[▼](javascript:collapse('expandText22',%20'collapseText22');)18.8.1

**How do you obtain the midpoint between two points?**

The midpoint between p1 and p2 is ((p1.x + p2.x)/2, (p1.y + p2.y)/2), which can be obtained by invoking p1.midpoint(p2).

[▼](javascript:collapse('expandText23',%20'collapseText23');)18.8.2

**What is the base case for the displayTriangles method?**

The base case for the displayTriangles method is order == 0.

[▼](javascript:collapse('expandText24',%20'collapseText24');)18.8.3

**How many times is the displayTriangles method invoked for a Sierpinski triangle of order 0, order 1, order 2, and order n?**

The displayTriangles method is invoked one time for order 1, 4 times for order 1, 1 + 3 \* 3 times for order 2, and 1 + 3^n for order n.

[▼](javascript:collapse('expandText25',%20'collapseText25');)18.8.4

**What happens if you enter a negative order? How do you fix this problem in the code?**

Will be an infinite loop. To fix it, add if (order < 0) return in the beginning of the method displayTriangle.

[▼](javascript:collapse('expandText26',%20'collapseText26');)18.8.5

**Instead of drawing a triangle using a polygon, rewrite the code to draw a triangle by drawing three lines to connect the points in lines 71-77.**

**Replace lines 71-77 with the following code:**

// Draw a triangle to connect three points

Line line1 = new Line(p1.getX(), p1.getY(), p2.getX(), p2.getY());

Line line2 = new Line(p2.getX(), p2.getY(), p3.getX(), p3.getY());

Line line3 = new Line(p3.getX(), p3.getY(), p1.getX(), p1.getY());

this.getChildren().addAll(line1, line2, line3);

Section 18.9

[▼](javascript:collapse('expandText27',%20'collapseText27');)18.9.1

**Which of the following statements are true?**a. Any recursive method can be converted into a nonrecursive method.   
b. Recursive methods take more time and memory to execute than nonrecursive methods.   
c. Recursive methods are always simpler than nonrecursive methods.   
d. There is always a selection statement in a recursive method to check whether a base case is reached.

a. (TRUE)   
b. (TRUE)   
c. (FALSE)   
d. (TRUE)

[▼](javascript:collapse('expandText28',%20'collapseText28');)18.9.2

**What is a cause for a stack-overflow exception?**

When a method is invoked, its contents are placed into a stack. If a method is recursively invoked, it is possible that the stack space is exhausted. This causes stack overflow.

Section 18.10

[▼](javascript:collapse('expandText29',%20'collapseText29');)18.10.1

**Identify tail-recursive methods in this chapter.**

The isPalindrome method in Listing 18.4, sort method in Listing 18.5, and binarySearch method in Listing 18.6 are tail-recursive.

[▼](javascript:collapse('expandText30',%20'collapseText30');)18.10.2

**Rewrite the fib method in Listing 18.2 using tail recursion.**

/\*\* Return the Fibonacci number for the specified index \*/

public static long fib(long index) {

return fib(index, 1, 0);

}

/\*\* Auxiliary tail-recursive method for fib \*/

private static int fib(long index, int next, int result) {

if (index == 0)

return result;

else

return fib(index - 1, next + result, next);

}

**Chapter 19 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)19.2.1

**Are there any compile errors in (a) and (b)?**

(a) Prior to JDK 1.5

ArrayList dates = new ArrayList();

dates.add(new Date());

dates.add(new String());

(b) Since JDK 1.5

ArrayList<Date> dates = new ArrayList<>();

dates.add(new Date());

dates.add(new String());

(a) will compile fine, but (b) has a compilation error on Line 3, because dates is declared as a list of Date objects. You cannot assign a string to the list.

[▼](javascript:collapse('expandText1',%20'collapseText1');)19.2.2

**What is wrong in (a)? Is the code in (b) correct?**

(a) Prior to JDK 1.5

ArrayList dates = new ArrayList();

dates.add(new Date());

Date date = dates.get(0);

(b) Since JDK 1.5

ArrayList<Date> dates = new ArrayList<>();

dates.add(new Date());

Date date = dates.get(0);

Casting is needed in (a), but no casting is necessary in (b) with the generic type ArrayList<Date> .

[▼](javascript:collapse('expandText2',%20'collapseText2');)19.2.3

**What are the benefits of using generic types?**

One important benefit is improving reliability and robustness. Potential errors can be detected by the compiler.

Section 19.3

[▼](javascript:collapse('expandText3',%20'collapseText3');)19.3.1

**What is the generic definition for java.lang.Comparable in the Java API?**

package java.lang;

public interface Comparable<E> {

public int compareTo(E o) { }

}

[▼](javascript:collapse('expandText4',%20'collapseText4');)19.3.2

**Since you create an instance of ArrayList of strings using new ArrayList<String>(), should the constructor in the ArrayList class be defined as public ArrayList<E>()**

No.

[▼](javascript:collapse('expandText5',%20'collapseText5');)19.3.3

**Can a generic class have multiple generic parameters?**

Yes.

[▼](javascript:collapse('expandText6',%20'collapseText6');)19.3.4

**How do you declare a generic type in a class?**

To declare a generic type for a class, place the generic type after the class name, such as GenericStack<E>. To declare a generic type for a method, place the generic type for the method return type, such as <E> void max(E o1, E o2).

Section 19.4

[▼](javascript:collapse('expandText7',%20'collapseText7');)19.4.1

**How do you declare a generic method? How do you invoke a generic method?**

To declare a generic method, you place the generic type <E> immediately after the keyword static in the method. A generic method can be invoked just like a regular method. The compiler automatically discovers the actual type.

[▼](javascript:collapse('expandText8',%20'collapseText8');)19.4.2

**What is a bounded generic type?**

Bounded generic type such as <E extends AClass> specifies that a generic type must be a subclass of AClass.

Section 19.5

[▼](javascript:collapse('expandText9',%20'collapseText9');)19.5.1

**Given int[] list = {1, 2, -1}, can you invoke sort(list) using the sort method in Listing 19.4?**

No, because list is of type int[], but the sort method requires E[], where E is an object type.

[▼](javascript:collapse('expandText10',%20'collapseText10');)19.5.2

**Given int[] list = {new Integer(1), new Integer(2), new Integer(-1)}, can you invoke sort(list) using the sort method in Listing 19.4?**

No, because list is still of type int[], but the sort method requires E[], where E is an object type.

Section 19.6

[▼](javascript:collapse('expandText11',%20'collapseText11');)19.6.1

**What is a raw type? Why is a raw type unsafe? Why is the raw type allowed in Java?**

When you use generic type without specifying an actual parameter, it is called a raw type. A raw type is unsafe, because some errors cannot be detected by the compiler. The raw type is allowed in Java for backward compatibility.

[▼](javascript:collapse('expandText12',%20'collapseText12');)19.6.2

**What is the syntax to declare an ArrayList reference variable using the raw type and assign a raw type ArrayList object to it?**

ArrayList list = new ArrayList();

Section 19.7

[▼](javascript:collapse('expandText13',%20'collapseText13');)19.7.1

**Is GenericStack the same as GenericStack<Object>?**

GenericStack is roughly equivalent to GenericStack<Object>, but they are not the same. GenericStack<Object> is a generic instantiation, but GenericStack is a raw type.

[▼](javascript:collapse('expandText14',%20'collapseText14');)19.7.2

**What are an unbounded wildcard, a bounded wildcard, and a lower-bound wildcard?**

? is unbounded wildcard

? extends T is bounded wildcard

? super T is lower bounded wildcard

[▼](javascript:collapse('expandText15',%20'collapseText15');)19.7.3

**What happens if lines 12-13 in Listing 19.9 are changed to**

public static <T> void add(GenericStack<T> stack1,

GenericStack<T> stack2)

The program cannot be compiled, because the element type in stack1 is GenericStack<String>, but the element type is stack2 is GenericStack<Object>. add(stack1, stack2) cannot be matched.

[▼](javascript:collapse('expandText16',%20'collapseText16');)19.7.4

**What happens if lines 12-13 in Listing 19.9 are changed to**

public static <T> void add(GenericStack<? extends T> stack1,

GenericStack<T> stack2)

The program can be compiled and run fine.

Section 19.8

[▼](javascript:collapse('expandText17',%20'collapseText17');)19.8.1

**What is erasure? Why are Java generics implemented using erasure?**

Generic type information is used by the compiler to check whether the type is used safely. Afterwards the type information is erased. The type information is not available at runtime. This approach enables the generic code to be backward-compatible with the legacy code that uses raw types.

[▼](javascript:collapse('expandText18',%20'collapseText18');)19.8.2

**If your program uses ArrayList<String> and ArrayList<Date> , does the JVM load both of them?**

No. Only ArrayList is loaded.

[▼](javascript:collapse('expandText19',%20'collapseText19');)19.8.3

**Can you create an instance using new E() for a generic type E? Why?**

No, because the type information is not available at runtime.

[▼](javascript:collapse('expandText20',%20'collapseText20');)19.8.4

**Can a method that uses a generic class parameter be static? Why?**

Since all instances of a generic class have the same runtime class, the static variables and methods of a generic class is shared by all its instances. Therefore, it is illegal to refer a generic type parameter for a class in a static method or initializer.

[▼](javascript:collapse('expandText21',%20'collapseText21');)19.8.5

**Can you define a custom generic exception class? Why?**

No. The JVM have to check the exception thrown from the try clause to see if it matches the type specified in a catch clause. This is impossible, because the type information is not present at runtime.

Section 19.9

[▼](javascript:collapse('expandText22',%20'collapseText22');)19.9.1

**Why are the add, multiple, and zero methods defined abstract in the GenericMatrix class?**

Because these methods cannot be implemented in the GenericMatric class.

[▼](javascript:collapse('expandText23',%20'collapseText23');)19.9.2

**How are the add, multiple, and zero methods implemented in the IntegerMatrix class?**

In the IntegerMatrix class, the add method is implemented by adding the two numbers using the + operator. The multiply method is implemented by multiplying the two numbers using the \* operator. The zero method is implemented to return 0.

[▼](javascript:collapse('expandText24',%20'collapseText24');)19.9.3

**How are the add, multiple, and zero methods implemented in the RationalMatrix class?**

In the RationalMatrix class, the add method is implemented by adding the two numbers using the add method in the Rational class. The multiply method is implemented by multiplying the two numbers using the multiply method in the Rational class. The zero method is implemented to return new Rational(0, 1).

[▼](javascript:collapse('expandText25',%20'collapseText25');)19.9.4

**What would be wrong if the printResult method defined as follows?**

public static void printResult(

E[][] m1, E[][] m2, E[][] m3, char op)

You have to define it using:

public static <T> void printResult(

T[][] m1, T[][] m2, T[][] m3, char op)

**Chapter 20 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)20.2.1

**What is a data structure?**

A data structure is a collection of data organized in some fashion.In object-oriented thinking, a data structure is an object that stores other objects, referred to as data or elements. So some people refer a data structure as a container object or a collection object. To define a data structure is essentially to declare a class.

[▼](javascript:collapse('expandText1',%20'collapseText1');)20.2.2

**Describe the Java Collections Framework. List the interfaces, convenience abstract classes, and concrete classes under the Collection interface.**

The Java Collections Framework defines the Java API for handling common data structures tasks in Java. It defines classes and interfaces for storing and manipulating data in sets, lists, and maps.  
A convenience class is an abstract class that partially implements an interface. The Java Collections Framework defines interfaces, convenience abstract classes, and concrete classes.

[▼](javascript:collapse('expandText2',%20'collapseText2');)20.2.3

**Can a collection object be cloned and serialized?**

Yes. The concrete classes of Set, List, and Map implements the clone() method in the Cloneable interface.

[▼](javascript:collapse('expandText3',%20'collapseText3');)20.2.4

**What method do you use to add all the elements from one collection to another collection?**

addAll(Collection c).

[▼](javascript:collapse('expandText4',%20'collapseText4');)20.2.5

**When should a method throw an UnsupportedOperationException?**

If a method has no meaning in the subclass, you can implement it in the subclass to throw java.lang.UnsupportedOperationException, a subclass of RuntimeException. This is a good design that you can use in your project. If a method has no meaning in the subclass, you can implement it as follows:

public void someMethod() {

throw new UnsupportedOperationException

("Method not supported");

}

Section 20.3

[▼](javascript:collapse('expandText5',%20'collapseText5');)20.3.1

**How do you obtain an iterator from a collection object?**

The Collection interface extends the Iterable interface. You can obtain an iterator from a collection using the iterator() method.

[▼](javascript:collapse('expandText6',%20'collapseText6');)20.3.2

**What method do you use to obtain an element in the collection from an iterator?**

Use the next() method.

[▼](javascript:collapse('expandText7',%20'collapseText7');)20.3.3

**Can you use a foreach loop to traverse the elements in any instance of Collection?**

Yes.

[▼](javascript:collapse('expandText8',%20'collapseText8');)20.3.4

**When using a foreach loop to traverse all elements in a collection, do you need to use the next() or hasNext() methods in an iterator?**

No. They are implicitly used in a foreach loop.

Section 20.4

[▼](javascript:collapse('expandText9',%20'collapseText9');)20.4.1

**Can you use the forEach method on any instance of Collection? Where is the forEach method defined?**

Yes. It is defined in the Iterable interface which is a super interface for Collection.

[▼](javascript:collapse('expandText10',%20'collapseText10');)20.4.2

**Suppose each element in list is a StringBuilder, write a statement using a forEach method to change the first character to uppercase for each element in list.**

list.forEach(e -> {

if (((StringBuilder)e).length() > 0) {

char ch = ((StringBuilder)e).charAt(0);

if (Character.isLowerCase(ch)) {

((StringBuilder)e).setCharAt(0, Character.toUpperCase(ch));

};

};

});

Section 20.5

[▼](javascript:collapse('expandText11',%20'collapseText11');)20.5.1

**How do you add and remove elements from a list? How do you traverse a list in both directions?**

Use the add or remove method to add or remove elements from a list. Use the listIterator() to obtain an iterator. This iterator allows you to traverse the list bi-directional.

[▼](javascript:collapse('expandText12',%20'collapseText12');)20.5.2

**Suppose that list1 is a list that contains the strings red, yellow, and green, and that list2 is another list that contains the strings red, yellow, and blue.** Answer the following questions:   
a. What are list1 and list2 after executing list1.addAll(list2)?   
b. What are list1 and list2 after executing list1.add(list2)?   
c. What are list1 and list2 after executing list1.removeAll(list2)?   
d. What are list1 and list2 after executing list1.remove(list2)?   
e. What are list1 and list2 after executing list1.retainAll(list2)?   
f. What is list1 after executing list1.clear()?

list2 is not changed by all these methods.   
a. list1 is [red, yellow, green, red, yellow, blue]   
b. list1 is [red, yellow, green, [red, yellow, blue]]   
c. list1 is [green] What is list1 and list2 after executing list1.remove(list2);   
d. list1 is [red, yellow, green]   
e. list1 is [red, yellow] What is list1 after executing list1.clear();   
f. list1 is empty

[▼](javascript:collapse('expandText13',%20'collapseText13');)20.5.3

**What are the differences between ArrayList and LinkedList? Which list should you use to insert and delete elements at the beginning of a list?**

ArrayList and LinkedList can be operated similarly. The critical differences between them are their internal implementation, which impacts the performance. ArrayList is efficient for retrieving elements, and for adding and removing elements from the end of the list. LinkedList is efficient for adding and removing elements anywhere in the list.

[▼](javascript:collapse('expandText14',%20'collapseText14');)20.5.4

**Are all the methods in ArrayList also in LinkedList? What methods are in LinkedList but not in ArrayList?**

All the methods in ArrayList are also in LinkedList except the trimToSize() method. The methods getFirst, getLast, addFirst, addLast are in LinkedList, but not in ArrayList.

[▼](javascript:collapse('expandText15',%20'collapseText15');)20.5.5

**How do you create a list from an array of objects?**

A simple way to create a list from an array of objects is to use

new ArrayList(Arrays.asList(arrayObject))

or

new LinkedList(Arrays.asList(arrayObject)).

Section 20.6

[▼](javascript:collapse('expandText16',%20'collapseText16');)20.6.1

**What are the differences between the Comparable interface and the Comparator interface? In which package is Comparable, and in which package is Comparator?**

The Comparable interface contains the compareTo method and Comparator interface contains the compare method and equals method. Normally, if the objects of a class have natural order (e.g., String, Date), let the class implement the Comparable interface. The Comparator interface is more flexible in the sense that it enables you to define a new class that contains the compare(Object, Object) method to compare two objects of other classes.   
The Comparable interface is in the java.lang package, and the Comparator interface is in the java.util package.

