Oracle 1Z0-804 Exam Questions & Answers

Number: 1Z0-804
Passing Score: 800
Time Limit: 120 min
File Version: 32.2

Oracle 1Z0-804 Exam Questions & Answers

Exam Name: Java SE 7 Programmer II Exam

Certkey

QUESTION 1
Given the code fragment:

```
DataFormat df;
```

Which statement defines a new Dateformat object that displays the default date format for the UK Locale?

A. df = DateFormat.getdatDataInstance (DateFormat.DEFAULT, Locale (UK));
B. df = DateFormat.getdatDataInstance (DateFormat.DEFAULT, UK);
C. df = DateFormat.getdatDataInstance (DateFormat.DEFAULT, Locale.UK);
D. df = new DateFormat.getdatDataInstance (DateFormat.DEFAULT, Locale.UK);
E. df = new DateFormat.getdatDataInstance (DateFormat.DEFAULT, Locale (UK));

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The UK locale is constructed withLocale.UK.
Example:
To format a date for a different Locale, specify it in the call to getDateInstance(). DateFormat df = DateFormat.getDateInstance(DateFormat.LONG, Locale.FRANCE);

Note:getDateInstance(int style, Locale aLocale)
Gets the date formatter with the given formatting style for the given locale.

Reference:Class DateFormat

QUESTION 2
Given:

```
public class DoubleThread {

public static void main(String[] args) {

Thread t1 = new Thread() {

public void run() {

System.out.print("Greeting");

}



Thread t2 = new Thread(t1); // Line 9

t2.run();

}

}
```

Which two are true?

A. A runtime exception is thrown on line 9.
B. No output is produced.
C. Greeting is printed once.
D. Greeting is printed twice.

E. No new threads of execution are started within the main method.
F. One new thread of execution is started within the main method.
G. Two new threads of execution are started within the main method.

Correct Answer: CE
Section: (none)
Explanation

Explanation/Reference:
Explanation: Thread t2 is executed. Execution of T2 starts executionen of t1. Greeting is printed during the execution of t1.

QUESTION 3
Given:

```
import java.util.*;

public class AccessTest {

public static void main(String[] args) {

Thread t1 = new Thread(new WorkerThread());

Thread t2 = new Thread(new WorkerThread());

t1.start(); t2.start; // line1

}

}

class WorkPool {
```

```
static ArrayList<Integer> list = new ArrayList<>(); // line2

public static void addItem() { // line3

list.add(1); // Line4

}

}

class WorkerThread implements Runnable {

static Object bar = new Object ();

public void run() { //line5

for (int i=0; i<5000;i++) WorkPool.addItem(); // line6

}

}
```

Which of the four are valid modifications to synchronize access to the valid list between threads t1 and t2?

A. Replace line 1 with:
Synchronized (t2) (t1.start();) synchronized(t1) (t2.start();)
B. Replace Line 2 with:

static CopyWriteArrayList<Integer> list = new CopyWriteArrayList<>();

C. Replace line 3 with:
synchronized public static void addItem () {

D. Replace line 4 with:
synchronized (list) (list.add(1);)

E. Replace line 5 with:
Synchronized public void run () {

F. replace line 6 with:
Synchronized (this) {for (in i = 0, i<5000, i++) WorkPool.addItem(); }

G. Replace line 6 with:
synchronized (bar) {for (int i= 0; i<5000; i++) WorkPool.addItem(); }

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:
Explanation: Away to create synchronized code is with synchronized statements. Unlike synchronized methods, synchronized statements must specify the object that provides the intrinsic lock:
For example:

```
public void addName(String name) {
```

```
synchronized(this) {
lastName = name;
nameCount++;
}
nameList.add(name);
}
```

In this example, the addName method needs to synchronize changes to lastName and nameCount, but also needs to avoid synchronizing invocations of other objects' methods. Without synchronized statements, there would have to be a separate, unsynchronized method for the sole purpose of invoking nameList.add.

Reference: The Java Tutorial,Intrinsic Locks and Synchronization

QUESTION 4
Sam has designed an application. It segregates tasks that are critical and executed frequently from tasks that are non critical and executed less frequently. He has prioritized these tasks based on their criticality and frequency of execution. After close scrutiny, he finds that the tasks designed to be non critical are rarely getting executed.

From what kind of problem is the application suffering?

A. race condition

B. starvation

C. deadlock

D. livelock

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Starvation describes a situation where a thread is unable to gain regular access to shared resources and is unable to make progress. This happens when shared resources are made unavailable for long periods by "greedy" threads. For example, suppose an object provides a synchronized method that

often takes a long time to return. If one thread invokes this method frequently, other threads that also need frequent synchronized access to the same object will often be blocked.

Reference: The Java Tutorial, Starvation and Livelock

QUESTION 5
Give:

```
Class Employee {

public int checkEmail() {/* . . . */}

public void sendEmail (String email) {/* . . . */}

public Boolean validDateEmail(){/* . . . */}

public void printLetter (String letter) {/* . . . */}

}
```

Which is correct?

A. Employee takes advantage of composition.
B. Employee "has-an" Email.
C. Employee "is-a" LetterPrinter.
D. Employee has low cohesion.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The relationship between Employee and e-mail is poorly implemented here.
There is low cohesion.

Note:
Low cohesion is associated with undesirable traits such as being difficult to maintain, difficult to test, difficult to reuse, and even difficult to understand.

Cohesion is decreased if:
The functionalities embedded in a class, accessed through its methods, have little in common. Methods carry out many varied activities, often using coarsely-grained or unrelated sets of data. Disadvantages of low cohesion (or"weak cohesion") are:
Increased difficulty in understanding modules.
Increased difficulty in maintaining a system, because logical changes in the domain affect multiple modules, and because changes in one module require changes in related modules. Increased difficulty in reusing a module because most applications won't need the random set of operations provided by a module.

Reference:Cohesion (computer science)

QUESTION 6
Which two demonstrate the valid usage of the keyword synchronized?

A.
```
interface ThreadSafe {
synchronized void doIt();
}
```
B.
```
abstract class ThreadSafe {
synchronized abstract void doIt();
}
```

C.
```
class ThreadSafe {
synchronized static void soIt () {}
}
```

D.
```
enum ThreadSafe {
ONE, TWO, Three;
Synchronized final void doIt () {}
}
```

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The Java programming language provides two basic synchronization idioms: synchronized methods and synchronized statements.
To make a method synchronized, simply add the synchronized keyword to its declaration.

QUESTION 7
Given the incomplete pseudo-code for a fork/join framework application:

```
submit(Data) {

if(Data.size < SMALL_ENOUGH) {

_________________(Data); // line x

}

else {

List<Data> x = _________________(Data); // line Y

for(Data d: x

______________(d); // line z

}

}
```

And given the missing methods:

process, submit, and splitInHalf

Which three insertions properly complete the pseudo-code?

A. Insert submit at line X.
B. Insert splitInHalf at line X.
C. Insert process at line X.
D. Insert process at line Y.
E. Insert splitInHalf at line Y.
F. Insert process at line Z.
G. Insert submit at line Z.