[▼](javascript:collapse('expandText17',%20'collapseText17');)20.6.2

**How do you define a class A that implements the Comparable interface? Are two instances of class A comparable? How do you define a class B that implements the Comparator interface and override the compare method to compare two objects of type B1? How do you invoke the sort method to sort a list of objects of the type B1 using a comparator?**

How do you define a class A that implements the Comparable interface?

public class A implements Comparable<A> {

public int compareTo(A o) {

return an integer;

}

}

**Are two instances of class A comparable? Yes.How do you define a class B that implements the Comparator interface and override the compare method to compare two objects of type B1?**

public class B implements Comparator<B1> {

public int compare(B1 o1, B1 o2) {

return an integer;

}

}

**How do you invoke the sort method to sort a list of objects of the type B1?**

list.sort(new B());

To sort an array x of objects of the type B1, use

java.util.Arrays.sort(x, new B());

[▼](javascript:collapse('expandText18',%20'collapseText18');)20.6.3

**Write a lambda expression to create a comparator that compares two Loan objects by their annualInterestRate. Create a comparator using the Comparator.comparing method to compare Loan objects on annualInterestRate. Create a comparator to compare Loan objects first on annualInterestRate then on loanAmount.**

(e1, e2) -> e1.getAnnualInterestRate() < e2.getAnnualInterestRate() ? -1 :

e1.getAnnualInterestRate() == e2.getAnnualInterestRate() ? 0 : 1

Comparator.comparing(Loan::getAnnualInterestRate);

Comparator.comparing(Loan::getAnnualInterestRate)

.thenComparing(Loan:getLoanAmount);

[▼](javascript:collapse('expandText19',%20'collapseText19');)20.6.4

**Create a comparator using a lambda expression and using the Comparator.comparing method, respectively, to compare Collection objects on their size.**

(e1, e2) -> e1.size() - e2.size()

Comparator.comparing(Collection::size)

[▼](javascript:collapse('expandText20',%20'collapseText20');)20.6.5

**Write a statement that sorts an array named points of Point2D objects on their y values and then on their x values.**

java.util.sort(points,

Comparator.comparing(Point2D::x).thenComparing(Point2D::y));

[▼](javascript:collapse('expandText21',%20'collapseText21');)20.6.6

**Write a statement that sorts an ArrayList of strings named list in increasing order of their last character.**

list.sort((e1, e2) -> {

if (e1.length() == 0)

return -1;

else (e2.length() == 0)

return 1;

else

return charAt(e1.size() - 1) - charAt(e2.size() - 1);

}

[▼](javascript:collapse('expandText22',%20'collapseText22');)20.6.7

**Write a statement that sorts a two-dimensional array of double[][] in increasing order of their second column. For example, if the array is double[][] x** = {{3, 1}, {2, -1}, {2, 0}}, the sorted array will be {{2, -1}, {2, 0}, {3, 1}}.

java.util.Arrays.sort(x, (e1, e2) -> (int)(e1[1] - e2[1]));

[▼](javascript:collapse('expandText23',%20'collapseText23');)20.6.8

**Write a statement that sorts a two-dimensional array of double[][] in increasing order of their second column as the primary order and the first column as the secondary order. For example, if the array is double[][] x = {{3, 1}, {2, -1}, {2, 0}, {1, -1}}, the sorted array will be {{1, -1}, {2, -1}, {2, 0}, {3, 1}}.**

java.util.Arrays.sort(x, (e1, e2) -> {

if (e1[1] - e2[1] != 0)

return (int)(e1[1] - e2[1]);

else

return (int)(e1[0] - e2[0]);

});

[▼](javascript:collapse('expandText24',%20'collapseText24');)20.7.1

**Are all the methods in the Collections class static?**

Yes.

[▼](javascript:collapse('expandText25',%20'collapseText25');)20.7.2

**Which of the following static methods in the Collections class are for lists, and which are for collections?**

sort, binarySearch, reverse, shuffle, max, min, disjoint, frequency

The methods for lists are: sort, binarySearch, reverse, shuffle   
The methods for collections are: max, min, disjoint, frequency   
Note that all the methods for collections are also for lists, because lists are collections.

[▼](javascript:collapse('expandText26',%20'collapseText26');)20.7.3

**Show the output of the following code:**

import java.util.\*;

public class Test {

public static void main(String[] args) {

List<String> list =

Arrays.asList("yellow", "red", "green", "blue");

Collections.reverse(list);

System.out.println(list);

List<String> list1 =

Arrays.asList("yellow", "red", "green", "blue");

List<String> list2 = Arrays.asList("white", "black");

Collections.copy(list1, list2);

System.out.println(list1);

Collection<String> c1 = Arrays.asList("red", "cyan");

Collection<String> c2 = Arrays.asList("red", "blue");

Collection<String> c3 = Arrays.asList("pink", "tan");

System.out.println(Collections.disjoint(c1, c2));

System.out.println(Collections.disjoint(c1, c3));

Collection<String> collection =

Arrays.asList("red", "cyan", "red");

System.out.println(Collections.frequency(collection, "red"));

}

}

[blue, green, red, yellow]

[white, black, green, blue]

false

true

2

[▼](javascript:collapse('expandText27',%20'collapseText27');)20.7.4

**Which method can you use to sort the elements in an ArrayList or a LinkedList? Which method can you use to sort an array of strings?**

You can use Collections.sort(list) to sort an ArrayList or a LinkedList and use Arrays.sort(Object[]) to sort an array of strings. For example,

LinkedList<String> list = new LinkedList<>();

list.add("Java"); list.add("Python"); list.add("C++");

java.util.Collections.sort(list); // Sort the list

String[] languages = {"Java", "Python", "C++"};

java.util.Arrays.sort(languages); // Sort the array

[▼](javascript:collapse('expandText28',%20'collapseText28');)20.7.5

**Which method can you use to perform binary search for elements in an ArrayList or a LinkedList? Which method can you use to perform binary search for an array of strings?**

You can use Collections.binary(list, key) to perform binary search for an ArrayList or a LinkedList and use Arrays.binary(Object[], key) to sort an array of strings.

[▼](javascript:collapse('expandText29',%20'collapseText29');)20.7.6

**Write a statement to find the largest element in an array of comparable objects.**

Collections.max(Arrays.asList(arrayObject))

Section 20.8

[▼](javascript:collapse('expandText30',%20'collapseText30');)20.8.1

**What is the return value from invoking pane.getChildren() for a pane?**

The return value is an ObservableList<Node>, which is a subtype of List<Node>.

[▼](javascript:collapse('expandText31',%20'collapseText31');)20.8.2

**How do you modify the code in the MutilpleBallApp program to remove the first ball in the list when the button is clicked?**

**Replace line 75 with the following code:**

getChildren().remove(getChildren().size() - 1);

[▼](javascript:collapse('expandText32',%20'collapseText32');)20.8.3

**How do you modify the code in the MutilpleBallApp program so that each ball will get a random radius between 10 and 20?**

Change line 133 to

radius = Math.random\*11 + 10;

Section 20.9

[▼](javascript:collapse('expandText33',%20'collapseText33');)20.9.1

**How do you create an instance of Vector? How do you add or insert a new element into a vector? How do you remove an element from a vector? How do you find the size of a vector?**

Vector is the same as ArrayList except that, except that Vector contains the synchronized methods for accessing and modifying the vector. Since Vector implements List, you can use the methods in List to add, remove elements from a vector, and use the size() method to find the size of a vector. To create a vector, use either its constructors.

[▼](javascript:collapse('expandText34',%20'collapseText34');)20.9.2

**How do you create an instance of Stack? How do you add a new element to a stack? How do you remove an element from a stack? How do you find the size of a stack?**

Stack is a subclass of Vector. The Stack class represents a last-in-first-out stack of objects. The elements are accessed only from the top of the stack. You can retrieve, insert, or remove an element from the top of the stack. To add a new element to a stack, use the push method. To remove an element from the top of the stack, use the method pop. To find a stack size, use the size() method.

[▼](javascript:collapse('expandText35',%20'collapseText35');)20.9.3

**Does Listing 20.1, TestCollection.java, compile and run if all the occurrences of ArrayList are replaced by LinkedList, Vector, or Stack?**

Yes, because these classes are subtypes of the Collection interface.

Section 20.10

[▼](javascript:collapse('expandText36',%20'collapseText36');)20.10.1

**Is java.util.Queue a subinterface of java.util.Collection, java.util.Set, or java.util.List? Does LinkedList implement Queue?**

java.util.Queue is a subinterface of java.util.Collection, and LinkedList implements Queue.

[▼](javascript:collapse('expandText37',%20'collapseText37');)20.10.2

**How do you create a priority queue for integers? By default, how are elements ordered in a priority queue? Is the element with the least value assigned the highest priority in a priority queue?**

Use the constructors of PriorityQueue to create priority queues. By default, the elements in a priority queue are ordered in their natural order using the compareTo method in the Comparable interface. The element with the least value is assigned the highest priority in PriorityQueue.

[▼](javascript:collapse('expandText38',%20'collapseText38');)20.10.3

**How do you create a priority queue that reverses the natural order of the elements?**

new PriorityQueue(initialCapacity, Collections.reverseOrder()).

Section 20.11

[▼](javascript:collapse('expandText39',%20'collapseText39');)20.11.1

**Can the EvaluateExpression program evaluate the following expressions "1+2", "1 + 2", "(1) + 2", "((1)) + 2", and "(1 + 2)"?**

Yes.

[▼](javascript:collapse('expandText40',%20'collapseText40');)20.11.2

Show the change of the contents in the stacks when evaluating "3 + (4 + 5) \* (3 + 5) + 4 \* 5" using the EvaluateExpression program.

Omitted.

[▼](javascript:collapse('expandText41',%20'collapseText41');)20.11.3

**If you enter an expression "4 + 5 5 5", the program will display 10. How do you fix this problem?**

You can fix this problem by throwing an exception if operandStack is not empty after poping the result out of the operandStack stack.

**Chapter 21 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)21.2.1

**How do you create an instance of Set? How do you insert a new element in a set? How do you remove an element from a set? How do you find the size of a set?**

The Set is an interface. To create an instance of Set, you need to use HashSet or TreeSet. To insert an element to a set, use the add method. To remove an element from a set, use the remove method. To find the size of a set, use the size() method.

[▼](javascript:collapse('expandText1',%20'collapseText1');)21.2.2

**If two objects o1 and o2 are equal, what is o1.equals(o2) and o1.hashCode() == o2.hashCode()?**

If two objects o1 and o2 are equal, o1.equals(o2) is true and o1.hashCode() == o1.hashCode() is true.

[▼](javascript:collapse('expandText2',%20'collapseText2');)21.2.3

**What are the differences between HashSet, LinkedHashSet, and TreeSet?**

HashSet is unsorted, but TreeSet is sorted. HashSet is more efficient than TreeSet if you don't want the elements in a set to be sorted. If you create a TreeSet using its default constructor, the compareTo method is used to compare the elements in the set, assuming that the class of the elements implements the Comparable interface. To use a comparator, you have to use the constructor TreeSet(Comparator comparator) to create a sorted set that uses the compare method in the comparator to order the elements in the set. A runtime error would occur if you add an element that cannot be compared with the existing elements in the tree set?

[▼](javascript:collapse('expandText3',%20'collapseText3');)21.2.4

**How do you traverse the elements in a set?**

To traverse a set, use the iterator method to obtain an iterator. You can then traverse the set through the iterator. Using an iterator, you can traverse only sequentially from the beginning to the end.

[▼](javascript:collapse('expandText4',%20'collapseText4');)21.2.5

**How do you sort the elements in a set using the compareTo method in the Comparable interface? How do you sort the elements in a set using the Comparator interface? What would happen if you added an element that could not be compared with the existing elements in a tree set? How does the TreeeSet test if two elements are equal?**

To sort the elements in a set using the Comparable interface, there are two ways: (1) create a TreeSet using new TreeSet(), (2) create a TreeSet from a set using new TreeSet(set). To sort the elements in a set using the Comparator interface, create a TreeSet using new TreeSet(Comparator), then add the elements to the tree set.   
How does the TreeeSet test if two elements are equal? two elements are considered equal in a TreeSet if e1.compare(e2) is 0 using the comparator. If two elements are equal, only one can be stored in a TreeSet. Note that two elements e1 and e2 in a HashSet and LinkedHashSet are considered equal, if e1.equals(e2) is true and their hashCode() are the same.

[▼](javascript:collapse('expandText5',%20'collapseText5');)21.2.6

**Suppose that set1 is a set that contains the strings red, yellow, and green, and that set2 is another set that contains the strings red, yellow, and blue.** Answer the following questions:  
(a) What are in set1 and set2 after executing set1.addAll(set2)?   
(b) What are in set1 and set2 after executing set1.add(set2)?   
(c) What are in set1 and set2 after executing set1.removeAll(set2)?   
(d) What are in set1 and set2 after executing set1.remove(set2)?   
(e) What are in set1 and set2 after executing set1.retainAll(set2)?   
(f) What is in set1 after executing set1.clear()?

set2 is not changed by all these methods. (If you use HashSet, the order of the elements in the set is unpredicatable.)   
**What is set1 and set2 after executing set?;**set1 is [green, red, blue, yellow]   
What is set1 and set2 after executing set1.add(set2)?   
set1 is [green, red, yellow, [red, blue, yellow]] Note set2 is added as a whole element into set1.   
**What is set1 and set2 after executing set1.removeAll(set2)?**set1 is [green]   
**What is set1 and set2 after executing set1.remove(set2)?**set1 is [green, red, yellow]   
**What is set1 and set2 after executing set1.retainAll(set2)?**set1 is [red, yellow] What is set1 after executing set1.clear();

[▼](javascript:collapse('expandText6',%20'collapseText6');)21.2.7

**Show the output of the following code:**

import java.util.\*;

public class Test {

public static void main(String[] args) {

LinkedHashSet<String> set1 = new LinkedHashSet<>();

set1.add("New York");

LinkedHashSet<String> set2 = set1;

LinkedHashSet<String> set3 =

(LinkedHashSet<String>)(set1.clone());

set1.add("Atlanta");

System.out.println("set1 is " + set1);

System.out.println("set2 is " + set2);

System.out.println("set3 is " + set3);

set1.forEach(e -> System.out.print(e + " "));

}

}

set1 is [New York, Atlanta]   
set2 is [New York, Atlanta]   
set3 is [New York]   
New York Atlanta

[▼](javascript:collapse('expandText7',%20'collapseText7');)21.2.8

**Show the output of the following code:**

Set<String> set = new LinkedHashSet<>();

set.add("ABC");

set.add("ABD");

System.out.println(set);

[ABC, ABD]

[▼](javascript:collapse('expandText8',%20'collapseText8');)21.2.9

**What will the output be if lines 6-7 in Listing 21.5 is replaced by the following code:**

Set<GeometricObject> set = new HashSet<>();

It will display the area of four geometric objects in random order. Note that the eqauls method is not overidden in the Circle class. Therefore, both new Circle(40) in lines 9-10 are stored in the hash set.

[▼](javascript:collapse('expandText9',%20'collapseText9');)21.2.10

**Show the output of the following code:**

Set<String> set = new TreeSet<>(

Comparator.comparing(String::length));

set.add("ABC");

set.add("ABD");

System.out.println(set);

[ABC]

[▼](javascript:collapse('expandText10',%20'collapseText10');)21.2.11

**(This is not in the printed book) Show the output of the following code:**

import java.util.\*;

import java.io.\*;

public class Test {

public static void main(String[] args) throws Exception {

ObjectOutputStream output = new ObjectOutputStream(

new FileOutputStream("c:\\test.dat"));

LinkedHashSet<String> set1 = new LinkedHashSet<>();

set1.add("New York");

LinkedHashSet<String> set2 =

(LinkedHashSet<String>)set1.clone();

set1.add("Atlanta");

output.writeObject(set1);

output.writeObject(set2);

output.close();

ObjectInputStream input = new ObjectInputStream(

new FileInputStream("c:\\test.dat"));

set1 = (LinkedHashSet<String>)input.readObject();

set2 = (LinkedHashSet<String>)input.readObject();

System.out.println(set1);

System.out.println(set2);

input.close();

}

}

[New York, Atlanta]   
[New York]

Section 21.3

[▼](javascript:collapse('expandText11',%20'collapseText11');)21.3.1

**Suppose you need to write a program that stores unordered, non-duplicate elements, what data structure should you use?**

Suppose you need to write a program that stores non-duplicate elements, you should use a HashSet.

[▼](javascript:collapse('expandText12',%20'collapseText12');)21.3.2

**Suppose you need to write a program that stores non-duplicate elements in the order of insertion, what data structure should you use?**

Use a LinkedHashSet.

[▼](javascript:collapse('expandText13',%20'collapseText13');)21.3.3

**Suppose you need to write a program that stores non-duplicate elements in increasing order of the element values, what data structure should you use?**

Use a TreeSet.

[▼](javascript:collapse('expandText14',%20'collapseText14');)21.3.4

**Suppose you need to write a program that stores a fixed number of the elements (possibly duplicates), what data structure should you use?**

Use an array.

[▼](javascript:collapse('expandText15',%20'collapseText15');)21.3.5

**Suppose you need to write a program that stores the elements in a list with frequent operations to append and delete elements at the end of the list, what data structure should you use?**

Use an ArrayList.

[▼](javascript:collapse('expandText16',%20'collapseText16');)21.3.6

**Suppose you need to write a program that stores the elements in a list with frequent operations to insert and delete elements at the beginning of the list, what data structure should you use?**

Use a LinkedList.

Section 21.4

[▼](javascript:collapse('expandText17',%20'collapseText17');)21.4.1

**Will the CountKeywords program work if lines 33-34 are changed to**

Set<String> keywordSet =

new LinkedHashSet<>(Arrays.asList(keywordString));

Yes.

[▼](javascript:collapse('expandText18',%20'collapseText18');)21.4.2

**Will the CountKeywords program work if lines 33-34 are changed to**

List<String> keywordSet =

new ArrayList<String>(Arrays.asList(keywordString));

Yes.

Section 21.5

[▼](javascript:collapse('expandText19',%20'collapseText19');)21.5.1

**How do you create an instance of Map? How do you add an entry to a map consisting of a key and a value? How do you remove an entry from a map? How do you find the size of a map? How do you traverse entries in a map?**

Map is an interface. To create an instance of Map, you need to use the HashMap class or the TreeMap class. The HashMap and TreeMap have various constructors that you can use to create a HashMap or a TreeMap. You can use the put method to add an entry to a map, and the remove method to remove an entry with the specified key from the map. Use the size method to find the size of a map.

[▼](javascript:collapse('expandText20',%20'collapseText20');)21.5.2

**Describe and compare HashMap, LinkedHashMap, and TreeMap.**

The HashMap, LinkedHashMap, and TreeMap classes are three concrete implementations of the Map interface. The HashMap class is efficient for locating a value, inserting a mapping, and deleting a mapping. The entries in a HashMap are not ordered, but the entries in a LinkedHashMap can be retrieved in the order in which they were inserted into the map (known as the insertion order), or the order in which they were last accessed, from least recently accessed to most recently (access order). The TreeMap class, implementing SortedMap, is efficient for traversing the keys in a sorted order.

[▼](javascript:collapse('expandText21',%20'collapseText21');)21.5.3

**Show the printout of the following code:**

import java.util.\*;

public class Test {

public static void main(String[] args) {

Map<Integer, String> map = new LinkedHashMap<>();

map.put("123", "John Smith");

map.put("111", "George Smith");

map.put("123", "Steve Yao");

map.put("222", "Steve Yao");

System.out.println("(1) " + map);

System.out.println("(2) " + new TreeMap<, String>(map));

map.forEach((k, v) -> {

if (k.equals("123")) System.out.println(v);});

}

}

(1) {123=Steve Yao, 111=George Smith, 222=Steve Yao}   
(2) {111=George Smith, 123=Steve Yao, 222=Steve Yao}

[▼](javascript:collapse('expandText22',%20'collapseText22');)21.6.1

**Will the CountOccurrenceOfWords program work if line 10 is changed to**

Map<String, int> map = new TreeMap<>();

No. A concrete type must of an object type.

[▼](javascript:collapse('expandText23',%20'collapseText23');)21.6.2

**Will the CountOccurrenceOfWords program work if line 17 is changed to**

if (map.get(key) == null) {

Yes. It will work.

[▼](javascript:collapse('expandText24',%20'collapseText24');)21.6.3

**Will the CountOccurrenceOfWords program work if lines 32-33 are changed to**

for (String key: map)

System.out.println(key + "\t" + map.getValue(key));

No. You cannot iterate on keys in a map in Java.

[▼](javascript:collapse('expandText25',%20'collapseText25');)21.6.4

**Replace the code in lines 17-24 in one line of code using a conditional expression.**

map.put(key, map.containsKey(key) ? map.get(key) + 1 : 1);

Section 21.7

[▼](javascript:collapse('expandText26',%20'collapseText26');)21.7.1

**What is wrong in the following code?**

Set<String> set = Collections.singleton("Chicago");

set.add("Dallas");

The singleton set is immutable. You cannot add an element into the singleton set.

[▼](javascript:collapse('expandText27',%20'collapseText27');)21.7.2

**What happens when you run the following code?**

List<String> list = Collections.unmodifiableList(

Arrays.asList("Chicago", "Boston"));

list.remove("Dallas");

The list created from the unmodifiableList method is immutable is an unmodifiable view of the list. You cannot modify the list through this view.

**Chapter 22 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)22.2.1

**Why is a constant factor ignored in the Big O notation? Why is a nondominating term ignored in the Big O notation?**

The constant factor is ignored in big O notation, because it has no impact on the growth rate of the time complexity function. A nondominating term is ignored in Big O notation, because as the input size grows, the dominating term grows much faster than the nondominating term.

[▼](javascript:collapse('expandText1',%20'collapseText1');)22.2.2

**What is the order of each of the following functions?**

(a) (n2 + 1)2/n

(b) (n2 + log2n)2 / n

(c) n3 + 100n2 + n

(d) 2n + 100n2 + 45n

(e) n2n + n22n

(a) (n2 + 1)2/n = O(n3)

(b) (n2 + log2n)2 / n = O(n3)

(c) n3 + 100n2 + n = O(n3)

(d) 2n + 100n2 + 45n = O(2n)

(e) n2n + n22n = O(n22n)

Section 22.3

[▼](javascript:collapse('expandText2',%20'collapseText2');)22.3.1

**Count the number of iterations in the following loops.**

(a)

int count = 1;

while (count < 30) {

count = count \* 2;

}

(b)

int count = 15;

while (count < 30) {

count = count \* 3;

}

(c)

int count = 1;

while (count < n) {

count = count \* 2;

}

(d)

int count = 15;

while (count < n) {

count = count \* 3;

}

(A) 5   
(B) 1   
(C) The ceiling of log2n times   
(D) The ceiling of log3(n/15) times

[▼](javascript:collapse('expandText3',%20'collapseText3');)22.3.2

**How many stars are displayed in the following code if n is 10? How many if n is 20? Use the Big O notation to estimate the time complexity.**

(a)

for (int i = 0; i < n; i++) {

System.out.print('\*');

}

(b)

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.print('\*');

}

}

(c)

for (int k = 0; k < n; k++) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.print('\*');

}

}

}

(d)

for (int k = 0; k < 10; k++) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < n; j++) {

System.out.print('\*');

}

}

}

if n is 10: (a) 10 (b) 10^2 (c) 10^3 (d) 10\*10^2   
if n is 20: (a) 20 (b) 20^2 (c) 20^3 (d) 20\*20^2   
Using Big-O notation: O(n), O(n^2), O(n^3), O(n^2)

[▼](javascript:collapse('expandText4',%20'collapseText4');)22.3.3

**Use the Big O notation to estimate the time complexity of the following methods:**

(a)

public static void mA(int n) {

for (int i = 0; i < n; i++) {

System.out.print(Math.random());

}

}

(b)

public static void mB(int n) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < i; j++)

System.out.print(Math.random());

}

}

(c)

public static void mC(int[] m) {

for (int i = 0; i < m.length; i++) {

System.out.print(m[i]);

}

for (int i = m.length - 1; i >= 0; )

{

System.out.print(m[i]);

i--;

}

}

(d)

public static void mD(int[] m) {

for (int i = 0; i < m.length; i++) {

for (int j = 0; j < i; j++)

System.out.print(m[i] \* m[j]);

}

}

(a): O(n)   
(b): O(n^2)   
(c): O(n)   
(d): O(n^2)

[▼](javascript:collapse('expandText5',%20'collapseText5');)22.3.4

**Design an O(n) time algorithm for computing the sum of numbers from n1 to n2 for (n1 < n2). Can you design an O(1) for performing the same task?**

An O(n) time algorithm for this is

int sum = 0;

for (int i = n1; i <= n2; i++)

sum += i;

An O(1) time algorithm for this is

int sum = n2 \* (n2 + 1) / 2 - n1 \* (n1 - 1) / 2;

[▼](javascript:collapse('expandText6',%20'collapseText6');)22.3.5

**Example 7 in Section 22.3 assumes n = 2k Revise the algorithm for an arbitrary n and prove that the complexity is still O(logn).**

result = a;

i = 2;

while (i <= n) {

result = result \* result;

i \*= 2;

}

for (int j = i / 2 + 1; j <= n; j++)

result = result \* a;

Assume that 2k-1 <= n < 2k. The while loop is executed k-1 times. The for loop is executed at most 2k-2k-1=2k-1 times. So, the total complexity is O(n). **Consider another implementation:**

public static int f(int a, int n) {

if (n == 1) {

return a;

}

else {

int temp = f(a, n / 2);

if (n % 2 == 0) {

return temp \* temp;

}

else {

return a \* temp \* temp;

}

}

}

This implementation results in O(logn) complexity.

[▼](javascript:collapse('expandText7',%20'collapseText7');)22.4.1

**Put the following growth functions in order: 5n3/4032, 44logn, 10nlogn, 500, 2n2, 2n/45, 3n**

500,44logn,3n,10nlogn 2n2,5n3/4032,2n/45

[▼](javascript:collapse('expandText8',%20'collapseText8');)22.4.2

**Estimate the time complexity for adding two n by m matrices, and for multiplying an n by m matrix by an m by k matrix.**

Adding two matrices: O(nm). Multiplying two matrices: O(nmk)

[▼](javascript:collapse('expandText9',%20'collapseText9');)22.4.3

**Describe an algorithm for finding the occurrence of the max element in an array. Analyze the complexity of the algorithm.**

The algorithm can be designed as follows: Maintain two variables, max and count. max stores the current max number, and count stores its occurrences. Initially, assign the first number to max and 1 to count. Compare each subsequent number with max. If the number is greater than max, assign it to max and reset count to 1. If the number is equal to max, increment count by 1. Since each element in the array is examined only once, the complexity of the algorithm is O(n).

[▼](javascript:collapse('expandText10',%20'collapseText10');)22.4.4

**Describe an algorithm for removing duplicates from an array. Analyze the complexity of the algorithm.**

The algorithm can be designed as follows: For each element in the input array, store it to a new array if it is new. If the number is already in the array, ignore it. The time for checking whether an element is already in the new array is O(n), so the complexity of the algorithm is O(n^2).

[▼](javascript:collapse('expandText11',%20'collapseText11');)22.4.5

**Analyze the following sorting algorithm:**

for (int i = 0; i < list.length - 1; i++) {

if (list[i] > list[i + 1]) {

swap list[i] with list[i + 1];

i = -1;

}

}

This is similar to bubble sort. Whenever a swap is made, it goes back to the beginning of the loop. In the worst case, there will be O(n^2) of swaps. For each swap, O(n) number of comparisons may be made in the worst case. So, the total is O(n^3) in the worst case.

[▼](javascript:collapse('expandText12',%20'collapseText12');)22.4.6

**Analyze the complexity for computing a polynomial f(x) of degree n for a given x value using a brute-force approach and the Horner's approach,** respectively. A brute-force approach is to compute each term in the polynomial and add them together. The Horner's approach was introduced in Section 6.7.   
f(x) = anxn + an-1xn-1 + an-2xn-2 + ... + a1x1 + a0

A brute-force for approach to evaluate a polynomial f(x) of degree n will take n+(n-1)+…+2+1=O(n^2) time. The Horner's method takes O(n) time.

Section 22.5

[▼](javascript:collapse('expandText13',%20'collapseText13');)22.5.1

**What is dynamic programming? Give an example of dynamic programming.**

See the definition and example in the text.

[▼](javascript:collapse('expandText14',%20'collapseText14');)22.5.2

**Why is the recursive Fibonacci algorithm inefficient, but the nonrecursive Fibonacci algorithm efficient?**

The recursive Fibonacci algorithm is inefficient, because the subproblems in the recursive Fibonacci algorithm overlaps, which causes redundant work. The non- recursive Fibonacci algorithm is dynamic algorithm that avoids redundant work.

[▼](javascript:collapse('expandText15',%20'collapseText15');)22.6.1

**Prove that the following algorithm for finding the GCD of the two integers m and n is incorrect.**

int gcd = 1;

for (int k = Math.min(Math.sqrt(n), Math.sqrt(m)); k >= 1; k--) {

if (m % k == 0 && n % k == 0) {

gcd = k;

break;

}

}

To prove this is wrong, all you need is to give a counter example to show the algorithm does not work. Try n = 64 and m = 48. The algorithm will produce the gcd 8, but the actual gcd is 16.

Section 22.7

[▼](javascript:collapse('expandText16',%20'collapseText16');)22.7.1

**Prove that if n is not prime, there must exist a prime number p such that p <= sqrt(n) and p is a factor of n.**

If n is not a prime, then there exists two number n1 and n2 such that n1 \* n2 = n. Assume n1 <= n2, n1 <= srqt(n). If n1 is not a prime, you can continue the same process to find the factors of n1, until a factor is a prime.

[▼](javascript:collapse('expandText17',%20'collapseText17');)22.7.2

**Describe how the sieve of Eratosthenes is used to find the prime numbers.**

See the text.

Section 22.8

[▼](javascript:collapse('expandText18',%20'collapseText18');)22.8.1

**What is the divide-and-conquer approach? Give an example.**

See the definition and example in the text.

[▼](javascript:collapse('expandText19',%20'collapseText19');)22.8.2

**What is the difference between divide-and-conquer and dynamic programming?**

[▼](javascript:collapse('expandText20',%20'collapseText20');)22.8.3

**Can you design an algorithm for finding the minimum element in a list using divide-and-conquer? What is the complexity of this algorithm?**

Yes. Finding the minimum in the first half and the second half of the list and return the minimum of these two. So, the time complexity is O(n) = 2 \* O(n/2) + O(1) = O(n).

Section 22.9

[▼](javascript:collapse('expandText21',%20'collapseText21');)22.9.1

**What is backtracking? Give an example.**

See the definition and example in the text.

[▼](javascript:collapse('expandText22',%20'collapseText22');)22.9.2

**If you generalize the Eight Queens problem to the n-Queens problem in an n-by-n chessboard, what will be the complexity of the algorithm?**

O(n!)

Section 22.10

[▼](javascript:collapse('expandText23',%20'collapseText23');)22.10.1

**What is a convex hull?**

See the text.

[▼](javascript:collapse('expandText24',%20'collapseText24');)22.10.2

**Describe the gift-wrapping algorithm for finding a convex hull. Should list H be implemented using an ArrayList or a LinkedList?**

See the text.

[▼](javascript:collapse('expandText25',%20'collapseText25');)22.10.3

**Describe Graham's algorithm for finding a convex hull. Why does the algorithm use a stack to store the points in a convex hull?**

See the text.

**Chapter 23 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)23.2.1

**Describe how an insertion sort works. What is the time complexity for an insertion sort?**

See the text. The time complexity for an insertion sort is O(n^2).

[▼](javascript:collapse('expandText1',%20'collapseText1');)23.2.2

**Use Figure 23.1 as an example to show how to apply an insertion sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.**

**Omitted.**

[▼](javascript:collapse('expandText2',%20'collapseText2');)23.2.3

**If a list is already sorted, how many comparisons will the insertionSort method perform?**

n - 1 times.

Section 23.3

[▼](javascript:collapse('expandText3',%20'collapseText3');)23.3.1

**Describe how a bubble sort works. What is the time complexity for a bubble sort?**

See the text. The time complexity for a bubble sort is O(n^2).

[▼](javascript:collapse('expandText4',%20'collapseText4');)23.3.2

**Use Figure 23.3 as an example to show how to apply a bubble sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.**

Omitted.

[▼](javascript:collapse('expandText5',%20'collapseText5');)23.3.3

**If a list is already sorted, how many comparisons will the bubbleSort method perform?**

n - 1 times.

Section 23.4

[▼](javascript:collapse('expandText6',%20'collapseText6');)23.4.1

**Describe how a merge sort works. What is the time complexity for a merge sort?**

See the text. The time complexity for a merge sort is O(nlogn).

[▼](javascript:collapse('expandText7',%20'collapseText7');)23.4.2

**Use Figure 23.4 as an example to show how to apply a merge sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.**

Omitted.

[▼](javascript:collapse('expandText8',%20'collapseText8');)23.4.3

**What is wrong if lines 6-15 in Listing 23.6, MergeSort.java, are replaced by the following code?**

// Merge sort the first half

int[] firstHalf = new int[list.length / 2 + 1];

System.arraycopy(list, 0, firstHalf, 0, list.length / 2 + 1);

mergeSort(firstHalf);

// Merge sort the second half

int secondHalfLength = list.length - list.length / 2 - 1;

int[] secondHalf = new int[secondHalfLength];

System.arraycopy(list, list.length / 2 + 1,

secondHalf, 0, secondHalfLength);

mergeSort(secondHalf);

Consider a list with two elements. firstHalf will be the entire list.

[▼](javascript:collapse('expandText9',%20'collapseText9');)23.5.1

**Describe how quick sort works. What is the time complexity for a quick sort?**

See the text. The time complexity for a quick sort is O(n^2) in the worst case and O(nlogn) in the average case.

[▼](javascript:collapse('expandText10',%20'collapseText10');)23.5.2

**Why is quick sort more space efficient than merge sort?**

Quick sort does not need to create temporary arrays, while merge sort needs temporary arrrys.

[▼](javascript:collapse('expandText11',%20'collapseText11');)23.5.3

**Use Figure 23.7 as an example to show how to apply a quick sort on {45, 11, 50, 59, 60, 2, 4, 7, 10}.**

Omitted.

[▼](javascript:collapse('expandText12',%20'collapseText12');)23.5.4

**If lines 37-38 in the QuickSort program is removed, will it still work? Give a counter example to show that it will not work.**

Try to sort this list {2, 22, 2, 5, -3} without the code in ines 37-38.

Section 23.6

[▼](javascript:collapse('expandText13',%20'collapseText13');)23.6.1

**What is a complete binary tree? What is a heap? Describe how to remove the root from a heap and how to add a new object to a heap.**

A binary tree is complete if every level of the tree is full except that the last level may not be full and all the leaves on the last level are placed left-most. A heap is a binary tree with the following properties:  
1. It is a complete binary tree.  
2. Each node is greater than or equal to any of its children.