Correct Answer: CEG
Section: (none)
Explanation

Explanation/Reference:

Explanation: C: If data is small enough then process it. Line X
E: If data is not small enough then split it half. Line Y
G: After the data has been split (line Y) then recursively submit the splitted data (Line z).

QUESTION 8
ITEM Table

* ID, INTEGER: PK

* DESCRIP, VARCHAR(100)

* PRICE, REAL

* QUALITY, INTEGER

And given the code fragment (assuming that the SQL query is valid):

```
try {

String query = "SELECT * FROM Item WHERE ID=110";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query);

while (rs.next ()) {

System.out.println("ID: " + rs.getInt("Id"));

System.out.println("Description: " + rs.getString("Descrip"));

System.out.println("Price: " + rs.getDouble("Price"));

System.out.println("Quantity: " + rs.getInt("Quantity"));

}

} catch (SQLException se) {

System.out.println("Error");

}
```

What is the result of compiling and executing this code?

A. An exception is thrown at runtime
B. Compile fails
C. The code prints Error
D. The code prints information about Item 110

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The connection conn is not defined. The code will not compile.

QUESTION 9
Given:

```
class Deeper {
```

```
public Number getDepth() {

return 10;

}

}
```

Which two classes correctly override the getDepth method?

A. public class deep extends Deeper {
protected integer getDepth(){
return 5;
}
}

B. public class deep extends Deeper {
public double getDepth() {
return"5";
}
}

C. public class deep extends Deeper {
public String getDepth () {
}
}

D. public class deep extends Deeper {
public Long getDepth (int d) {
return 5L;
}
}

E. public class deep extends Deeper {
public short getDepth () {
return 5;
}
}

Correct Answer: AE
Section: (none)
Explanation

Explanation/Reference:
Explanation: Note:The abstract class Number is the superclass of classes Byte, Double, Float, Integer, Long, and Short.

Subclasses of Number must provide methods to convert the represented numeric value to byte, double, float, int, long, and short.

When class C extends B, we say that C is a"subclass"of B, and B is the"superclass"of C. This is called inheritence, because C inherited from B.

QUESTION 10
Given the code fragment:

```
public class App {

public static void main (String [] args){

Path path = Paths.get("C:\\education\\institute\\student\\report.txt");
System.out.println("get.Name(0): %s", path.getName(0));

System.out.println ("subpath(0, 2): %s", path.subpath (0, 2));}

}
```

What is the result?

A. getName (0): C:\
subpath (0, 2): C:\education\report.txt

B. getName(0): C:\
subpth(0, 2): C:\education

C. getName(0): education
subpath (0, 2): education\institute

D. getName(0): education
subpath(0, 2): education\institute\student

E. getName(0): report.txt
subpath(0, 2): insritute\student

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Example:
Path path = Paths.get("C:\\home\\joe\\foo");
getName(0)
-> home

subpath(0,2)

Reference: The Java Tutorial, Path Operations

QUESTION 11
To provide meaningful output for:

System.out.print(new Item ()):

A method with which signature should be added to the Item class?

A. public String asString()
B. public Object asString()
C. public Item asString()
D. public String toString()
E. public object toString()
F. public Item toString()

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Implementing toString method in java is done by overriding the Object's toString method. The java toString() method is used when we need a string representation of an object. It is defined in Object class. This method can be overridden to customize the String representation of the Object.

Note:
Below is an example shown of Overriding the default Object toString() method. The toString() method must be descriptive and should generally cover all the contents of the object.

```
class PointCoordinates {

private int x, y;
public PointCoordinates(int x, int y) {
this.x = x;
this.y = y;
}
public int getX() {
return x;
}
public int getY() {
return y;
}
// Custom toString() Method.
public String toString() {
return "X=" + x + " " + "Y=" + y;
}
}
```

QUESTION 12
Given the code fragment:

```
public class DisplaValues {

public void printNums (int [] nums){
for (int number: nums) {

System.err.println(number);

}

}

}
```

Assume the method printNums is passed a valid array containing data. Why is this method not producing output on the console?

A. There is a compilation error.
B. There is a runtime exception.
C. The variable number is not initialized.
D. Standard error is mapped to another destination.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The codecompilesfine.
The code runs fine.
The err stream can be redirected.

Note:
System.out.println -> Sends the output to a standard output stream. Generally monitor.

System.err.println -> Sends the output to a standard error stream. Generally monitor. err is the "standard" error output stream. This stream is already open and ready to accept output data.
Typically this stream corresponds to display output or another output destination specified by the host environment or user. By convention, this output stream is used to display error messages or other information that should come to the immediate attention of a user even if the principal output stream, the value of the variable out, has been redirected to a file or other destination that is typically not continuously monitored.

Reference:java.lang.System

QUESTION 13
Which method would you supply to a class implementing the Callable interface?

A. callable ()
B. executable ()
C. call ()
D. run ()
E. start ()

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: public interface Callable<V>

A task that returns a result and may throw an exception. Implementors define a single method with no arguments called call.

Note:
Interface Callable<V>

Type Parameters:
V - the result type of method call

The Callable interface is similar to Runnable, in that both are designed for classes whose instances are potentially executed by another thread. A Runnable, however, does not return a result and cannot throw a checked exception.

The Executors class contains utility methods to convert from other common forms to Callable classes.

Reference:java.util.concurrent

QUESTION 14
Given the existing destination file, a source file only 1000 bytes long, and the code fragment:

```
public void process (String source, String destination) {

try (InputStream fis = new FileInputStream(source);

OutputStream fos = new FileOutputStream(destination)

) {
byte [] buff = new byte[2014];

int i;

while ((i = fis.read(buff)) != -1) {

fos.write(buff,0,i); // line ***

}

} catch (IOException e) {

System.out.println(e.getClass());

}

}
```

What is the result?

A. Overrides the content of the destination file with the source file content
B. Appends the content of the source file to the destination file after a new line
C. Appends the content of the source file to the destination file without a break in the flow
D. Throws a runtime exception at line***

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The whole of theFileInputStreamwill be read (see ** below). The content of theFileInputStreamwill overwrite the destination file (see *** below).

*A FileInputStream obtains input bytes from a file in a file system. What files are available depends on the host environment.

FileInputStream is meant for reading streams of raw bytes such as image data. For reading streams of characters, consider using FileReader.

**FileInputStream.read(byte[] b)
Reads up to b.length bytes of data from this input stream into an array of bytes.

Parameters:
b - the buffer into which the data is read.
Returns:
the total number of bytes read into the buffer, or -1 if there is no more data because the end of the file has been reached.

***FileOutputStream
You can construct a FileOutputStream object by passing a string containing a path name or a File object. You can also specify whether you want to append the output to an existing file.

public FileOutputStream (String path)
public FileOutputStream (String path, boolean append)
public FileOutputStream (File file)
public FileOutputStream (File file, boolean append)

With the first and third constructors, if a file by the specified name already exists, the file will be overwritten. To append to an existing file, pass true to the second or fourth constructor.