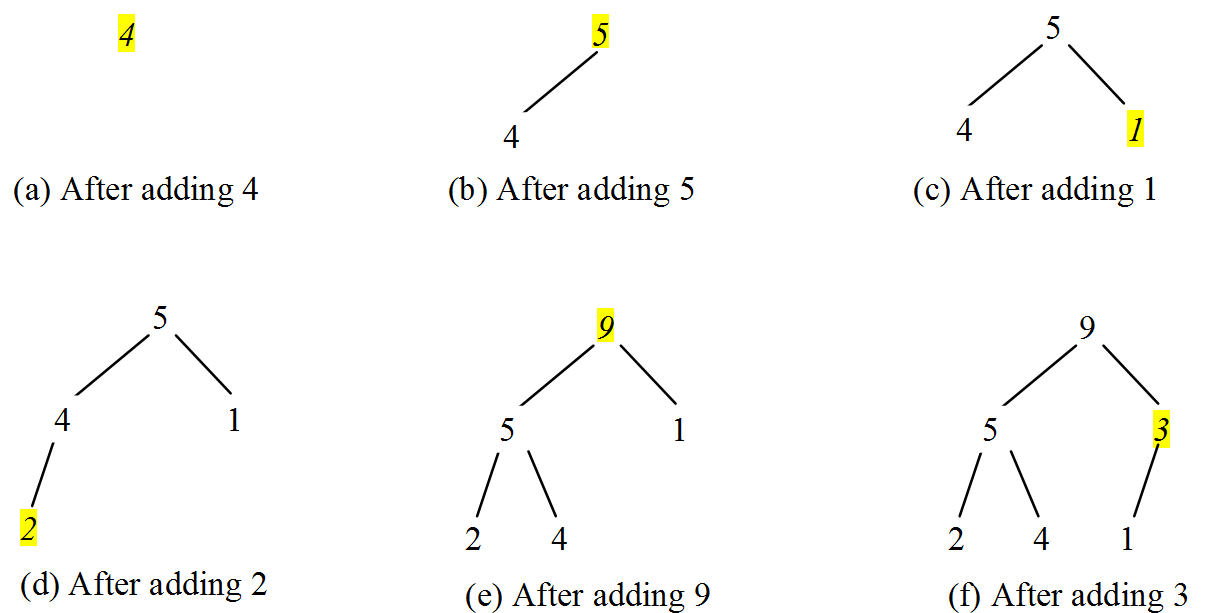
[▼](javascript:collapse('expandText14',%20'collapseText14');)23.6.2

**When a heap in Figure 23.11a is stored in a list, what is index for element 44? What is the index of parent for 44 and what is the index of left and right children of 44?**

The index for 44 is 5. The parent index for 44 is 2. The left child index for 44 is 11 and the right child index for 44 is 12.

[▼](javascript:collapse('expandText15',%20'collapseText15');)23.6.3

**Add the elements 4, 5, 1, 2, 9, and 3 into a heap in this order. Draw the diagrams to show the heap after each element is added.**



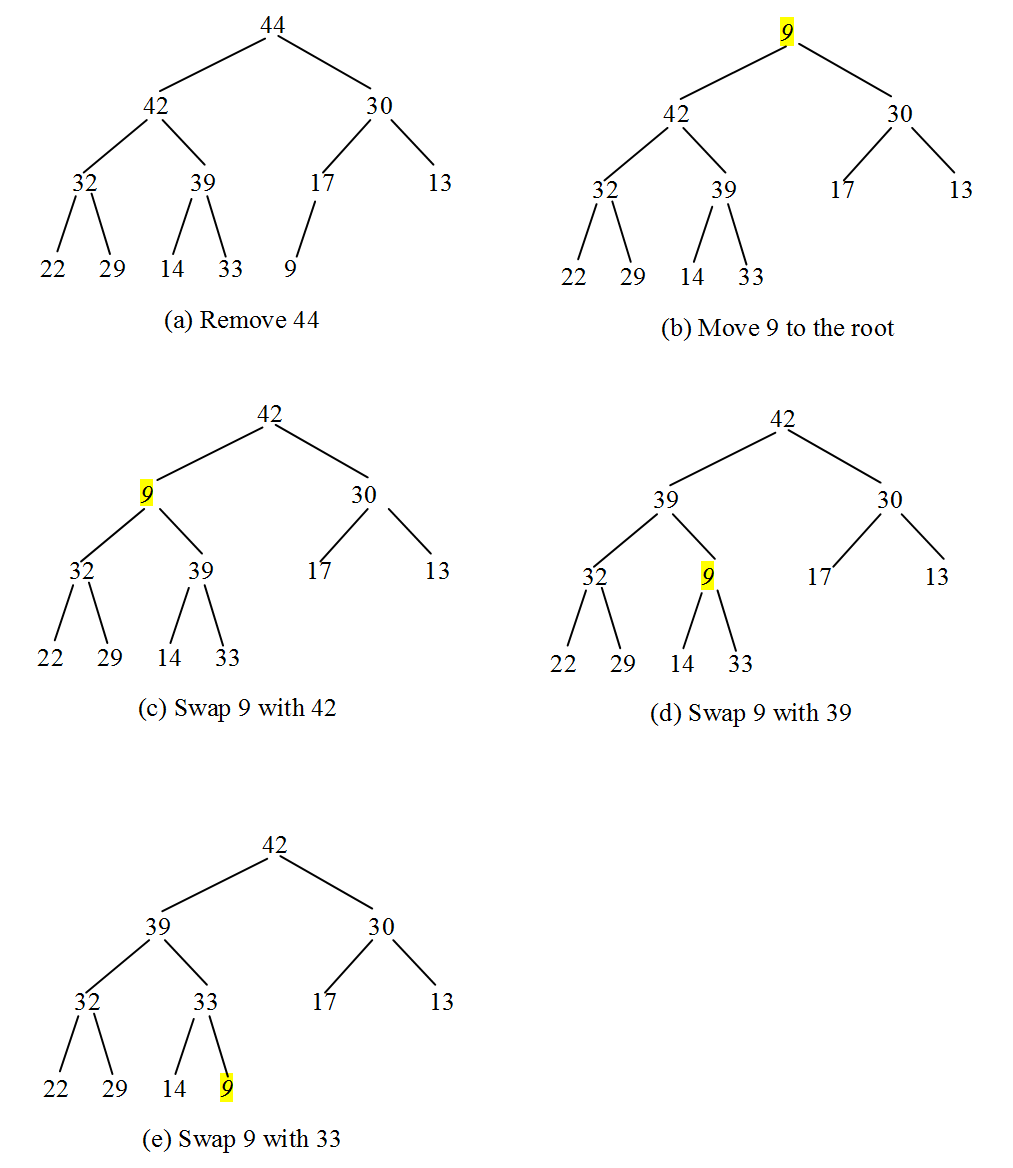
[▼](javascript:collapse('expandText16',%20'collapseText16');)23.6.4

**Show the steps of creating a heap using {45, 11, 50, 59, 60, 2, 4, 7, 10}.**

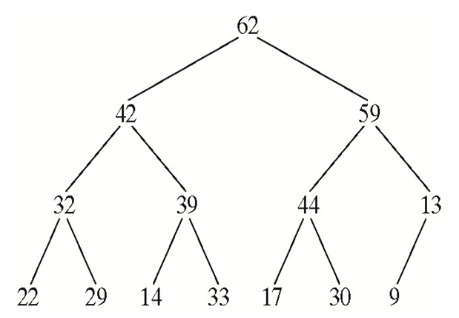
Omitted.

[▼](javascript:collapse('expandText17',%20'collapseText17');)23.6.5

**Show the heap after the root in the heap in Figure 23.15c is removed.**



[▼](javascript:collapse('expandText18',%20'collapseText18');)23.6.6

**Given the following heap, show the steps of removing all nodes from the heap.**

Omitted.

[▼](javascript:collapse('expandText19',%20'collapseText19');)23.6.7

**Which of the following statements are wrong?**

1 Heap<Object> heap1 = new Heap<>();

2 Heap<Number> heap2 = new Heap<>();

3 Heap<BigInteger> heap3 = new Heap<>();

4 Heap<Calendar> heap4 = new Heap<>();

5 Heap<> heap5 = new Heap<>();

Lines 1 and 2 are wrong, because Object and Number do not implement Comparable.

[▼](javascript:collapse('expandText20',%20'collapseText20');)23.6.8

**Modify line 12 in Listing 23.10 so that the elements are sorted in a non-increasing order.**

**Change it to**

for (int i = 0; i < list.length; i++)

[▼](javascript:collapse('expandText21',%20'collapseText21');)23.6.9

**What is the return value from invoking the remove method if the heap is empty?**

The return value will be null.

[▼](javascript:collapse('expandText22',%20'collapseText22');)23.6.10

**What is the time complexity of inserting a new element into a heap and what is the time complexity of deleting an element from a heap?**

O(logn) for both insertion and deletion.

[▼](javascript:collapse('expandText23',%20'collapseText23');)23.6.11

**What is the height of an empty heap? What is the height of a heap with only one element? What is the height of a non-empty heap? What is the** height of a heap with 16, 17, and 512 elements? If the height of a heap is 5, what is the maximum number of nodes in the heap?

The height of an empty heap is -1 be definition. The height of a heap with only one element is 0. The height of a non-empty heap is the length of a longest path from the root to a leaf. The height of heap with 16 elements is 4. The height of heap with 17 elements is 4. The height of heap with 512 elements is 9. The maximum number of nodes in a heap of height 5 is 63.

Section 23.7

[▼](javascript:collapse('expandText24',%20'collapseText24');)23.7.1

**Can you sort a list of strings using a bucket sort?**

Bucket sort is not suitable for sorting strings.

[▼](javascript:collapse('expandText25',%20'collapseText25');)23.7.2

Show how the radix sort works using the numbers 454, 34, 23, 43, 74, 86, and 76.

Omitted.

Section 23.8

[▼](javascript:collapse('expandText26',%20'collapseText26');)23.8.1

**Describe how external sort works. What is the complexity of the external sort algorithm?**

See the text. The time complexity for an external sort is O(nlogn) on disk I/O of blocks of data.

[▼](javascript:collapse('expandText27',%20'collapseText27');)23.8.2

Ten numbers {2, 3, 4, 0, 5, 6, 7, 9, 8, 1} are stored in the external file largedata.dat. Trace the SortLargeFile program by hand with MAX\_ARRAY\_SIZE 2.

Omitted.

**Chapter 24 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)24.2.1

**Suppose list is an instance of MyList, can you get an iterator for list using list.iterator()?**

Yes. Because MyList is Iterable.

[▼](javascript:collapse('expandText1',%20'collapseText1');)24.2.2

**Can you create a list using new MyList()?**

No. Because MyList is an interface.

[▼](javascript:collapse('expandText2',%20'collapseText2');)24.2.3

**What methods in Collection are overridden as default methods in MyList?**

The methods isEmpty(), contains(Object e), add(E e), remove(Object e), size(), containsAll(Collection<?> c), addAll(Collection<? extends E> c),removeAll(Collection<?> c), retainAll(Collection<?> c), toArray(), and toArray(T[] are overridden in MyList.

[▼](javascript:collapse('expandText3',%20'collapseText3');)24.2.4

**What are the benefits of overriding the methods in Collection as default methods in MyList?**

Default methods is a new feature in Java 8. The default methods providedefualt implementation for some methods in the interface.

Section 24.3

[▼](javascript:collapse('expandText4',%20'collapseText4');)24.3.1

**What are the limitations of the array data type?**

An array is a fixed-size data structure. Once an array is created, its size cannot be changed.

[▼](javascript:collapse('expandText5',%20'collapseText5');)24.3.2

**MyArrayList is implemented using an array, and an array is a fixed-size data structure. Why is MyArrayList considered a dynamic data structure?**

MyArrayList is implemented using an array and an array is a fixed-size data structure. But MyArrayList is considered as a dynamic data structure, because its storage size changes behind the scene and hidden from the user.

[▼](javascript:collapse('expandText6',%20'collapseText6');)24.3.3

**Show the length of the array in MyArrayList after each of the following statements is executed.**

1 MyArrayList<Double> list = new MyArrayList<>();

2 list.add(1.5);

3 list.trimToSize();

4 list.add(3.4);

5 list.add(7.4);

6 list.add(17.4);

After line 1, list's length is 16. After line 2, list's length is 16. After line 3, list's length is 1. After line 4, list's length is 3. After line 5, list's length is 3. After line 6, list's length is 7.

[▼](javascript:collapse('expandText7',%20'collapseText7');)24.3.4

**What is wrong if lines 11-12 in Listing 24.2, MyArrayList.java,**

for (int i = 0; i < objects.length; i++)

add(objects[i]);

**are replaced by**

data = objects;

size = objects.length;

If

for (int i = 0; i < objects.length; i++)

add(objects[i]);

are replaced by

super(objects);

When constructing an ArrayList using new ArrayList(objects), the super class' constructor is invoked first to add element in objects to data. However, data has not been initialized yet. data will be initialized after the body of the superclass' constructor is executed. So you will get a NullPointerException when attempting to add an element to data. See Supplement III.I, "Initialization Block," for reference.If

for (int i = 0; i < objects.length; i++)

add(objects[i]);

are replaced by

data = objects;

size = objects.length;

Then data and objects refer to the same array. This is a security hole. You may change ArrayList by directing changing the array elements through objects.

[▼](javascript:collapse('expandText8',%20'collapseText8');)24.3.5

**If you change the code in line 33 in Listing 24.2, MyArrayList.java, from**

E[] newData = (E[])(new Object[size \* 2 + 1]);

to

E[] newData = (E[])(new Object[size \* 2]);

the program is incorrect. Can you find the reason?

(Hint: To find the bug, perform trimToSize() on an empty list, then add a new element to the list.)When an empty array list is trimmed, its size becomes 0. If you create a new array by doubling its size, the new array size is still 0. Adding a new element now would cause an ArrayIndexOutOfBounds exception.

[▼](javascript:collapse('expandText9',%20'collapseText9');)24.3.6

**Will the MyArrayList class have memory leak if the following code in line 41 is deleted?**

data = (E[])new Object[INITIAL\_CAPACITY];

Yes.

[▼](javascript:collapse('expandText10',%20'collapseText10');)24.3.7

**The get(index) method invokes the checkIndex(index) method (lines 59-63 in Listing 24.2) to throw an IndexOutOfBoundsException if the index is out of bounds. Suppose the add(index, e) method is implemented as follows:**

public void add(int index, E e) {

checkIndex(index);

// Same as lines 23-33 in Listing 24.2 MyArrayList.java

}

**What will happen if you run the following code?**

MyArrayList<> list = new MyArrayList<>();

list.add("New York");

list.add(e) invokes list.add(list.size(),e), which will throw an exception, because size is now 0.

[▼](javascript:collapse('expandText11',%20'collapseText11');)24.4.1

**If a linked list does not contain any nodes, what are the values in head and tail?**

head and tail are null.

[▼](javascript:collapse('expandText12',%20'collapseText12');)24.4.2

**If a linked list has only one node, is head == tail true? List all cases in which head == tail is true.**

Yes. When the list is empty, head == tail is also true.

[▼](javascript:collapse('expandText13',%20'collapseText13');)24.4.3

**Draw a diagram to show the linked list after each of the following statements is executed.**

MyLinkedList<Double> list = new MyLinkedList<>();

list.add(1.5);

list.add(6.2);

list.add(3.4);

list.add(7.4);

list.remove(1.5);

list.remove(2);

Omitted.

[▼](javascript:collapse('expandText14',%20'collapseText14');)24.4.4

**When a new node is inserted to the head of a linked list, will the head and the tail be changed?**

The head will be changed. The tail will also be changed if the list is empty before the insertion.

[▼](javascript:collapse('expandText15',%20'collapseText15');)24.4.5

**When a new node is appended to the end of a linked list, will the head and the tail be changed?**

The head will be changed if the list is empty before the insertion. The tail will always be changed.

[▼](javascript:collapse('expandText16',%20'collapseText16');)24.4.6

**Simplify the following code in Listing 24.5 using a conditional expression.**

if (current != null) {

result.append(", "); // Separate two elements with a comma

}

else {

result.append("]"); // Insert the closing ] in the string

}

result.append((current != null) ? ", " : "]");

[▼](javascript:collapse('expandText17',%20'collapseText17');)24.4.7

**Simplify the code for the removeLast() method by invoking the removeFirst() method when the size is less than or equal to 1. Is the new code more efficient in execution time?**

public E removeLast() {

if (size <= 1) {

return removeFirst();

}

else {

Node<E> current = head;

for (int i = 0; i < size - 2; i++) {

current = current.next;

}

Node<E> temp = tail;

tail = current;

tail.next = null;

size--;

return temp.element;

}

}

This new code is simpler and better. It reuses the existing code and makes the code easy to maintain.   
The time complexity for the new code is O(1), which is the same as before.   
(Thanks to the UT Dallas students for their contribution. 10/14/2015)

[▼](javascript:collapse('expandText18',%20'collapseText18');)24.4.8

**What is the time complexity of the addFirst(e) and removeFirst() methods in MyLinkedList?**

O(1)

[▼](javascript:collapse('expandText19',%20'collapseText19');)24.4.9

**What would be the time complexity for the size() method if the size data field is not used in MyLinkedList?**

O(n)

[▼](javascript:collapse('expandText20',%20'collapseText20');)24.4.10

**Suppose you need to store a list of elements. If the number of elements in the program is fixed, what data structure should you use? If the number of elements in the program changes, what data structure should you use?**

If the number of elements is fixed in the program, use array is more efficient. If the number of elements changes in the program, you may use MyArrayList or MyLinkedList.

[▼](javascript:collapse('expandText21',%20'collapseText21');)24.4.11

**If you have to add or delete the elements at the beginning of a list, should you use MyArrayList or MyLinkedList? If most of the operations on a list involve retrieving an element at a given index, should you use MyArrayList or MyLinkedList?**

If you have to add or delete the elements anywhere in a list, use MyLinkedList. If most of operations on a list involve retrieving an element at a given index, use MyArrayList.

[▼](javascript:collapse('expandText22',%20'collapseText22');)24.4.12

**Both MyArrayList and MyLinkedList are used to store a list of objects. Why do we need both types of lists?**

Both MyArrayList and MyLinkedList are used to store a list of objects. Why do we need two? MyLinkedList is more efficient for deletion and insertion at the beginning of the list. MyArrayList is more efficient for all other operations.

[▼](javascript:collapse('expandText23',%20'collapseText23');)24.4.13

**Implement the removeLast() method in a doubly linked list in O(1) time.**

Note that in a singly linked list, to find the second-to-last node, you have loop all the way from head to the second-to-last node, which takes O(n) time. Here is the copy of the code from the text:

1 public E removeLast() {

2 if (size == 0) return null; // Nothing to remove

3 else if (size == 1) { // Only one element in the list

4 Node<E> temp = head;

5 head = tail = null; // list becomes empty

6 size = 0;

7 return temp.element;

8 }

9 else {

10 Node<E> current = head;

11

12 for (int i = 0; i < size - 2; i++)

13 current = current.next;

14

15 Node<E> temp = tail;

16 tail = current;

17 tail.next = null;

18 size--;

19 return temp.element;

20 }

21 }

**In a doubly linked list, the second-to-last node can be found in O(1) time using tail.previous. Here is the complete code:**

1 public E removeLast() {

2 if (size == 0) return null; // Nothing to remove

3 else if (size == 1) { // Only one element in the list

4 Node<E> temp = head;

5 head = tail = null; // list becomes empty

6 size = 0;

7 return temp.element;

8 }

9 else {

10 Node<E> temp = tail;

11 tail = tail.previous;

12 tail.next = null;

13 size--;

14 return temp.element;

15 }

[▼](javascript:collapse('expandText24',%20'collapseText24');)24.5.1

**You can use inheritance or composition to design the data structures for stacks and queues. Discuss the pros and cons of these two approaches.**

Using inheritance: You can declare the stack class by extending the array list class, and the queue class by extending the linked list class. Using composition: You can declare an array list as a data field in the stack class, and a linked list as a data field in the queue class. Both designs are fine, but using composition is better because it enables you to declare a complete new stack class and queue class without inheriting the unnecessary and inappropriate methods from the array list and linked list.

[▼](javascript:collapse('expandText25',%20'collapseText25');)24.5.2

**If LinkedList is replaced by ArrayList in lines 2-3 in Listing 24.6 GenericQueue.java, what will be the time complexity for the enqueue and dequeue methods?**

The time complexity for enqueue will be O(1) and for dequeue will be O(n).

[▼](javascript:collapse('expandText26',%20'collapseText26');)24.5.3

**Which lines of the following code are wrong?**

1 List<> list = new ArrayList<>();

2 list.add("Tom");

3 list = new LinkedList<>();

4 list.add("Tom");

5 list = new GenericStack<>();

6 list.add("Tom");

**Line 5 will be wrong, because GenericStack is not a subtype of MyList.**

Section 24.6

[▼](javascript:collapse('expandText27',%20'collapseText27');)24.6.1

**What is a priority queue?**

In a priority queue, elements are assigned with priorities. When accessing elements, the element with the highest priority is removed first.

[▼](javascript:collapse('expandText28',%20'collapseText28');)24.6.2

**What are the time complexity of the enqueue, dequeue , and getSize methods in MyProrityQueue?**

For enqueue and dequeue in a priority queue, the complexity is O(logn). For getSize(), the complexity is O(1).

[▼](javascript:collapse('expandText29',%20'collapseText29');)24.6.3

**Which of the following statements are wrong?**

1 MyPriorityQueue<Object> q1 = new MyPriorityQueue<>();

2 MyPriorityQueue<Number> q2 = new MyPriorityQueue<>();

3 MyPriorityQueue<Integer> q3 = new MyPriorityQueue<>();

4 MyPriorityQueue<Date> q4 = new MyPriorityQueue<>();

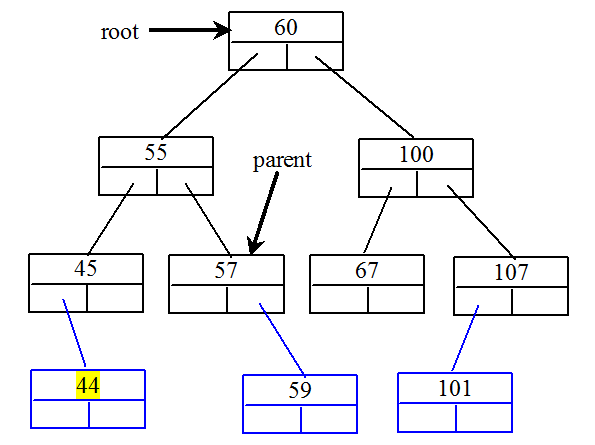
5 MyPriorityQueue<> q5 = new MyPriorityQueue<>();

Lines 1 and 2 are wrong, because Object and Number don't implement Comparable.

**Chapter 25 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)25.2.1

**Show the result of inserting 44 into Figure 25.4b.**



[▼](javascript:collapse('expandText1',%20'collapseText1');)25.2.2

**Show the inorder, preorder, and postorder of traversing the elements in the binary tree shown in Figure 25.1b.**

Inorder: A F G M R T   
Preorder: G F A R M T   
Postorder: A F M T R G

[▼](javascript:collapse('expandText2',%20'collapseText2');)25.2.3

**If a set of elements is inserted into a BST in two different orders, will the two corresponding BSTs look the same? Will the inorder traversal be the same? Will the postorder traversal be the same? Will the preorder traversal be the same?**

If a set of the same elements is inserted into a binary tree in two different orders, will the two corresponding binary trees look the same? No. Will the inorder traversal be the same? Yes. Will the postorder traversal be the same? No. Will the preorder traversal be the same? No.

[▼](javascript:collapse('expandText3',%20'collapseText3');)25.2.4

**What is the time complexity of inserting an element into a BST?**

The time complexity of inserting an element to a binary tree is O(n).

[▼](javascript:collapse('expandText4',%20'collapseText4');)25.2.5

Implement the search(element) method using recursion.

@Override

/\*\* Returns true if the element is in the tree \*/

public boolean search(E e) {

return search(root, e);

}

public boolean search(TreeNode<E> root, E e) {

if (root == null)

return false;

else if (e.compareTo(root.element) < 0)

return search(root.left, e);

else if (e.compareTo(root.element) > 0)

return search(root.right, e);

else

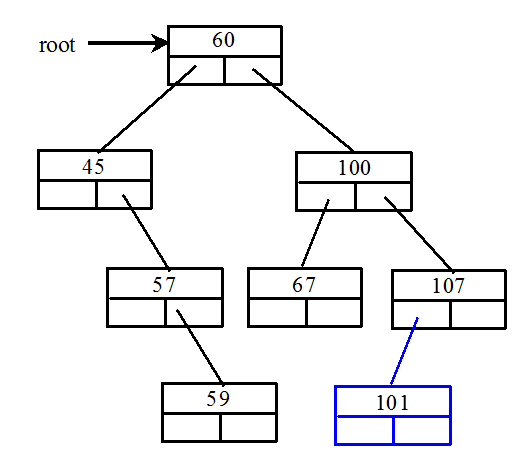
return true;

}

Section 25.3

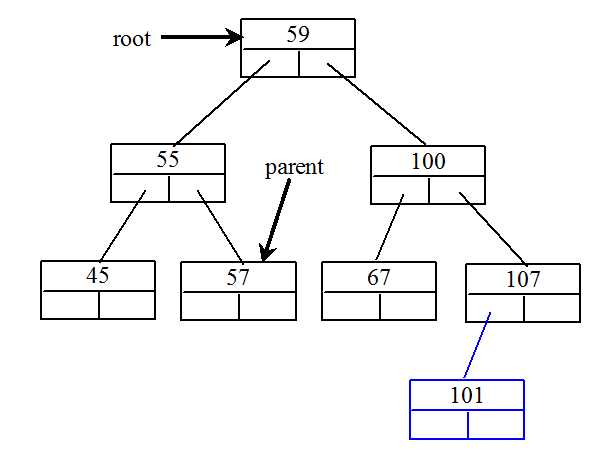
[▼](javascript:collapse('expandText5',%20'collapseText5');)25.3.1

**Show the result of deleting 55 from the tree in Figure 25.4b.**



[▼](javascript:collapse('expandText6',%20'collapseText6');)25.3.2

**Show the result of deleting 60 from the tree in Figure 25.4b.**



[▼](javascript:collapse('expandText7',%20'collapseText7');)25.3.3

**What is the time complexity of deleting an element from a BST?**

The time complexity of deleting an element from a binary tree is O(n).

[▼](javascript:collapse('expandText8',%20'collapseText8');)25.3.4

**Is the algorithm correct if lines 203-207 in Listing 25.4 in Case 2 of the delete() method are replaced by the following code?**

parentOfRightMost.right = rightMost.left;

No. Consider the case when current is parentOfRightMost and current.left is rightMost. You have to assign rightMost.left to parentOfRightMost.left.

Section 25.4

[▼](javascript:collapse('expandText9',%20'collapseText9');)25.4.1

**How many times will the displayTree method be invoked if the tree is empty? How many times will the displayTree method be invoked if the tree has 100 nodes?**

The displayTree method will be invoked 0 time if the tree is empty. The displayTree method will be invoked 100 times if the tree has 100 nodes.

[▼](javascript:collapse('expandText10',%20'collapseText10');)25.4.2

**In what order are the nodes in the tree visited by the displayTree method: inorder, preorder, or postorder?**

The nodes in the tree are visited in preorder.

[▼](javascript:collapse('expandText11',%20'collapseText11');)25.4.3

**What would happen if the code in lines 47-52 in BTView.java is moved to line 33?**

You will see the line that connecting the nodes to be displayed on top of nodes starting from the center of the node, not from the edge of the node, because the line would be displayed after the node.

[▼](javascript:collapse('expandText12',%20'collapseText12');)25.4.4

**What is MVC? What are the benefits of the MVC?**

See the text.

[▼](javascript:collapse('expandText13',%20'collapseText13');)25.5.1

**What is an iterator?**

An iterator is an object that provides a uniform way for traversing the elements in a container such as a set, list, binary tree, etc.

[▼](javascript:collapse('expandText14',%20'collapseText14');)25.5.2

**What method is defined in the java.lang.Iterable<E> interface?**

The iterator() method is defined in the java.lang.Iterable interface.

[▼](javascript:collapse('expandText15',%20'collapseText15');)25.5.3

**Suppose you delete implements Collection<E> from line 3 in Listing 25.3, Tree.java. Will Listing 25.10 still compile?**

No. Since Tree is not iterable now, the foreach loop cannot be used.

[▼](javascript:collapse('expandText16',%20'collapseText16');)25.5.4

**What is the benefit of being a subtype of Iterable<E>?**

Being a subtype of Iterable, the elements of the container can be traversed using a for-each loop.

[▼](javascript:collapse('expandText17',%20'collapseText17');)25.5.5

**Write one statement that displays the maximum and minimum element in a BST object named tree.**

System.out.println("Max elememt in the tree is " +

java.util.Collections.max(tree) +

" and the min elememt in the tree is " +

java.util.Collections.min(tree));

[▼](javascript:collapse('expandText18',%20'collapseText18');)25.6.1

**Every internal node in a Huffman tree has two children. Is it true?**

Yes.

[▼](javascript:collapse('expandText19',%20'collapseText19');)25.6.2

**What is a greedy algorithm? Give an example.**

A greedy algorithm is often used in solving optimization problems. The algorithm makes the choice that is optimal locally in the hope that this choice will lead to a globally optimal solution.

[▼](javascript:collapse('expandText20',%20'collapseText20');)25.6.3

**If the Heap class in line 50 in Listing 25.9 is replaced by java.util.PriorityQueue, will the program still work?**

Yes, except that you also have to change heap.getSize() to heap.size().

[▼](javascript:collapse('expandText21',%20'collapseText21');)25.6.4

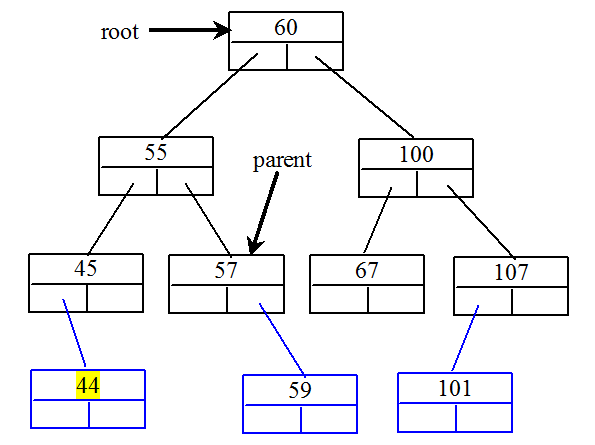
**How do you replace lines 94-99 in Listing 25.11 using one line?**

return root.weight < t.weight ? 1 : root.weight == t.root.weight ? 0 : -1;

**Chapter 25 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)25.2.1

**Show the result of inserting 44 into Figure 25.4b.**



[▼](javascript:collapse('expandText1',%20'collapseText1');)25.2.2

**Show the inorder, preorder, and postorder of traversing the elements in the binary tree shown in Figure 25.1b.**

Inorder: A F G M R T   
Preorder: G F A R M T   
Postorder: A F M T R G

[▼](javascript:collapse('expandText2',%20'collapseText2');)25.2.3

**If a set of elements is inserted into a BST in two different orders, will the two corresponding BSTs look the same? Will the inorder traversal be the same? Will the postorder traversal be the same? Will the preorder traversal be the same?**

If a set of the same elements is inserted into a binary tree in two different orders, will the two corresponding binary trees look the same? No. Will the inorder traversal be the same? Yes. Will the postorder traversal be the same? No. Will the preorder traversal be the same? No.

[▼](javascript:collapse('expandText3',%20'collapseText3');)25.2.4

**What is the time complexity of inserting an element into a BST?**

The time complexity of inserting an element to a binary tree is O(n).

[▼](javascript:collapse('expandText4',%20'collapseText4');)25.2.5

**Implement the search(element) method using recursion.**

@Override

/\*\* Returns true if the element is in the tree \*/

public boolean search(E e) {

return search(root, e);

}

public boolean search(TreeNode<E> root, E e) {

if (root == null)

return false;

else if (e.compareTo(root.element) < 0)

return search(root.left, e);

else if (e.compareTo(root.element) > 0)

return search(root.right, e);

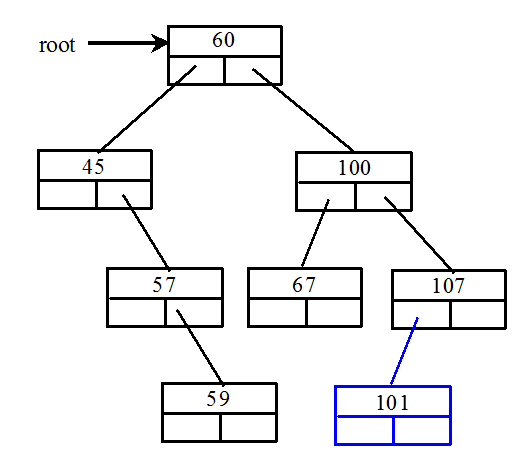
else

return true;

}

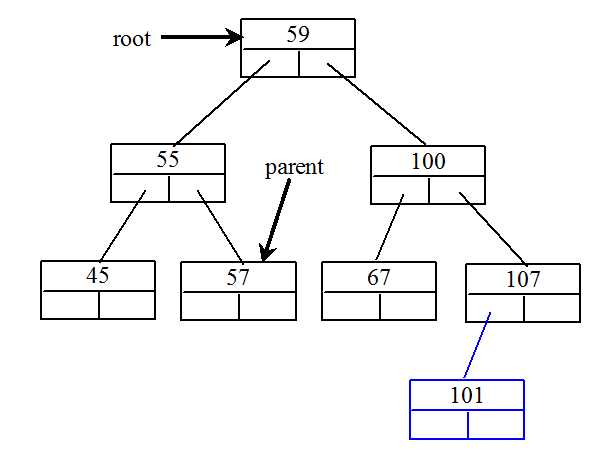
[▼](javascript:collapse('expandText5',%20'collapseText5');)25.3.1

**Show the result of deleting 55 from the tree in Figure 25.4b.**



[▼](javascript:collapse('expandText6',%20'collapseText6');)25.3.2

**Show the result of deleting 60 from the tree in Figure 25.4b.**



[▼](javascript:collapse('expandText7',%20'collapseText7');)25.3.3

**What is the time complexity of deleting an element from a BST?**

The time complexity of deleting an element from a binary tree is O(n).