Reference:Class FileInputStream

Reference:Class FileOutputStream

QUESTION 15
Which two codes correctly represent a standard language locale code?

A. ES
B. FR
C. U8
D. Es
E. fr
F. u8

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation: Language codes are defined by ISO 639, an international standard that assigns two- and three-letter codes to most languages of the world. Locale uses the two-letter codes to identify the target language.
ES, Es: Spanish

Reference: IS0 639

QUESTION 16
Which code fragment demonstrates the proper way to handle JDBC resources?

A.
```
Try {
ResultSet rs = stmt.executableQuery (query);
statement stmt = con.createStatement();
while (rs.next()) (/* . . . */)
} catch (SQLException e) {}
```

B.
```
Try {statement stmt = con.createStatement();
ResultSet rs = stmt.executableQuery (query);
while (rs.next()) (/* . . . */)
} catch (SQLException e) {}
```

C.
```
Try {
statement stmt = con.createStatement();
ResultSet rs = stmt.executableQuery (query);
while (rs.next()) (/* . . . */)
} finally {
rs.close();
stmt.close();
}
```

D.
```
Try {ResultSet rs = stmt.executableQuery (query);
statement stmt = con.createStatement();
while (rs.next()) (/* . . . */)
} finally {
rs.close();
stmt.close();
}
```

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: We should use the finally statement to gracefully close the connection.

QUESTION 17
Given:

```
import java.io.IOException;

import java.io.file.Path;

import java.io.file.Paths;

public class Path12 {

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

Path path = Paths.get("\\sales\\quarter\\..\\qtrlreport.txt");

path.relativize(Paths.get("\\sales\\annualreport.txt"));

if(path.endsWith("annualreport.txt")) {
```

```
System.out.println(true);

} else {

System.out.println(false);

}

System.out.println(path);

}

}
```

What is the result?

A. false
\sales\quarter\ . . \qtrlreport.txt

B. false
\quarter\ . . \qtrlreport.txt

C. true
. . \ . . \ . . \ annualreport.txt

D. true
\ . . \ . . \annualreport.txt

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The relativize method that can be used to construct a relative path between two paths.

relativize
Path relativize(Path other)
Constructs a relative path between this path and a given path.
Parameters:
other - the path to relativize against this path
Returns:
the resulting relative path, or an empty path if both paths are equal

Note:
Relativization is the inverse of resolution. This method attempts to construct a relative path that when resolved against this path, yields a path that locates the same file as the given path. For example, on UNIX, if this path is "/a/b" and the given path is "/a/b/c/d" then the resulting relative path would be "c/d". Where this path and the given path do not have a root component, then a relative path can be constructed. A relative path cannot be constructed if only one of the paths have a root component. Where both paths have a root component then it is implementation dependent if a relative path can be constructed. If this path and the given path are equal then an empty path is returned.

For any two normalized paths p and q, where q does not have a root component,

p.relativize(p.resolve(q)).equals(q)
When symbolic links are supported, then whether the resulting path, when resolved against this path, yields a path that can be used to locate the same file as other is implementation dependent. For example, if this path is "/a/b" and the given path is "/a/x" then the resulting relative path may be "../x". If "b" is a symbolic link then is implementation dependent if "a/b/../x" would locate the same file as "/a/x".

QUESTION 18
Given the fragment:

```
public class CustomerApplication {

public static void main (String args[]) {

CustomerDAO custDao= new CustomerDAOMemoryImpl(); // Line 3

// ... other methods

}

}
```

Which two valid alternatives to line 3 would decouple this application from a specific implementation of CustomerDAO?

A. CustomerDAO custDao = CustomerDAO();
B. CustomerDAO custDao = (CustomerDAO) new Object ();
C. CustomerDAO custDao = CustomerDAO.getInstance();
D. CustomerDAO custDao = (CustomerDAO) new CustomerDAOmemoryImp1();
E. CustomerDAO custDao = customerDAOFactory.getInstance();

Correct Answer: BE
Section: (none)
Explanation

Explanation/Reference:

Explanation:
Note: In software development, the term"decoupling"is used to identify the separation of software blocks that shouldn't depend on each other. Some building blocks are generic and shouldn't know details of others. Special design techniques allow software designers to have as few dependencies as possible. This typically reduces the risk of malfunction in one part of a system when the other part changed. It also forces the developer to focus on one thing at a time.
Decoupling lowers or minimizes Coupling.

QUESTION 19
Given this error message when running your application:

```
Exception in thread "main" java.util.MissingResourceException: Can't find bundle for base name MessageBundle, locale
```

And given that the MessageBundle.properties file has been created, exists on your disk, and is properly formatted.

What is the cause of the error message?

A. The file is not in the environment path.
B. The file is not in the classpath.
C. The file is not in the javapath.
D. You cannot use a file to store a ResourceBundle.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

Explanation: ResourceBundle.getBundle is using a resource name; it isn't called ResourceBundle for nothing.You can create a custom ClassLoader and use that to load the data.

QUESTION 20
Given a language code of fr and a country code of FR, which file name represents a resource bundle file name that is not the default?

A. MessageBundle_fr_FR.properties
B. MessageBundle_fr_FR.profile
C. MessageBundle_fr_FR.xinl
D. MessageBundle__fr__FR.Java
E. MessageBundle__fr__FR.Locale

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The default file is MessageBundle.properties. The non-default file name is MessageBundle_fr_FR.properties

Note 0:.properties is a file extension for files mainly used in Java related technologies to store the configurable parameters of an application. They can also be used for storing strings for Internationalization and localization; these are known as Property Resource Bundles. Each parameter is stored as a pair of strings, one storing the name of the parameter (called the key), and the other storing the value.

Note 1:You can obtain an instance of ResourceBundle by calling its static getBundle method.

public static ResourceBundle getBundle(java.lang.String baseName) public static ResourceBundle getBundle(java.lang.String baseName, Locale locale) For example:

ResourceBundle rb = ResourceBundle.getBundle("MyResources", Locale.US); This will load the ResourceBundle object with the values in the corresponding properties file.

1.If a suitable properties file is not found, the ResourceBundle object will use the default properties file, which will be the one whose name equals the base name and has the properties extension. In this case, the default file would be MyResources.properties. 2.If this file is not found, a java.util.MissingResourceException will be thrown.

Note2:java.util.ResourceBundle class enables you to choose and read the properties file specific to the user's locale and look up the values.

A ResourceBundle object has a base name. In order for a ResourceBundle object to pick up a properties file, the filename must be composed of the ResourceBundle base name, followed by an underscore, followed by the language code, and optionally followed by another underscore and the country code.

The format for the properties file name is as follows:

basename_languageCode_countryCode
For example, suppose the base name is MyResources and you define the following three locales:

US-en
DE-de

CN-zh
Then you would have these three properties files:

MyResources_en_US.properties
MyResources_de_DE.properties

MyResources_zh_CN.properties

Reference:Reading Properties Files using ResourceBundle

QUESTION 21
Given these facts about Java types in an application:

- Type x is a template for other types in the application.