[▼](javascript:collapse('expandText8',%20'collapseText8');)25.3.4

**Is the algorithm correct if lines 203-207 in Listing 25.4 in Case 2 of the delete() method are replaced by the following code?**

parentOfRightMost.right = rightMost.left;

No. Consider the case when current is parentOfRightMost and current.left is rightMost. You have to assign rightMost.left to parentOfRightMost.left.

Section 25.4

[▼](javascript:collapse('expandText9',%20'collapseText9');)25.4.1

**How many times will the displayTree method be invoked if the tree is empty? How many times will the displayTree method be invoked if the tree has 100 nodes?**

The displayTree method will be invoked 0 time if the tree is empty. The displayTree method will be invoked 100 times if the tree has 100 nodes.

[▼](javascript:collapse('expandText10',%20'collapseText10');)25.4.2

**In what order are the nodes in the tree visited by the displayTree method: inorder, preorder, or postorder?**

The nodes in the tree are visited in preorder.

[▼](javascript:collapse('expandText11',%20'collapseText11');)25.4.3

**What would happen if the code in lines 47-52 in BTView.java is moved to line 33?**

You will see the line that connecting the nodes to be displayed on top of nodes starting from the center of the node, not from the edge of the node, because the line would be displayed after the node.

[▼](javascript:collapse('expandText12',%20'collapseText12');)25.4.4

**What is MVC? What are the benefits of the MVC?**

See the text.

[▼](javascript:collapse('expandText13',%20'collapseText13');)25.5.1

**What is an iterator?**

An iterator is an object that provides a uniform way for traversing the elements in a container such as a set, list, binary tree, etc.

[▼](javascript:collapse('expandText14',%20'collapseText14');)25.5.2

**What method is defined in the java.lang.Iterable<E> interface?**

The iterator() method is defined in the java.lang.Iterable interface.

[▼](javascript:collapse('expandText15',%20'collapseText15');)25.5.3

**Suppose you delete implements Collection<E> from line 3 in Listing 25.3, Tree.java. Will Listing 25.10 still compile?**

No. Since Tree is not iterable now, the foreach loop cannot be used.

[▼](javascript:collapse('expandText16',%20'collapseText16');)25.5.4

**What is the benefit of being a subtype of Iterable<E>?**

Being a subtype of Iterable, the elements of the container can be traversed using a for-each loop.

[▼](javascript:collapse('expandText17',%20'collapseText17');)25.5.5

**Write one statement that displays the maximum and minimum element in a BST object named tree.**

System.out.println("Max elememt in the tree is " +

java.util.Collections.max(tree) +

" and the min elememt in the tree is " +

java.util.Collections.min(tree));

[▼](javascript:collapse('expandText18',%20'collapseText18');)25.6.1

**Every internal node in a Huffman tree has two children. Is it true?**

Yes.

[▼](javascript:collapse('expandText19',%20'collapseText19');)25.6.2

**What is a greedy algorithm? Give an example.**

A greedy algorithm is often used in solving optimization problems. The algorithm makes the choice that is optimal locally in the hope that this choice will lead to a globally optimal solution.

[▼](javascript:collapse('expandText20',%20'collapseText20');)25.6.3

**If the Heap class in line 50 in Listing 25.9 is replaced by java.util.PriorityQueue, will the program still work?**

Yes, except that you also have to change heap.getSize() to heap.size().

[▼](javascript:collapse('expandText21',%20'collapseText21');)25.6.4

**How do you replace lines 94-99 in Listing 25.11 using one line?**

return root.weight < t.weight ? 1 : root.weight == t.root.weight ? 0 : -1;

**Chapter 27 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)27.2.1

**What is a hash function? What is a perfect hash function? What is a collision?**

If you know the index of an element in the array, you can retrieve the element using the index in O(1) time. So, can we store the values in an array and use the key as the index to find the value? The answer is yes if you can map a key to an index. The array that stores the values is called a hash table. The function that maps a key to an index in the hash table is called a hash function.  
How do you design a hash function that produces an index from a key? Ideally, we would like to design a function that maps each search key to a different index in the hash table. Such a function is called a perfect hash function. However, it is difficult to find a perfect hash function. When two or more keys are mapped to the same hash value, we say that a collision has occurred.

Section 27.3

[▼](javascript:collapse('expandText1',%20'collapseText1');)27.3.1

**What is a hash code? What is the hash code for Byte, Short, Integer, and Character?**

A typical hash function first converts a search key to an integer value called a hash code, and then compresses the hash code into an index to the hash table.   
For a search key of the type byte, short, int, and char, simply cast it to int. So two different search keys of any one of these types will have different hash codes.

[▼](javascript:collapse('expandText2',%20'collapseText2');)27.3.2

**How is the hash code for a Float object computed?**

For a search key of the type float, use Float.floatToIntBits(key) as the hash code. Note that floatToIntBits(float f) returns an int value whose bit representation is the same as the bit representation for the floating number f. So, two different search keys of the float type will have different hash codes.

[▼](javascript:collapse('expandText3',%20'collapseText3');)27.3.3

**How is the hash code for a Long object computed?**

For a search key of the type long, simply casting it to int would not be a good choice, because all keys that differ in only the first 32 bits will have the hash code. To take the first 32 bits into consideration, divide the 64 bits into two halves and perform the exclusive or operator to combine the two halves. This process is called folding. So, the hashing code is

int hashCode = (int)(key ^ (key >> 32));

Note that >> is the right-shift operator that shifts the bits 32 position to the right.

[▼](javascript:collapse('expandText4',%20'collapseText4');)27.3.4

**How is the hash code for a Double object computed?**

For a search key of the type double, first convert it to a long value using doubleToLongBits, then perform a folding as follows:

long bits = Double.doubleToLongBits(key);

int hashCode = (int)(bits ^ (bits >> 32));

[▼](javascript:collapse('expandText5',%20'collapseText5');)27.3.5

**How is the hash code for a String object computed?**

The hashcode for a string in Java is

(...((s0\*b + s1)b + s2)b+...+sn-2)b+ sn-1

where b is 2.

[▼](javascript:collapse('expandText6',%20'collapseText6');)27.3.6

**How is a hash code compressed to an integer representing the index in a hash table?**

The hash code for a key can be a large integer that is out of the range for the hash table index. You need to scale it down to fit in the range of the index. Assume the index for a hash table is between 0 and N-1. The most common way to scale an integer to between 0 and N-1 is to use

h(hashCode) = hashCode % N

To ensure that the indices are spread evenly, choose N to be a prime number greater than 2.

[▼](javascript:collapse('expandText7',%20'collapseText7');)27.3.7

**If N is a value of the power of 2, is N / 2 same as N >> 1?**

Yes.

[▼](javascript:collapse('expandText8',%20'collapseText8');)27.3.8

**If N is a value of the power of 2, is m % N same as m & (N - 1) for any integer m?**

Yes.Open addressing is to find an open location in the hash table in

[▼](javascript:collapse('expandText9',%20'collapseText9');)27.3.9

**What is new Integer("-98").hashCode() and what is "ABCDEFGHIJK".hashCode()?**

new Integer("-98").hashCode() is -98 and"ABCDEFGHIJK".hashCode() is -331017146.

Section 27.4

[▼](javascript:collapse('expandText10',%20'collapseText10');)27.4.1

**What is open addressing? What is linear probing? What is quadratic probing? What is double hashing?**

Open addressing is to find an open location in the hash table in the event of collision. Open addressing has several variations: linear probing, quadratic probing, and double hashing. When a collision occurs during the insertion of an entry to a hash table, linear probing finds the next available location sequentially.  
Quadratic probing can avoid the clustering problem in linear probing. Linear probing looks at the consecutive cells beginning at index k. Quadratic probing, on the other hand, looks at the cells at indices (k + j2) % n, for j ≥ 0, i.e., k, (k + 1) % n, (k + 4) % n, (k + 9) % n, ..., and so on.   
Another open addressing scheme that avoids the clustering problem is known as double hashing. Starting from the initial index k, both linear probing and quadratic probing add an increment to k to define a search sequence. The increment is 1 for linear probing and j2 for quadratic probing. These increments are independent of the keys. Double hashing uses a secondary hash function on the keys to determine the increments to avoid the clustering problem.

[▼](javascript:collapse('expandText11',%20'collapseText11');)27.4.2

Describe the clustering problem for linear probing.

Linear probing tends to cause groups of consecutive cells in the hash table to be occupied. Each group is called a cluster. Each cluster is actually a probe sequence that you must search when retrieving, adding, or removing an entry. As clusters grow in size, they may merge into even larger clusters, further slowing down the search time. This is a big disadvantage of linear probing.

[▼](javascript:collapse('expandText12',%20'collapseText12');)27.4.3

**What is secondary clustering?**

Quadratic probing works in the same way as linear probing except for the change of search sequence. Quadratic probing avoids the clustering problem in linear probing, but it has its own clustering problem, called secondary clustering, i.e., the entries that collide with an occupied entry use the same probe sequence.

[▼](javascript:collapse('expandText13',%20'collapseText13');)27.4.4

**Show the hash table of size 11 after inserting entries with keys 34, 29, 53, 44, 120, 39, 45, and 40, using linear probing.**

Use the animation https://liveexample.pearsoncmg.com/dsanimation/LinearProbingeBook.html to verify your anwser. To start, click the Remove All button to empty the hash table and set the load factor threshhold to 0.99.

[▼](javascript:collapse('expandText14',%20'collapseText14');)27.4.5

**Show the hash table of size 11 after inserting entries with keys 34, 29, 53, 44, 120, 39, 45, and 40, using quadratic probing.**

Use the animation https://liveexample.pearsoncmg.com/dsanimation/QuadraticProbingeBook.html to verify your anwser. To start, click the Remove All button to empty the hash table and set the load factor threshhold to 0.99.

[▼](javascript:collapse('expandText15',%20'collapseText15');)27.4.6

**Show the hash table of size 11 after inserting entries with keys 34, 29, 53, 44, 120, 39, 45, and 40, using double hashing with the following functions:**

h(k) = k % 11;

h'(k) = 7 - k % 7;

See the text.

Section 27.5

[▼](javascript:collapse('expandText16',%20'collapseText16');)27.5.1

**Show the hash table of size 11 after inserting entries with the keys 34, 29, 53, 44, 120, 39, 45, and 40, using separate chaining.**

Use the animation https://liveexample.pearsoncmg.com/dsanimation/SeparateChainingeBook.html to verify your anwser. To start, click the Remove All button to empty the hash table and set the load factor threshhold to 0.99.

Section 27.6

[▼](javascript:collapse('expandText17',%20'collapseText17');)27.6.1

**What is load factor? Assume the hash table has the initial size 4 and its load factor is 0.5; show the hash table after inserting entries with the keys 34,** 29, 53, 44, 120, 39, 45, and 40, using linear probing.

Load factor is the ratio between the number of elements in the hash table and the hash table size. It measures how full in the hash table. The other part is omitted. Use the animation https://liveexample.pearsoncmg.com/dsanimation/LinearProbingeBook.html to verify your anwser. To start, click the Remove All button to empty the hash table and set the load factor threshhold to 0.5.

[▼](javascript:collapse('expandText18',%20'collapseText18');)27.6.2

**Assume the hash table has the initial size 4 and its load factor is 0.5; show the hash table after inserting entries with the keys 34, 29, 53, 44, 120, 39, 45, and 40, using quadratic probing.**

See the text

[▼](javascript:collapse('expandText19',%20'collapseText19');)27.6.3

**Assume the hash table has the initial size 4 and its load factor is 0.5; show the hash table after inserting entries with the keys 34, 29, 53, 44, 120, 39, 45, and 40, using separate chaining.**

See the text

[▼](javascript:collapse('expandText20',%20'collapseText20');)27.7.1

**What is 1 << 30 in line 8 in Listing 27.2? What are the integers resulted from 1 << 1, 1 << 2, and 1 << 3?**

230   
21   
22   
23

[▼](javascript:collapse('expandText21',%20'collapseText21');)27.7.2

**What are the integers resulted from 32 >> 1, 32 >> 2, 32 >> 3, and 32 >> 4?**

32 >> 1 is 16   
32 >> 2 is 8   
32 >> 3 is 4   
32 >> 4 is 2

[▼](javascript:collapse('expandText22',%20'collapseText22');)27.7.3

**In Listing 27.2, will the program work if LinkedList is replaced by ArrayList? In Listing 27.2, how do you replace the code in lines 56-59 using one line of code?**

Yes.  
The second part of this question: return get(key) != null

[▼](javascript:collapse('expandText23',%20'collapseText23');)27.7.4

**Describe how the put(key, value) method is implemented in the MyHashMap class.**

See the text.

[▼](javascript:collapse('expandText24',%20'collapseText24');)27.7.5

**In Listing 27.2, the supplementalHash method is declared static, can the hash method be declared static?**

No. The reason: The hash method implementation uses capacity. capacity is an instance data field in MyHashMap.

[▼](javascript:collapse('expandText25',%20'collapseText25');)27.7.6

**Show the output of the following code.**

MyMap<String, String> map = new MyHashMap<>();

map.put("Texas", "Dallas");

map.put("Oklahoma", "Norman");

map.put("Texas", "Austin");

map.put("Oklahoma", "Tulsa");

System.out.println(map.get("Texas"));

System.out.println(map.size());

Austin   
2

[▼](javascript:collapse('expandText26',%20'collapseText26');)27.7.7

**If x is a negative int value, will x & (capacity - 1) be negative?**

capacity is a 32-bit int value. Since the maximum capacity for the hash table is 230 (line 8 in MyHashMap.java), the first bit in capacity is 0. Thus, the first bit in the result of x & (capacity - 1) is 0. Therefore, x & (capacity - 1) is positive, even though x is negative.

Section 27.8

[▼](javascript:collapse('expandText27',%20'collapseText27');)27.8.1

**Why can you use a foreach loop to traverse the elements in a set?**

Because MySet implements Iterable.

[▼](javascript:collapse('expandText28',%20'collapseText28');)27.8.2

**Describe how the add(e) method is implemented in the MyHashSet class.**

See the text

[▼](javascript:collapse('expandText29',%20'collapseText29');)27.8.3

**Can lines 100-103 in Listing 27.4 be removed?**

Yes. It is redundant, because contains(e) is true after lines 94-95 and table[bucketIndex] is not null.

[▼](javascript:collapse('expandText30',%20'collapseText30');)27.8.4

**Implement the remove() method in lines 150-152?**

@Override /\*\* Remove the element returned by the last next() \*/

public void remove() {

if (current == 0) // next() has not been called yet

throw new IllegalStateException();

set.remove(list.get(--current));

list.remove(current); // Remove the element from the list

}

**Chapter 28 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)28.2.1

**What is the famous Seven Bridges of Königsberg problem?**

See §28.1.

[▼](javascript:collapse('expandText1',%20'collapseText1');)28.2.2

**What is a graph? Explain the following terms: undirected graph, directed graph, weighted graph, degree of a vertex, parallel edge, simple graph, complete graph, connected graph, cycle, subgraph, tree, and spanning tree.**

See §28.2.

[▼](javascript:collapse('expandText2',%20'collapseText2');)28.2.3

**How many edges are in a complete graph with 5 vertices? How many edges are in a tree of 5 vertices?**

A complete graph of n vertices has 10 edges. A tree always has 4 edges.

[▼](javascript:collapse('expandText3',%20'collapseText3');)28.2.4

**How many edges are in a complete graph with n vertices? How many edges are in a tree of n vertices?**

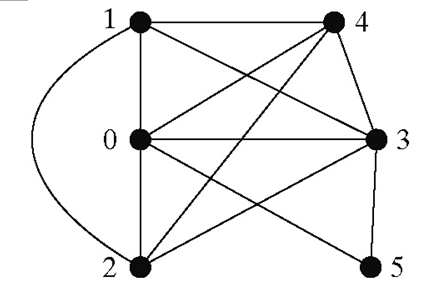
A complete graph of n vertices has n(n-1)/2 edges. A tree always has n-1 edges.

[▼](javascript:collapse('expandText4',%20'collapseText4');)28.3.1

**How do you represent vertices in a graph? How do you represent edges using an edge array? How do you represent an edge using an edge object? How do you represent edges using an adjacency matrix? How do you represent edges using adjacency lists?**

See §28.3.

[▼](javascript:collapse('expandText5',%20'collapseText5');)28.3.2

Represent the following graph using an edge array, a list of edge objects, an adjacency matrix, an adjacency vertex list, and an adjacency edge list, respectively.   