- Type x implements dostuff ().

- Type x declares, but does NOT implement doit().

- Type y declares doOther() .

Which three are true?

A. Type y must be an interface.
B. Type x must be an abstract class.
C. Type y must be an abstract class.
D. Type x could implement or extend from Type y.
E. Type x could be an abstract class or an interface.
F. Type y could be an abstract class or an interface.

Correct Answer: BDF
Section: (none)
Explanation

Explanation/Reference:
Explanation: Unlike interfaces, abstract classes can contain fields that are not static and final, and they can contain implemented methods. Such abstract classes are similar to interfaces, except that they provide a partial implementation, leaving it to subclasses to complete the implementation. If an abstract class contains only abstract method declarations, it should be declared as an interface instead.

Note:
An interface in the Java programming language is an abstract type that is used to specify an interface (in the generic sense of the term) that classes must implement. Interfaces are declared using the interface keyword, and may only contain method signature and constant declarations (variable declarations that are declared to be both static and final). An interface may

never contain method definitions.

Note 2: an abstract class is a class that is declared abstract--it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed. An abstract method is a method that is declared without an implementation (without braces, and followed by a semicolon)

QUESTION 22
Given:

class A {

int a = 5;

String doA() { return "a1"; }

protected static String doA2 () { return "a2"; }

}

class B extends A {

int a = 7;

```
String doA() { return "b1"; }

public static String doA2() { return "b2"; }

void go() {

A myA = new B();

System.out.print(myA.doA() + myA.doA2() + myA.a);

}

public static void main (String[] args) { new B().go(); }

}
```

Which three values will appear in the output?

A. 5
B. 7
C. a1
D. a2
E. b1
F. b2

Correct Answer: BEF
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 23
Which represents part of a DAO design pattern?

```
A. interface EmployeeDAO {
   int getID();
   Employee findByID (intid);
   void update();
   void delete();
   }
B. class EmployeeDAO {
   int getID() { return 0;}
   Employee findByID (int id) { return null;}
   void update () {}
   void delete () {}
   }
C. class EmployeeDAO {
   void create (Employee e) {}
   void update (Employee e) {}
   void delete (int id) {}
   Employee findByID (int id) {return id}
   }
D. interface EmployeeDAO {
   void create (Employee e);
   void update (Employee e);
   void delete (int id);
   Employee findByID (int id);
   }
E. interface EmployeeDAO {
```

```
void create (Connection c, Employee e);
void update (Connection c, Employee e);
void delete (Connection c, int id);
Employee findByID (Connection c, int id);
}
```

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 24
Assuming the port statements are correct, which two code fragments create a one-byte file?

A. OutputStream fos = new FileOutputStream(new File("/tmp/data.bin")); OutputStream bos = new BufferedOutputStream(fos);
DataOutputStream dos = new DataOutputStream(bos);
dos.writeByte(0);
dos.close();

B. OutputStream fos = new FileOutputStream ("/tmp/data.bin"); dataOutputStream dos = new DataOutputStream(fos);
dos.writeByte(0);
dos.close();

C. OutputStream fos = new FileOutputStream (new File ("/tmp/data.bin")); dataOutputStream dos = new DataOutputStream(os);
dos.writeByte(0);
dos.close();

D. OutputStream fos = new FileOutputStream ("/tmp/data.bin"); fos.writeByte(0);
fos.close();

Correct Answer: BC
Section: (none)
Explanation

Explanation/Reference:
Explanation: B:Create DataOutputStream from FileOutputStream public static void main(String[] args) throws Exception { FileOutputStream fos = new FileOutputS tream("C:/demo.txt"); DataOutputStream dos = new DataOutputStream(fos);

Note:
The FileOutputStream class is a subclass of OutputStream. You can construct a FileOutputStream object by passing a string containing a path name or a File object.
You can also specify whether you want to append the output to an existing file.
public FileOutputStream (String path)
public FileOutputStream (String path, boolean append)
public FileOutputStream (File file)
public FileOutputStream (File file, boolean append)

With the first and third constructors, if a file by the specified name already exists, the file will be overwritten. To append to an existing file, pass true to the second or fourth constructor.

Note 2:public class DataOutputStream
extends FilterOutputStream
implements DataOutput
A data output stream lets an application write primitive Java data types to an output stream in a portable way. An application can then use a data input stream to read the data back in.

Reference:java.io Class DataOutputStream

QUESTION 25

Given:

```
public class StringSplit01 {

public static void main(String[] args) {

String names = "John-.-George-.-Paul-.-Ringo";

String[] results = names.split("-. .");

for(String str:results) {

System.out.println(str);

}

}

}
```

What is the result?

A. John . George . Paul . Ringo

B. John
George
Paul
Ringo

C. John
George
Paul
Ringo

D. An exception is thrown at runtime

E. Compilation fails

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The split() method is used to split a string into an array of substrings, and returns the new array.

QUESTION 26
Given the integer implements comparable:

```
import java.util.*;

public class SortAndSearch2 {

static final Comparator<Integer> IntegerComparator =

new Comparator<Integer>() {

public int compare (Integer n1, Integer n2) {

return n2.compareTo(n1);

}

};

public static void main(String args[]) {
```

```
ArrayList<Integer> list = new ArrayList<>();

list.add (4);

list.add (1);

list.add (3);

list.add (2);

Collections.sort(list, null);

System.out.println(Collections.binarySearch(list, 3));
Collections.sort(list,IntegerComparator);

System.out.println(Collections.binarySearch(list, 3));

}

}
```

What is the result?

A. 2
B. 1
C. 2
D. 2
E. 3

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 27
Given:

```
public class A { //Line 1

private void a() {}; //Line 2

class B { //Line 3

private void b(){{ //Line 4

a();{ //Line 5

}{ //Line 6

}{ //Line 7

public static void main(String[] args) {{ //Line 8
```

A. B b = new A().new B();{ //Line 9
B. b();{ //Line 10
}{ //Line 11
}{ //Line 12
What is the result?

C. Compilation fails at line 9
D. Compilation fails at line 10
E. Compilation fails at line 5
F. Compilation fails at line 3
G. Compilation succeeds

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 28
Given the interface:

Public interface Idgenerator {

int getNextId();

}

Which class implements IdGenerator in a thread-safe manner, so that no threads can get a duplicate id value current access?

A. Public class generator Implements IdGenerator {
Private AtomicInteger id = new AtomicInteger (0);
return id.incrementAndget();
}
}
B. Public class Generator Implements idGenerator {
private int id = 0;
return ++id;
}
}
C. Public class Generator Implements IdGenerator {
private volatile int Id = 0;
return + + Id;
}
D. Public class Generator Implements IdGenerator {
private int id = 0;
public int getNextId() {
synchronized (new Generator()) {
return + + id;
}
}
}
E. Public class Generator Implements IdGenerator {
private int id = 0;
public int getnextId() {
synchronized (id) {
return + + id;
}
}
}

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Code that is safe to call by multiple threads simultanously is called thread safe. If a piece of code is thread safe, then it contains no race conditions. Race condition only occur when multiplethreads update shared resources. Therefore it is important to know what resources Java threads share when executing.