Edge array:

int[][] edges = {

{0, 1}, {0, 2}, {0, 3}, {0, 4}, {0, 5},

{1, 0}, {1, 2}, {1, 3}, {1, 4},

{2, 0}, {2, 1}, {2, 3}, {2, 4},

{3, 0}, {3, 1}, {3, 2}, {3, 4}, {3, 5},

{4, 0}, {4, 1}, {4, 2}, {4, 3},

{5, 0}, {5, 3}

}

List of edge objects:

java.util.ArrayList<Edge> list = new java.util.ArrayList<Edge>();

list.add(new Edge(0, 1));

list.add(new Edge(0, 2));

list.add(new Edge(0, 3));

list.add(new Edge(0, 4));

list.add(new Edge(0, 5));

…

…

…

Adjacency matrix:

int[][] adjacencyMatrix = {

{0, 1, 1, 1, 1, 1}, // node 0

{1, 0, 1, 1, 1, 0}, // node 1

{1, 1, 0, 1, 1, 0}, // node 2

{1, 1, 1, 0, 1, 1}, // node 3

{1, 1, 1, 1, 0, 0}, // node 4

{1, 0, 0, 1, 0, 0} // node 5

};

Adjacency list:

LinkedList<Integer> list[] = new LinkedList<>();

list[0].add(1); list[0].add(2); list[0].add(3); list[0].add(4);

list[0].add(5);

list[1].add(0); list[1].add(2); list[1].add(3); list[1].add(4);

list[2].add(0); list[2].add(1); list[2].add(3); list[2].add(4);

list[3].add(0); list[3].add(1); list[3].add(2); list[3].add(4);

list[3].add(5);

list[4].add(0); list[4].add(1); list[4].add(2); list[4].add(3);

list[5].add(0); list[5].add(3);

Adjacency edge list:

ArrayList<ArrayList<Edge>> list = new ArrayList<>();

list.add(new ArrayList<>());

list.get(0).add(new Edge(0, 1));

list.get(0).add(new Edge(0, 2));

list.get(0).add(new Edge(0, 3));

list.get(0).add(new Edge(0, 4));

list.get(0).add(new Edge(0, 5));

list.add(new ArrayList<>());

list.get(1).add(new Edge(1, 0));

list.get(1).add(new Edge(1, 2));

list.get(1).add(new Edge(1, 3));

list.get(1).add(new Edge(1, 4));

list.add(new ArrayList<>());

list.get(2).add(new Edge(2, 1));

list.get(2).add(new Edge(2, 2));

list.get(2).add(new Edge(2, 3));

list.get(2).add(new Edge(2, 4));

list.add(new ArrayList<>());

list.get(3).add(new Edge(3, 0));

list.get(3).add(new Edge(3, 1));

list.get(3).add(new Edge(3, 2));

list.get(3).add(new Edge(3, 4));

list.get(3).add(new Edge(3, 5));

list.add(new ArrayList<>());

list.get(4).add(new Edge(4, 0));

list.get(4).add(new Edge(4, 1));

list.get(4).add(new Edge(4, 2));

list.get(4).add(new Edge(4, 3));

list.add(new ArrayList<>());

list.get(5).add(new Edge(5, 0));

list.get(5).add(new Edge(5, 3));

[▼](javascript:collapse('expandText6',%20'collapseText6');)28.4.1

**Describe the relationships between Graph, UnweightedGraph, and WeightedGraph.**

Graph is an interface that defines the common features for graphs. UnweightedGraph is a concreate graph that represents unweighted graphs. WeightedGraph is a subtype of UnweightedGraph that represents weighted graphs.

[▼](javascript:collapse('expandText7',%20'collapseText7');)28.4.2

**For the code in Listing 28.2, TestGraph.java, what is graph1.getIndex("Seattle")? What is graph1.getDegree(5)? What is graph1.getVertex(4)?**

**What is graph1.getIndex("Seattle")? 0   
What is graph1.getDegree(5)? 5   
What is graph1.getVertex(4)? Kansas City**

[▼](javascript:collapse('expandText8',%20'collapseText8');)28.4.3

**Show the output of the following code.**

public class Test {

public static void main(String[] args) {

Graph<Character> graph = new UnweightedGraph<>();

graph.addVertex('U');

graph.addVertex('V');

int indexForU = graph.getIndex('U');

int indexForV = graph.getIndex('V');

System.out.println("indexForU is " + indexForU);

System.out.println("indexForV is " + indexForV);

System.out.println("Degree of U is " +

graph.getDegree(indexForU));

System.out.println("Degree of V is " +

graph.getDegree(indexForV));

}

}

indexForU is 0   
indexForV is 1   
Degree of U is 1   
Degree of V is 0

[▼](javascript:collapse('expandText9',%20'collapseText9');)28.4.4

**What will getIndex(v) return if v is not in the graph? What happens to getVertex(index) if index is not in the graph? What happens to addVertex(v) if v is already in the graph? What happens to addEdge(u, v) if u or v is not in the graph?**

getIndex(v) returns -1 if v is not in the graph.   
What happens to getVertex(index) if index is not in the graph? IndexOutBoundsException   
What happens to addVertex(v) if v is already in the graph? v is not added to graph and the method returns false.   
What happens to addEdge(u, v) if u or v is not in the graph? Throws IlleagalArgumentException.

Section 28.5

[▼](javascript:collapse('expandText10',%20'collapseText10');)28.5.1

**Will Listing 28.7 DisplayUSMap.java work, if the code in lines 38-42 in Listing 28.6 GraphView.java is replaced by the following code?**

if (i < v) {

double x2 = graph.getVertex(v).getX();

double y2 = graph.getVertex(v).getY();

// Draw an edge for (i, v)

getChildren().add(new Line(x1, y1, x2, y2));

}

Yes. Because the graph in Figure 18.1 is undirected. The edge from v to u and the edge from u to v are both contained in the graph. You need to draw just one such edge.

[▼](javascript:collapse('expandText11',%20'collapseText11');)28.5.2

**For the graph1 object created in Listing 28.1, TestGraph.java, can you create a GraphView object as follows?**

GraphView view = new GraphView(graph1);

No, because graph1 is not an instance of Displayable.

Section 28.6

[▼](javascript:collapse('expandText12',%20'collapseText12');)28.6.1

**Does UnweightedGraph<V>.SearchTree implement the Tree interface defined in Listing 25.3 Tree.java?**

No.

[▼](javascript:collapse('expandText13',%20'collapseText13');)28.6.2

**What method do you use to find the parent of a vertex in the tree?**

The getParent(int index) method.

Section 28.7

[▼](javascript:collapse('expandText14',%20'collapseText14');)28.7.1

**What is depth-first search?**

See the text.

[▼](javascript:collapse('expandText15',%20'collapseText15');)28.7.2

**Draw a DFS tree for the graph in Figure 28.3b starting from node A.**

See the text.

[▼](javascript:collapse('expandText16',%20'collapseText16');)28.7.3

**Draw a DFS tree for the graph in Figure 28.1 starting from vertex Atlanta.**

See the text.

[▼](javascript:collapse('expandText17',%20'collapseText17');)28.7.4

**What is the return type from invoking dfs(v)?**

The return is an instance of AbstractGraph.Tree.

[▼](javascript:collapse('expandText18',%20'collapseText18');)28.7.5

**The depth-first search algorithm described in Listing 28.8 uses recursion. Alternatively, you can use a stack to implement it, as shown below. Point out the error in this algorithm and give a correct algorithm.**

// Wrong version

Tree dfs(vertex v) {

push v into the stack;

mark v visited;

while (the stack is not empty) {

pop a vertex, say u, from the stack

visit u;

for each neighbor w of u

if (w has not been visited)

push w into the stack;

}

}

There are several problems in this algorithm: 1. There is a possibility that a vertex may be pushed into the stack more than once. Can you give such an example? 2. The neighbors of a vertex is pushed into the stack. You should only push one neighbor into the stack and continue to search for the next vertex from this new vertex. The correct algorithm is given below:

\* Input: G = (V, E) and a starting vertex v

\* Output: A DFS tree rooted at v

\*

\* Tree dfs(v) {

\* push v to the stack;

\* mark x visited;

\* while (the stack is not empty) {

\* peek a vertex from the stack, say x;

\* if (x still has an unvisited neighbor, say y) {

\* parent[y] = x;

\* push y into the stack;

\* mark y visited;

\* }

\* else {

\* pop a vertex from the stack;

\* }

\* }

\* }

Section 28.8

[▼](javascript:collapse('expandText19',%20'collapseText19');)28.8.1

**How is a graph created for the connected circles problem?**

Each circle corresponds to a vertex and the vertices are connected if the corresponding two circles overlap.

[▼](javascript:collapse('expandText20',%20'collapseText20');)28.8.2

**When you click the mouse inside a circle, does the program create a new circle?**

No.

[▼](javascript:collapse('expandText21',%20'collapseText21');)28.8.3

**How does the program know if all circles are connected?**

This is done by performing a DFS. If the DFS tree contains the same number of vertices as the number of vertices, the graph is connected and all the circles are connected.

[▼](javascript:collapse('expandText22',%20'collapseText22');)28.9.1

**What is the return type from invoking bfs(v)?**

The return is an instance of AbstractGraph.Tree.

[▼](javascript:collapse('expandText23',%20'collapseText23');)28.9.2

**What is breadth-first search?**

See the text.

[▼](javascript:collapse('expandText24',%20'collapseText24');)28.9.3

**Draw a BFS tree for the graph in Figure 28.3b starting from node A.**

See the text.

[▼](javascript:collapse('expandText25',%20'collapseText25');)28.9.4

**Draw a BFS tree for the graph in Figure 28.1 starting from vertex Atlanta.**

See the text.

[▼](javascript:collapse('expandText26',%20'collapseText26');)28.9.5

**Prove that the path between the root and any node in the BFS tree is the shortest path between the root and the node.**

You may prove it using induction on the path length.

Section 28.10

[▼](javascript:collapse('expandText27',%20'collapseText27');)28.10.1

**How are the nodes created for the graph in NineTailModel?**

See the text.

[▼](javascript:collapse('expandText28',%20'collapseText28');)28.10.2

**How are the edges created for the graph in NineTailModel?**

See the text.

[▼](javascript:collapse('expandText29',%20'collapseText29');)28.10.3

**What is returned after invoking getIndex("HTHTTTHHH".toCharArray()) in Listing 28.13? What is returned after invoking getNode(46) in Listing** 28.13?

getIndex("HTHTTTHHH".toCharArray()) returns 184.  
getNode(46) returns HHHTHTTTH

[▼](javascript:collapse('expandText30',%20'collapseText30');)28.10.4

**If lines 26 and 27 are swapped in Listing 28.13, NineTailModel.java, will the program work? Why not?**

It will not work, because node is changed in the getFlippedNode(node, k).

**Chapter 29 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)29.2.1

**For the code WeightedEdge edge = new WeightedEdge(1, 2, 3.5), what is edge.u, edge.v, and edge.weight?**

edge.u is 1, edge.v is 2, and edge.weight is 3.5

[▼](javascript:collapse('expandText1',%20'collapseText1');)29.2.2

**What is the output of the following code?**

List<WeightedEdge> list = new ArrayList<>();

list.add(new WeightedEdge(1, 2, 3.5));

list.add(new WeightedEdge(2, 3, 4.5));

WeightedEdge e = java.util.Collections.max(list);

System.out.println(e.u);

System.out.println(e.v);

System.out.println(e.weight);

The output is

2

3

4.5

Section 29.3

[▼](javascript:collapse('expandText2',%20'collapseText2');)29.3.1

**If a priority queue is used to store weighted edges, what is the output of the following code?**

PriorityQueue<WeightedEdge> q = new PriorityQueue<>();

q.offer(new WeightedEdge(1, 2, 3.5));

q.offer(new WeightedEdge(1, 6, 6.5));

q.offer(new WeightedEdge(1, 7, 1.5));

System.out.println(q.poll().weight);

System.out.println(q.poll().weight);

System.out.println(q.poll().weight);

The output is

1.5

3.5

6.5

[▼](javascript:collapse('expandText3',%20'collapseText3');)29.3.2

**If a priority queue is used to store weighted edges, what is wrong in the following code? Fix it and show the output.**

List<PriorityQueue<WeightedEdge>> queues = new ArrayList<>();

queues.get(0).offer(new WeightedEdge(0, 2, 3.5));

queues.get(0).offer(new WeightedEdge(0, 6, 6.5));

queues.get(0).offer(new WeightedEdge(0, 7, 1.5));

queues.get(1).offer(new WeightedEdge(1, 0, 3.5));

queues.get(1).offer(new WeightedEdge(1, 5, 8.5));

queues.get(1).offer(new WeightedEdge(1, 8, 19.5));

System.out.println(queues.get(0).peek()

.compareTo(queues.get(1).peek()));

The code is wrong because there is no queues.get(0). You need to first create and add a queue into queues using the following statements:

queues.add(new PriorityQueue<WeightedEdge>());

queues.add(new PriorityQueue<WeightedEdge>());

After the fix, the output is -1.

[▼](javascript:collapse('expandText4',%20'collapseText4');)29.3.3

**Show the output of the following code.**

public class Test {

public static void main(String[] args) throws Exception {

WeightedGraph<Character> graph = new WeightedGraph<>();

graph.addVertex('U');

graph.addVertex('V');

int indexForU = graph.getIndex('U');

int indexForV = graph.getIndex('V');

System.out.println("indexForU is " + indexForU);

System.out.println("indexForV is " + indexForV);

graph.addEdge(indexForU, indexForV, 2.5);

System.out.println("Degree of U is " +

graph.getDegree(indexForU));

System.out.println("Degree of V is " +

graph.getDegree(indexForV));

System.out.println("Weight of UV is " +

graph.getWeight(indexForU, indexForV));

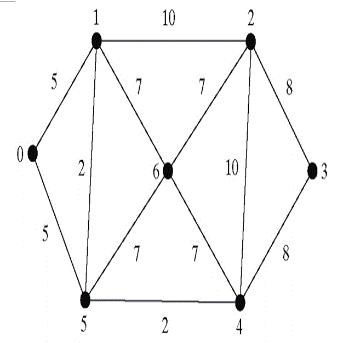
}

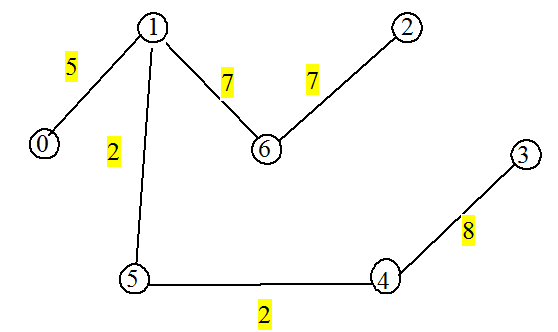
}

indexForU is 0   
indexForV is 1   
Degree of U is 1   
Degree of V is 0   
Weight of UV is 2.5

Section 29.4

[▼](javascript:collapse('expandText5',%20'collapseText5');)29.4.1

**Find a minimum spanning tree for the following graph.**



[▼](javascript:collapse('expandText6',%20'collapseText6');)29.4.2

**Is a minimum spanning tree unique if all edges have different weights?**

Yes.

[▼](javascript:collapse('expandText7',%20'collapseText7');)29.4.3

**If you use an adjacency matrix to represent weighted edges, what will be the time complexity for Prim's algorithm?**

O(n^2logn), n is the number of vertices.

[▼](javascript:collapse('expandText8',%20'collapseText8');)29.4.4

**What happens to the getMinimumSpanningTree() method in WeightedGraph if the graph is not connected? Verify your answer by writing a test program that creates an unconnected graph and invokes the getMinimumSpanningTree() method.**

Line 95 in WeightedGraph.java, the loop

while (T.size() < numberOfVertices) {

continues if T.size() < numberOfVertices. If the graph is not connected, the v will be set to -1 in line 98, if no edges are found to connect vertices between T and V - T. In this case, the statement

if (v != -1)

T.add(v); // Add a new vertex to the tree

else

break; // The tree is not connected, a partial MST is found

causes the while loop in line 95 to end.

[▼](javascript:collapse('expandText9',%20'collapseText9');)29.4.5

**Show the output of the following code:**

public class Test {

public static void main(String[] args) {

WeightedGraph<Character> graph = new WeightedGraph<>();

graph.addVertex('U');

graph.addVertex('V');

graph.addVertex('X');

int indexForU = graph.getIndex('U');

int indexForV = graph.getIndex('V');

int indexForX = graph.getIndex('X');

System.out.println("indexForU is " + indexForU);

System.out.println("indexForV is " + indexForV);

System.out.println("indexForX is " + indexForV);

graph.addEdge(indexForU, indexForV, 3.5);

graph.addEdge(indexForV, indexForU, 3.5);

graph.addEdge(indexForU, indexForX, 2.1);

graph.addEdge(indexForX, indexForU, 2.1);

graph.addEdge(indexForV, indexForX, 3.1);

graph.addEdge(indexForX, indexForV, 3.1);

WeightedGraph<Character>.MST mst

= graph.getMinimumSpanningTree();

graph.printWeightedEdges();

System.out.println(mst.getTotalWeight());

mst.printTree();

}

}

indexForU is 0

indexForV is 1

indexForX is 1

U (0): (0, 1, 3.5) (0, 2, 2.1)

V (1): (1, 0, 3.5) (1, 2, 3.1)

X (2): (2, 0, 2.1) (2, 1, 3.1)

5.2

Root is: U

Edges: (X, V) (U, X)

Section 29.5

[▼](javascript:collapse('expandText10',%20'collapseText10');)29.5.1

**Trace Dijkstra's algorithm for finding shortest paths from Boston to all other cities in Figure 29.1.**

See the text.

[▼](javascript:collapse('expandText11',%20'collapseText11');)29.5.2

**Is a shortest path between two vertices unique if all edges have different weights?**

No.

[▼](javascript:collapse('expandText12',%20'collapseText12');)29.5.3

**If you use an adjacency matrix to represent weighted edges, what would be the time complexity for Dijkstra's algorithm?**

O(n^2logn), n is the number of vertices.

[▼](javascript:collapse('expandText13',%20'collapseText13');)29.5.4

**What happens to the getShortestPath() method in WeightedGraph if the source vertex cannot reach all vertieces in the graph? Verify your answer by** writing a test program that creates an unconnected graph and invoke the getShortestPath() method.