In Java you can mark a method or a block of code as synchronized. Synchronized blocks can be used to avoid race conditions.

QUESTION 29
Given the code fragment:

```
public class Rank {

static CopyOnWriteArraySet<String> arr = new CopyOnWriteArraySet<>();

static void verify() {

String var ="";

Iterator<String> e=arr.iterator();

while (e.hasNext()) {

var = e.next();

if(var.equals("A"))

arr.remove(var);

}

}

public static void main (String[] args) {

ArrayList<String> list1 = new ArrayList<>();

list1.add("A"); list1.add("B");

ArrayList<String> list2 = new ArrayList<>();

list1.add("A"); list1.add("D");

arr.addAll(list1);

arr.addAll(list2);

verify();

for(String var : arr)

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

}

}
```

What is the result?

A. Null B D
B. Null B null D
C. B D

D. D

E. An exception is thrown at runtime

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 30
Which two code blocks correctly initialize a Locale variable?

A. Locale loc1 = "UK";

B. Locale loc2 = Locale.getInstance("ru");

C. Locale loc3 = Locale.getLocaleFactory("RU");

D. Locale loc4 = Locale.UK;

E. Locale loc5 = new Locale("ru", "RU");

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation: D:The Locale class provides a number of convenient constants that you can use to create Locale objects for commonly used locales. For example, the following creates a Locale object for the United States:
Locale.US

E:Create a Locale object using the constructors in this class:

Locale(String language)
Locale(String language, String country)
Locale(String language, String country, String variant)

Reference: java.utilClass Locale

QUESTION 31
Given:

```
class Product {

private int id;

public Product (int id) {

this.id = id;

}

public int hashCode() {

return id + 42;

}

public boolean equals (Object obj) {

return (this == obj) ? true : super.equals(obj);

}
}

public class WareHouse {

public static void main(String[] args) {

Product p1 = new Product(10);

Product p2 = new Product(10);

Product p3 = new Product(20);

System.out.print(p1.equals(p2) + " ");

System.out.print(p1.equals(p3) );

}

}
```

What is the result?

A. false false
B. true false
C. true true
D. Compilation fails
E. An exception is thrown at runtime

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Hint for line: public class Warehouse {
class WareHouse is public, should be declared in a file named WareHouse.java.
Compilation result:
Error: Main method not found in class product.Product, please define the main method as:
public static void main(String[] args)
Java Result: 1

QUESTION 32
Given:

```
class Erupt implements Runnable {

public void run() {

System.out.print(Thread.currentThread().getName());
```

```
}

}

public class Yellowstone {

static Erupt e = Erupt();

Yellowstone() { new Thread(e, "const").start(); } // line A
```

```
public static void main(String[] args) {

new Yellowstone();

new Faithful().go();

}

static class Faithful {

void go() { new Thread(e, "inner").start(); } // line B

}

}
```

What is the result?

A. Both const and inner will be in the output.
B. Only const will be in the output.
C. Compilation fails due to an error on line A.
D. Compilation fails due to an error on line B.
E. Anexceptionis thrown at runtime.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code compiles fine.

Note:The Runnable interface should be implemented by any class whose instances are intended to be executed by a thread. The class must define a method of no arguments called run.

This interface is designed to provide a common protocol for objects that wish to execute code while they are active. For example, Runnable is implemented by class Thread. Being active simply means that a thread has been started and has not yet been stopped.

In addition, Runnable provides the means for a class to be active while not subclassing Thread. A class that implements Runnable can run without subclassing Thread by instantiating a Thread instance and passing itself in as the target. In most cases, the Runnable interface should be used if you are only planning to override the run() method and no other Thread methods. This is important because classes should not be subclassed unless the programmer intends on modifying or enhancing the fundamental behavior of the class.

Note 2:start()
Causes this thread to begin execution; the Java Virtual Machine calls the run method of this thread.

Reference:java.lang

Interface Runnable

QUESTION 33
Given the code fragment:

```
try {

conn.setAutoCommit(false);
stmt.executeUpdate("insert into employees values(1,'Sam')");

Savepoint save1 = conn.setSavepoint("point1");

stmt.executeUpdate("insert into employees values(2,'Jane')");

conn.rollback();

stmt.executeUpdate("insert into employees values(3,'John')");

conn.setAutoCommit(true);

stmt.executeUpdate("insert into employees values(4,'Jack')");

ResultSet rs = stmt.executeQuery("select * from employees");

while (rs.next()) {

System.out.println(rs.getString(1) + " " + rs.getString(2));

}

} catch(Exception e) {
System.out.print(e.getMessage());

}
```

What is the result of the employees table has no records before the code executed?

A. 1 Sam

B. 4 Jack

C. 3 John
4 Jack

D. 1 Sam
3 John
4 Jack

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Autocommit is set to false. The two following statements will be within the same transaction. stmt.executeUpdate("insert into employees values(1,'Sam')"); stmt.executeUpdate("insert into employees values(2,'Jane')"); These two statements are rolled back through (the savepoint is ignored the savepoint must be specified in the rollback if you want to rollback to the savepoint):
conn.rollback();
The next two insert statements are executed fine. Their result will be in the output.

QUESTION 34
Which statement declares a generic class?

A. public class Example < T >{ }

B. public class <Example>{ }

C. public class Example <> { }

D. public class Example (Generic){ }

E. public class Example (G) { }

F. public class Example { }

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 35
Given:

```
public abstract class Account {

abstract void deposit (double amt);

public abstract Boolean withdraw (double amt);

}

public class CheckingAccount extends Account {

}
```

What two changes, made independently, will enable the code to compile?

A. Change the signature of Account to: public class Account.
B. Change the signature of CheckingAccount to: public abstract CheckingAccount
C. Implement private methods for deposit and withdraw in CheckingAccount.
D. Implement public methods for deposit and withdraw in CheckingAccount.
E. Change Signature of checkingAccount to: CheckingAccount implements Account.
F. Make Account an interface.

Correct Answer: BF
Section: (none)
Explanation

Explanation/Reference:
Explanation: B:Unlike interfaces, abstract classes can contain fields that are not static and final, and they can contain implemented methods. Such abstract classes are similar to interfaces, except that they provide a partial implementation, leaving it to subclasses to complete the implementation.

F:If an abstract class contains only abstract method declarations, it should be declared as an interface instead.

QUESTION 36
Given:

```
Deque <String> myDeque = new ArrayDeque<String>();

myDeque.push("one");
myDeque.push("two");

myDeque.push("three");

System.out.println(myDeque.pop());
```

What is the result?

A. Three

B. One
C. Compilation fails.
D. The program runs, but prints no output.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 37
Given the code fragment:

SimpleDataFormat sdf;

Which code fragment displays the three-character month abbreviation?

A. SimpleDateFormatsdf = new SimpleDateFormat ("mm", Locale.UK); System.out.println ("Result:" + sdf.format(new Date()));
B. SimpleDateFormatsdf = new SimpleDateFormat ("MM", Locale.UK); System.out.println ("Result:" + sdf.format(new Date()));
C. SimpleDateFormatsdf = new SimpleDateFormat ("MMM", Locale.UK); System.out.println ("Result:" + sdf.format(new Date()));
D. SimpleDateFormatsdf = new SimpleDateFormat ("MMMM", Locale.UK); System.out.println ("Result:" + sdf.format(new Date()));

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Output example: JUN

QUESTION 38
Given the code fragment:

```
1. path file = path.get (args.[0])

2. try {

3. } / / statements here

4. catch (IOException) i ) {

5. }
```

And a DOS-based file system:

Which option, containing statement(s), inserted at line 3, creates the file and sets its attributes to hidden and read-only?

A. DOSFileAttributes attrs = Files.setAttribute(file,"dos:hidden","dos: readonly") Files.createFile(file, attrs)
B. Files.craeteFile(file);
   Files.setAttribute(file,"dos:hidden","dos:readonly");
C. Files.createFile(file,"dos:hidden","dos:readonly");
D. Files.createFile(file);
   Files.setAttribute(file,"dos:hidden", true);
   Files.setAttribute(file,"dos:readonly", true);

Correct Answer: D

Section: (none)
Explanation

Explanation/Reference:
Explanation: You can set a DOS attribute using the setAttribute(Path, String, Object, LinkOption...) method, as follows:

```
Path file = ...;
Files.setAttribute(file, "dos:hidden", true);
```

Note:
setAttribute

```
public static Path setAttribute(Path path,
String attribute,
Object value,
LinkOption... options)
throws IOException
```

Sets the value of a file attribute.

Reference:Interface DosFileAttribute

QUESTION 39
When using the default file system provider with a JVM running on a DOS-based file system, which statement is true?

A. DOS file attributes can be read as a set in a single method call.
B. DOS file attributes can be changed as a set in a single method call.
C. DOS file attributes can be modified for symbolic links and regular files.
D. DOS file attributes can be modified in the same method that creates the file.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: File attributes associated with a file in a file system that supports legacy "DOS" attributes.
Usage Example:

```
Path file = ...
DosFileAttributes attrs = Files.readAttributes(file, DosFileAttributes.class);
```

Note:
The methodreadAttributes() reads a file's attributes as a bulk operation.

QUESTION 40
Given:

```
ConcurrentMap <String, String> PartList = new ConcurrentMap<>();
```

Which fragment puts a key/value pair in partList without the responsibility of overwriting an existing key?

A. partList.out(key,"Blue Shirt");
B. partList.putIfAbsent(key,"Blue Shirt");
C. partList.putIfNotLocked (key,"Blue Shirt");
D. partList.putAtomic(key,"Blue Shirt")
E. if (!partList.containsKey(key)) partList.put (key,"Blue Shirt");

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: putIfAbsent(K key, V value)
If the specified key is not already associated with a value, associate it with the given value.

Reference:java.util.concurrent,Interface ConcurrentMap<K,V>

QUESTION 41
Given the code fragment:

```
String query = "SELECT ID FROM Employee";

try (Statement stmt = conn.createStatement()) {

ResultSet rs = stmt.executeQuery(query);

stmt.executeQuery("SELECT ID FROM Customer"); // Line ***

while (rs.next()) {

// process the results

System.out.println ("Employee ID: " + rs.getInt("ID"));

}

} catch (Exception e) {

System.err.println ("Error");

}
```

Assume that the SQL queries return records. What is the result of compiling and executing this code fragment?

A. The program prints employee IDs
B. The program prints customer IDs
C. The program prints Error
D. Compilation fails on line***

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The program compiles and runs fine. Both executeQuery statements will run. The first executeQuery statement (ResultSet rs = stmt.executeQuery(query);) will set the rs Resultset. It will be usedin the while loop. EmployIDs will be printed.

Note:Executes the given SQL statement, which returns a single ResultSet object.
Parameters:
sql - an SQL statement to be sent to the database, typically a static SQL SELECT statement Returns:

a ResultSet object that contains the data produced by the given query; never null

QUESTION 42
Given the following code fragment:

```
public static void main(String[] args) {

Connection conn = null;

Deque<String> myDeque = new ArrayDeque<>();

myDeque.add("one");

myDeque.add("two");

myDeque.add("three");

System.out.println(myDeque.remove());

}
```

What is the result?

A. Three
B. One
C. Compilation fails
D. The program runs, but prints no outout

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: TheArrayDeque.remove() method retrieves and removes the head of the queue represented by this deque.The head of the queue is item "one".

Reference:java.util

Class ArrayDeque

QUESTION 43

Given the code fragment:

```
1. Thread t1 = new Thread ();

2. t1.start ()

3. t1.join ( );

4. // . . .
```

Which three are true?

A. On line 3, the current thread stops and waits until the t1 thread finishes.
B. On line 3, the t1 thread stops and waits until the current thread finishes.
C. On line 4, the t1 thread is dead.
D. On line 4, the t1 thread is waiting to run.
E. This code cannot throw a checked exception.
F. This code may throw a checked exception.

Correct Answer: ACF
Section: (none)
Explanation

Explanation/Reference:
Explanation: A, C: Thejoin()methods waits for this thread to die.

QUESTION 44
The two methods of code reuse that aggregate the features located in multiple classes are ____________ .

A. Inheritance
B. Copy and Paste
C. Composition
D. Refactoring
E. Virtual Method Invocation

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation: A: Inheritance is a way of reusing code and building bigger more functional objects from a basic object.
The original little object, the parent, is called the super-class. The more functional object that inherits from it is called the sub-class .
C: When your goal is code reuse, composition provides an approach that yields easier-to-change code.

QUESTION 45
Given the code fragment:

```
public static void main(String[] args) {

String source ="d:\\company\\info.txt";

String dest ="d:\\company\\emp\\info.txt";

// insert code fragment here. Line ***

} catch (IOException e) {

System.err.println("Caught IOException"+ e.getMessage());

}

}
```

Which two try statements, when inserted at line ***, enable the code to successfully move the file info.txt to the destination directory, even if a file by the same name already exists in the destination directory?

A. try (FileChannel in = new FileInputStream (source). getChannel(); FileChannel out = new FileOutputStream(dest).getChannel()) { in.transferTo(0, in.size(), out);
B. try (Files.copy(Paths.get(source),Paths.get(dest));
Files.delete (Paths.get(source));
C. try (Files.copy(Paths.get(source), Paths.get(dest),StandardCopyOption.REPLACE_Existing);
Files.delete(Paths.get(source));

D. try (Files.move(Paths.get(source),Paths.get(dest));

E. try(BufferedReader br = Files.newBufferedReader(Paths.get(source), Charset.forName("UTF- 8"));
BufferedWriter bw = Files.newBufferedWriter(Paths.get(dest), Charset.forName("UTF-8")); String record = "";
while ((record = br.readLine()) ! = null) {
bw.write(record);
bw.newLine();
}
Files.delete(Paths.get(source));

Correct Answer: CE
Section: (none)
Explanation

Explanation/Reference:
Explanation: C: Copies and overwrites the destination file (thanks

toStandardCopyOption.REPLACE_Existing).
Deletes the original file.
E: By default the buffered writer replaces the existing file.This is what is needed in this scenario.