Line 185 in WeightedGraph.java, the loop

while (T.size() < numberOfVertices) {

continues if T.size() < numberOfVertices. If the graph is not connected, the v is set to -1 in line 185, if no edges are found to connect vertices between T and V - T, -1 will add to T in line 208. The statement

if (v != -1)

T.add(v); // Add a new vertex to the tree

else

break; // The tree is not connected, a partial MST is found

causes the while loop in line 185 to end.

[▼](javascript:collapse('expandText14',%20'collapseText14');)29.5.5

**If there is no path from vertex v to the source vertex, what will be cost[v]?**

cost[v] will be infinity.

[▼](javascript:collapse('expandText15',%20'collapseText15');)29.5.6

**Assume that the graph is connected; will the getShortestPath method find the shortest paths correctly if lines 159-161 in WeightedGraph are deleted?**

No. cost[i] will be zero for i.

[▼](javascript:collapse('expandText16',%20'collapseText16');)29.5.7

**Show the output of the following code:**

public class Test {

public static void main(String[] args) {

WeightedGraph<Character> graph = new WeightedGraph<>();

graph.addVertex('U');

graph.addVertex('V');

graph.addVertex('X');

int indexForU = graph.getIndex('U');

int indexForV = graph.getIndex('V');

int indexForX = graph.getIndex('X');

System.out.println("indexForU is " + indexForU);

System.out.println("indexForV is " + indexForV);

System.out.println("indexForX is " + indexForV);

graph.addEdge(indexForU, indexForV, 3.5);

graph.addEdge(indexForV, indexForU, 3.5);

graph.addEdge(indexForU, indexForX, 2.1);

graph.addEdge(indexForX, indexForU, 2.1);

graph.addEdge(indexForV, indexForX, 3.1);

graph.addEdge(indexForX, indexForV, 3.1);

WeightedGraph<Character>.ShortestPathTree tree =

graph.getShortestPath(1);

graph.printWeightedEdges();

tree.printTree();

}

}

indexForU is 0

indexForV is 1

indexForX is 1

U (0): (0, 1, 3.5) (0, 2, 2.1)

V (1): (1, 0, 3.5) (1, 2, 3.1)

X (2): (2, 0, 2.1) (2, 1, 3.1)

Root is: V

Edges: (V, U) (V, X)

Section 29.6

[▼](javascript:collapse('expandText17',%20'collapseText17');)29.6.1

**Why is the tree data field in NineTailModel in Listing 28.13 defined protected?**

The tree data field in NineTailModel is accessed in WeightedTailModel. A new tree is created in WeightedTailModel.

[▼](javascript:collapse('expandText18',%20'collapseText18');)29.6.2

**How are the nodes created for the graph in WeightedNineTailModel?**

See the text.

[▼](javascript:collapse('expandText19',%20'collapseText19');)29.6.3

**How are the edges created for the graph in WeightedNineTailModel?**

See the text.

**Chapter 30 Check Point Questions**

[▼](javascript:collapse('expandText0',%20'collapseText0');)30.1.1

**What are the benefits of using aggregate operations on collection streams for processing data?**

Using aggregate operations on collection streams simplifies coding and improves performance.

Section 30.2

[▼](javascript:collapse('expandText1',%20'collapseText1');)30.2.1

**Show the output of the following code?**

Character[] chars = {'D', 'B', 'A', 'C'};

System.out.println(Stream.of(chars).sorted().findFirst().get());

System.out.println(Stream.of(chars).sorted(

java.util.Comparator.reverseOrder()).findFirst().get());

System.out.println(Stream.of(chars)

.limit(2).sorted().findFirst().get());

System.out.println(Stream.of(chars).distinct()

.skip(2).filter(e -> e > 'A').findFirst().get());

System.out.println(Stream.of(chars)

.max(Character::compareTo).get());

System.out.println(Stream.of(chars)

.max(java.util.Comparator.reverseOrder()).get());

System.out.println(Stream.of(chars)

.filter(e -> e > 'A').findFirst().get());

System.out.println(Stream.of(chars)

.allMatch(e -> e >= 'A'));

System.out.println(Stream.of(chars)

.anyMatch(e -> e > 'F'));

System.out.println(Stream.of(chars)

.noneMatch(e -> e > 'F'));

Stream.of(chars).map(e -> e + "").map(e -> e.toLowerCase())

.forEach(System.out::println);

Object[] temp = Stream.of(chars).map(e -> e + "Y")

.map(e -> e.toLowerCase()).sorted().toArray();

System.out.println(java.util.Arrays.toString(temp));

A

D

B

C

D

A

D

true

false

true

d

b

a

c

[ay, by, cy, dy]

[▼](javascript:collapse('expandText2',%20'collapseText2');)30.2.2

**What is wrong in the following code?**

Character[] chars = {'D', 'B', 'A', 'C'};

Stream<Character> stream = Stream.of(chars).sorted();

System.out.println(stream.findFirst());

System.out.println(stream.skip(2).findFirst());

A stream can only have one terminal operation. Once you apply stream.findFirst() . the stream is destroyed.

[▼](javascript:collapse('expandText3',%20'collapseText3');)30.2.3

**Rewrite (a) using a method reference and an anonymous inner class and (b) using lambda expression and an anonymous inner class.**

(a) sorted((s1, s2) -> s1.compareToIgnoreCase(s2))

(b) forEach(System.out::println)

(a)

Use a method reference:

sorted(String::compareToIgnoreCase);

Use an anonymous inner class:

sorted(new Comparator<String>() {

@Override

public int compare(String s1, String s2) {

return s1.compareToIgnoreCase(s2);

}

});

(b)

Use a lambda expression:

forEach(e -> System.out.println())

Use an anonymous inner class:

forEach(

new java.util.function.Consumer<String>() {

public void accept(String e) {

System.out.println();

}

}

)

[▼](javascript:collapse('expandText4',%20'collapseText4');)30.2.4

**Given a map of the type Map<String, Double>, write an expression that returns the sum of all the values in map. For example, if the map contains** {"john", 1.5} and {"Peter", 1.1}, the sum is 2.6.

map.entrySet().stream().mapToDouble(e -> e.getValue()).sum()

[▼](javascript:collapse('expandText5',%20'collapseText5');)30.3.1

**Show the output of the following code?**

int[] numbers = {1, 4, 2, 3, 1};

System.out.println(IntStream.of(numbers)

.sorted().findFirst().getAsInt());

System.out.println(IntStream.of(numbers)

.limit(2).sorted().findFirst().getAsInt());

System.out.println(IntStream.of(numbers).distinct()

.skip(1).filter(e -> e > 2).sum());

System.out.println(IntStream.of(numbers).distinct()

.skip(1).filter(e -> e > 2).average().getAsDouble());

System.out.println(IntStream.of(numbers).max().getAsInt());

System.out.println(IntStream.of(numbers).max().getAsInt());

System.out.println(IntStream.of(numbers)

.filter(e -> e > 1).findFirst().getAsInt());

System.out.println(IntStream.of(numbers)

.allMatch(e -> e >= 1));

System.out.println(IntStream.of(numbers)

.anyMatch(e -> e > 4));

System.out.println(IntStream.of(numbers).noneMatch(e -> e > 4));

IntStream.of(numbers).mapToObj(e -> (char)(e + 50))

.forEach(System.out::println);

Object[] temp = IntStream.of(numbers)

.mapToObj(e -> (char)(e + 'A')).toArray();

System.out.println(java.util.Arrays.toString(temp));

1

1

7

3.5

4

4

4

true

false

true

3

6

4

5

3

[B, E, C, D, B]

[▼](javascript:collapse('expandText6',%20'collapseText6');)30.3.2

**What is wrong in the following code?**

int[] numbers = {1, 4, 2, 3, 1};

DoubleSummaryStatistics stats =

DoubleStream.of(numbers).summaryStatistics();

System.out.printf("The summary of the stream is\n%-10s%10d\n" +

"%-10s%10.2f\n%-10s%10.2f\n%-10s%10.2f\n%-10s%10.2f\n",

" Count:", stats.getCount(), " Max:", stats.getMax(),

" Min:", stats.getMin(), " Sum:", stats.getSum(),

" Average:", stats.getAverage());

numbers is an int array, you have to use IntStream.of(numbers) rather than DoubleStream.of(numbers).

[▼](javascript:collapse('expandText7',%20'collapseText7');)30.3.3

**Rewrite the following code that maps an int to a Character using an anonymous inner class?**

mapToObj(e -> (char)(e + 50))

mapToObj(

new java.util.function.IntFunction<Character>() {

public Character apply(int e) {

return (char)(e + 50);

}

}

)

[▼](javascript:collapse('expandText8',%20'collapseText8');)30.3.4

**Show the output of the following code.**

int[][] m = {{1, 2}, {3, 4}, {5, 6}};

System.out.println(Stream.of(m)

.mapToInt(e -> IntStream.of(e).sum()).sum());

21

[▼](javascript:collapse('expandText9',%20'collapseText9');)30.3.5

**Given an array names in Listing 30.1, write the code to display the total number of characters in names.**

System.out.println("The number of characters in array names is " +

Stream.of(names).mapToInt(e -> e.length()).sum());

Section 30.4

[▼](javascript:collapse('expandText10',%20'collapseText10');)30.4.1

**What is a stateless method? What is a stateful method?**

A stateless method can apply to the elements in the stream independent from the others. A stateful method must consider all the elements in order to produce a result.

[▼](javascript:collapse('expandText11',%20'collapseText11');)30.4.2

**How do you create a parallel stream?**

You can create a parallel stream by invoking the parallel() method on a stream or invoking the parallelStream() method from a collection object such as a list or a set.

[▼](javascript:collapse('expandText12',%20'collapseText12');)30.4.3

**Suppose names is a set of strings, which of the following two streams is better?**

Object[] s = set.parallelStream().filter(e -> e.length() > 3)

.sorted().toArray();

Object[] s = set.parallelStream().sorted()

.filter(e -> e.length() > 3).toArray();

The former is better than the latter because the stream size is smaller after applying the filter method. This will make the sorted() method to run faster.

[▼](javascript:collapse('expandText13',%20'collapseText13');)30.4.4

**What will be the output of the following code?**

int[] values = {3, 4, 1, 5, 20, 1, 3, 3, 4, 6};

System.out.print("The values are ");

IntStream.of(values)

.forEach(e -> System.out.print(e + " "));

The values are 3 4 1 5 20 1 3 3 4 6

[▼](javascript:collapse('expandText14',%20'collapseText14');)30.4.5

**What will be the output of the following code?**

int[] values = {3, 4, 1, 5, 20, 1, 3, 3, 4, 6};

System.out.print("The values are ");

IntStream.of(values).parallel()

.forEach(e -> System.out.print(e + " "));

The output is unpredictable due to using a parallel stream with forEach method.

[▼](javascript:collapse('expandText15',%20'collapseText15');)30.4.6

**Write a statement to obtain an array of 1000 random double values between 0.0 and 1.0, excluding 1.0.**

Random r = new Random();

double[] numbers = r.doubles(1000, 0.0, 1.0).toArray();

Section 30.5

[▼](javascript:collapse('expandText16',%20'collapseText16');)30.5.1

**Show the output of the following code.**

int[] values = {1, 2, 3, 4};

System.out.println(IntStream.of(values)

.reduce(0, (e1, e2) -> e1 + e2));

System.out.println(IntStream.of(values)

.reduce(1, (e1, e2) -> e1 \* e2));

System.out.println(IntStream.of(values).map(e -> e \* e)

.reduce(0, (e1, e2) -> e1 + e2));

System.out.println(IntStream.of(values).mapToObj(e -> "" + e)

.reduce((e1, e2) -> e1 + " " + e2).get());

System.out.println(IntStream.of(values).mapToObj(e -> "" + e)

.reduce((e1, e2) -> e1 + ", " + e2).get());

10

24

30

1 2 3 4

1, 2, 3, 4

[▼](javascript:collapse('expandText17',%20'collapseText17');)30.5.2

**Show the output of the following code.**

int[][] m = {{1, 2}, {3, 4}, {5, 6}};

System.out.println(Stream.of(m)

.map(e -> IntStream.of(e).reduce(1, (e1, e2) -> e1 \* e2))

.reduce(1, (e1, e2) -> e1 \* e2));

720

[▼](javascript:collapse('expandText18',%20'collapseText18');)30.5.3

**Show the output of the following code.**

int[][] m = {{1, 2}, {3, 4}, {5, 6}, {1, 3}};

Stream.of(m).map(e -> IntStream.of(e))

.reduce((e1, e2) -> IntStream.concat(e1, e2))

.get().distinct()

.forEach(e -> System.out.print(e + " "));

1 2 3 4 5 6

[▼](javascript:collapse('expandText19',%20'collapseText19');)30.5.4

**Show the output of the following code.**

int[][] m = {{1, 2}, {3, 4}, {5, 6}, {1, 3}};

System.out.println(

Stream.of(m).map(e -> IntStream.of(e))

.reduce((e1, e2) -> IntStream.concat(e1, e2))

.get().distinct().mapToObj(e -> e + "")

.reduce((e1, e2) -> e1 + ", " + e2).get());

1, 2, 3, 4, 5, 6, 1, 3

Section 30.6

[▼](javascript:collapse('expandText20',%20'collapseText20');)30.6.1

**Show the output of the following code.**

int[] values = {1, 2, 3, 4, 1};

List<Integer> list = IntStream.of(values).mapToObj(e -> e)

.collect(Collectors.toList());

System.out.println(list);

Set<Integer> set = IntStream.of(values).mapToObj(e -> e)

.collect(Collectors.toSet());

System.out.println(set);

Map<Integer, Integer> map = IntStream.of(values).distinct()

.mapToObj(e -> e)

.collect(Collectors.toMap(e -> e, e -> e.hashCode()));

System.out.println(map);

System.out.println(

IntStream.of(values).mapToObj(e -> e)

.collect(Collectors.summingInt(e -> e)));

System.out.println(

IntStream.of(values).mapToObj(e -> e)

.collect(Collectors.averagingDouble(e -> e)));

[1, 2, 3, 4, 1]

[1, 2, 3, 4]

{1=1, 2=2, 3=3, 4=4}

11

2.2

[▼](javascript:collapse('expandText21',%20'collapseText21');)30.7.1

**Show the output of the following code.**

int[] values = {1, 2, 2, 3, 4, 2, 1};

IntStream.of(values).mapToObj(e -> e).collect(

Collectors.groupingBy(e -> e, TreeMap::new,

Collectors.counting())).

forEach((k, v) -> System.out.println(k + " occurs " + v

+ (v > 1 ? " times " : " time ")));

IntStream.of(values).mapToObj(e -> e).collect(

Collectors.groupingBy(e -> e, TreeMap::new,

Collectors.summingInt(e -> e))).

forEach((k, v) -> System.out.println(k + ": " + v));

MyStudent[] students = {

new MyStudent("John", "Johnson", "CS", 23, 89.2),

new MyStudent("Susan", "Johnson", "Math", 21, 89.1),

new MyStudent("John", "Peterson", "CS", 21, 92.3),

new MyStudent("Kim", "Yao", "Math", 22, 87.3),

new MyStudent("Jeff", "Johnson", "CS", 23, 78.5)};

Stream.of(students)

.sorted(Comparator.comparing(MyStudent::getLastName)

.thenComparing(MyStudent::getFirstName))

.forEach(e -> System.out.println(e.getLastName() + ", " +

e.getFirstName()));

Stream.of(students).collect(Collectors.

groupingBy(MyStudent::getAge, TreeMap::new,

Collectors.averagingDouble(MyStudent::getScore))).

forEach((k, v) -> System.out.printf("%10s%10.2f\n", k, v));

1 occurs 2 times

2 occurs 3 times

3 occurs 1 time

4 occurs 1 time

1: 2

2: 6

3: 3

4: 4

Johnson, Jeff

Johnson, John

Johnson, Susan

Peterson, John

Yao, Kim

21 90.70

22 87.30

23 83.85

Section 30.8

[▼](javascript:collapse('expandText22',%20'collapseText22');)30.8.1

**Can the following code be used to replace line 19 in Listing 30.7?**

DoubleStream.of(numbers).filter(e -> e >

DoubleStream.of(numbers).average()).count());

No. You have to use:

DoubleStream.of(numbers).filter(e -> e >

DoubleStream.of(numbers).average().getAsDouble()).count());

[▼](javascript:collapse('expandText23',%20'collapseText23');)30.8.2

**Can the following code be used to replace lines 15-16 in Listing 30.8?**

Stream.of(chars).forEach(e -> {

int count = 0;

System.out.print(e + (++count % 20 == 0 ? "\n" : " ")); });

No. count will always be 0.

[▼](javascript:collapse('expandText24',%20'collapseText24');)30.8.3

Show the output of the following code?

String s = "ABC";

Stream.of(s.toCharArray()).forEach(ch ->

System.out.println(ch));

ABC

[▼](javascript:collapse('expandText25',%20'collapseText25');)30.8.4

**Show the output of the following code? (The toCharacterArray method is presented in Listing 30.9)**

String s = "ABC";

Stream.of(toCharacterArray(s.toCharArray())).forEach(ch ->

System.out.println(ch));

A

B

C

[▼](javascript:collapse('expandText26',%20'collapseText26');)30.8.5

**Write the code to obtain a one-dimensional array list of strings from a two-dimensional array matrix of strings.**

String[] list = Stream.of(m).map(e -> Stream.of(e)).

reduce((e1, e2) -> Stream.concat(e1, e2)).get().toArray();