QUESTION 46
Which two actions can be used in registering a JDBC 3.0 driver?

A. Add the driver class to the META-INF/services folder of the JAR file.
B. Set the driver class name by using the jdbc.drivers system property.
C. Include the JDBC driver class in a jdbcproperties file.
D. Use the java.lang.class.forName method to load the driver class.
E. Use the DriverManager.getDriver method to load the driver class.

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation: A:f your JDBC Driver is NOT JDBC 4-compliant then we can update the driver using"jar"utility by adding the"META-INF/services/java.sql.Driver"inside it. as following:

D:Dynamic loading of Java classes at runtime provides tremendous flexibility in the development of enterprise systems. It provides for the basis of "application servers", and allows even simpler, lighter-weight systems to accomplish some of the same ends. Within Java, dynamic-loading is typically achieved by calling the forName method on the class java.lang.Class

QUESTION 47
Given:

interface Books {

//insert code here

}

Which fragment, inserted in the Books interface, enables the code to compile?

A. public abstract String type;
public abstract String getType();

B. public staticString type;

public abstract String getType();

C. public String type = "Fiction";
public static String getType();

D. public String type = "Fiction";
public abstract String getType();

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 48
The default file system includes a logFiles directory that contains the following files:

Log-Jan 2009

log_0l_20l0

log_Feb20l0

log_Feb2011

log_10.2012

log-sum-2012

How many files Hoes the matcher in this fragment match?

PathMatcher matcher = FileSystems.getDefault ().getPathMatcher ("glob: *???_*1?");

A. One
B. Two
C. Three
D. Four
E. Five
F. Six

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The pattern to match is*???_*1?
This means at least three characters before the symbol _, folloeded by any amount of characters. The next to last character must be 1. The last character can by any character.
The following file names match this pattern:

log_0l_20l0
log_Feb20l0

log_Feb2011
log_10.2012

QUESTION 49
Given:

```
interface Event {
String type = "Event";
public void details();
}
class Quiz {
static String type = "Quiz";
}
public classPracticeQuiz extends Quiz implements Event {
public void details() {
System.out.print(type);
}
public static void main(String[] args) {
new PracticeQuiz().details();
System.out.print(" " + type);
}
}
```

What is the result?

A. Event Quiz
B. Event Event
C. Quiz Quiz
D. Quiz Event
E. Compilation fails

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 50
Given that myfile.txt contains:

```
First
Second
Third
```

Given the following code fragment:

```
public class ReadFile02 {
public static void main(String[] args) {
```

```
String fileName1 = "myfile.txt";
String fileName2 = "newfile.txt";
try (BufferedReader buffIn =
new BufferedReader(new FileReader(fileName1));
BufferedWriter buffOut =
new BufferedWriter(new FileWriter(fileName2))
) {
String line = ""; int count = 1;
line = buffIn.readLine();
while (line != null) {
buffOut.write(count + ": " + line);
```

```
buffOut.newLine();
count++;
line = buffIn.readLine();
}
} catch (IOException e) {
System.out.println("Exception: " + e.getMessage());
}
}
}
```

What is the result?

A. new file.txt contains:
 1: First
 2: Second
 3: Third

B. new file.txt contains:
 1: First 2: Second 3: Third

C. newfile.txt is empty

D. an exception is thrown at runtime

E. compilation fails

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: For each line in the file myfile.text the line number and the line is written into newfile.txt.

QUESTION 51

Given:

```
public class MarkOutOfBoundsException extends ArrayIndexOutOfBoundsException {

public class Test {

public void verify(int[] arr)
throws ArrayIndexOutOfBoundsException {

for (int i = 1; i <= 3; i++) {

if(arr[i] > 100)

throw new MarkOutOfBoundsException();

System.out.println(arr[i]);

}

}

public static void main(String[] args) {

int[] arr = {105,78,56};

try {

new Test().verify(arr);

} catch (ArrayIndexOutOfBoundsException |

MarkOutOfBoundsException e) {

System.out.print(e.getClass());

}

}

}
```

What is the result?

A. Compilation fails.
B. 78
class java.lang.Array.IndexOutOfBoundException
C. class MarkOutOfBoundException
D. class java.lang.arrayIndexOutOfBoundException

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 52
View the exhibit: (*Missing*)

Given the code fragment:

```
class Finder extends SimpleFileVisitor<Path> {
```

```
private final PathMatcher matcher;
private static int numMatches = 0;
Finder () {
matcher = FileSystems.getDefault().getPathMatcher("glob:*java");
}
void find(Path file) {
Path name = file.getFileName();
if (name != null && matcher.matches(name)) {
numMatches++;
}
}
void report()
{
System.out.println("Matched: " + numMatches);
}
@Override
public FileVisitResult visitFile(Path file, BasicFileAttributes attrs)
find(file);
return CONTINUE;
}
}
public class Visitor {
public static void main(String[] args) throws IOException {
Finder finder = new Finder();
Files.walkFileTree(Paths.get("D:\\Project"), finder);
finder.report();
}
}
```

What is the result?

A. Compilation fails
B. 6
C. 4

D. 1
E. 3
F. Not possible to answer due to missing exhibit.

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Note: TheFileSystems.getDefault() returns the default FileSystem. The default file system creates objects that provide access to the file systems accessible to the Java virtual machine. The working directory of the file system is the current user directory, named by the system property user.dir.

QUESTION 53
Which concept allows generic collections to interoperate with java code that defines collections that use raw types?

A. bytocode manipulation
B. casting
C. autoboxing
D. auto-unboxing
E. type erasure

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Autoboxing, introduced in Java 5, is the automatic conversion the Java compiler makes between the primitive (basic) types and their corresponding object wrapper classes (eg, int and Integer, double and Double, etc). The underlying code that is generated is the same, but autoboxing provides a sugar coating that avoids the tedious and hard-to-read casting typically required by Java Collections, which can not be used with primitive types.

QUESTION 54
Given:

```
public class Task {

String title;

static class Counter {

int counter = 0;

void increment() {counter++}

}

public static void main(String[] args) {

// insert code here

}

}
```

Which statement, inserted at line 8, enables the code to compile?

A. new Task().new Counter().increment();
B. new Task().Counter().increment();
C. new Task.Counter().increment();
D. Task.Counter().increment();
E. Task.Counter.increment();

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 55
For which three objects must a vendor provide implementations in its JDBC driver?

A. Time
B. Date
C. Statement
D. ResultSet
E. Connection
F. SQLException
G. DriverManager

Correct Answer: CDE
Section: (none)
Explanation

Explanation/Reference:
Explanation: All JDBC drivers implement the four important JDBC classes: Driver, Connection, Statement, and ResultSet.

QUESTION 56
Given these facts about Java classes in an application:

- Class X is-a Class SuperX.

- Class SuperX has-a public reference to a Class Z.

- Class Y invokes public methods in Class Util.

- Class X uses public variables in Class Util.

Which three statements are true?

A. Class X has-a Class Z.
B. Class Util has weak encapsulation.
C. Class Y demonstrates high cohesion.
D. Class X is loosely coupled to Class Util.
E. Class SuperX's level of cohesion CANNOT be determined

Correct Answer: BDE
Section: (none)
Explanation

Explanation/Reference:
Explanation: B: Has class Util has both public methods and variables, it is an example of weak encapsulation.
Note:Inheritance is also sometimes said to provide "weak encapsulation," because if you have code that directly uses a subclass, such as Apple, that code can be broken by changes to a superclass, such as Fruit. One of the ways to look at inheritance is that it allows subclass code to reuse superclass code. For example, if Apple doesn't override a method defined in its superclass

Fruit, Apple is in a sense reusing Fruit's implementation of the method. But Apple only "weakly encapsulates" the Fruit code it is reusing, because changes to Fruit's interface can break code that directly uses Apple.

D:
Note:Tight coupling is when a group of classes are highly dependent on one another.

This scenario arises when a class assumes too many responsibilities, or when one concern is spread over many classes rather than having its own class.

Loose coupling is achieved by means of a design that promotes single-responsibility and separation of concerns.

A loosely-coupled class can be consumed and tested independently of other (concrete) classes.

Interfaces are a powerful tool to use for decoupling. Classes can communicate through interfaces rather than other concrete classes, and any class can be on the other end of that communication simply by implementing the interface.
E: Not enough information regardingSuperX'to determine the level of cohesion.

QUESTION 57
Given:

```
final class FinalShow { // Line 1

final String location; // Line 2

FinalShow(final String loc) { // Line 3

location = loc; // Line 4

} // Line 5

FinalShow(String loc, String title) { // Line 6

location = loc; // Line 7

loc = "unknown"; // Line 8

} // Line 9

} // Line 10
```

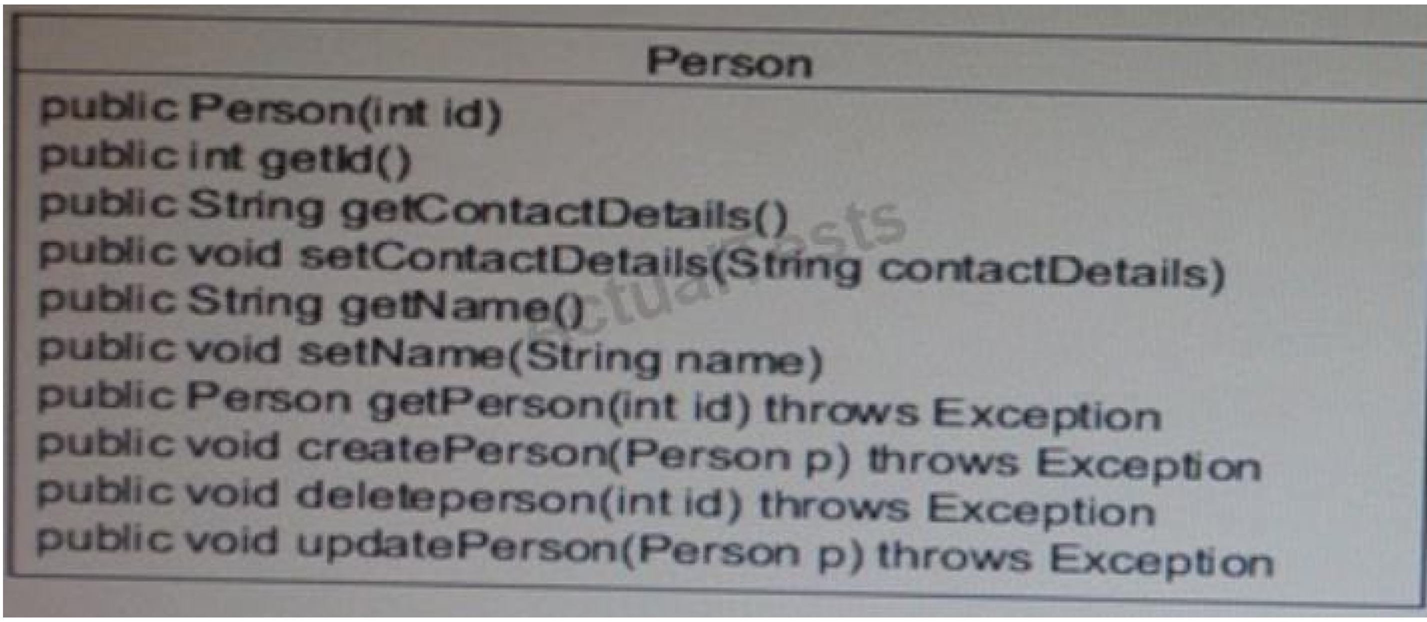
What is the result?

A. Compilation succeeds.
B. Compilation fails due to an error on line 1.
C. Compilation fails due to an error on line 2.
D. Compilation fails due to an error on line 3.
E. Compilation fails due to an error on line 4.
F. Compilation fails due to an error on line 8.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 58
Given:



```
Person
public Person(int id)
public int getId()
public String getContactDetails()
public void setContactDetails(String contactDetails)
public String getName()
public void setName(String name)
public Person getPerson(int id) throws Exception
public void createPerson(Person p) throws Exception
public void deleteperson(int id) throws Exception
public void updatePerson(Person p) throws Exception
```

Which group of method is moved to a new class when implementing the DAO pattern?

A. public in getId ()
public String getContractDetails ()
public Void setContractDetails(String contactDetails)
public String getName ()
public void setName (String name)

B. public int getId ()
public String getContractDetails()
public String getName()
public Person getPerson(int id) throws Exception

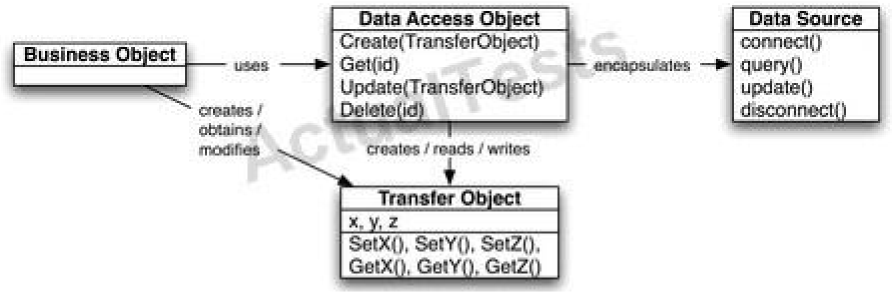
C. public void setContractDetails(String contractDetails) public void setName(String name)

D. public Person getPerson(int id) throws Exception
public void createPerson(Person p) throws Exception
public void deletePerson(int id) throws Exception
public void updatePerson(Person p) throws Exception

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The methods related directly to the entity Person is moved to a new class.

Note:DAO Design Pattern
*Abstracts and encapsulates all access to a data source *Manages the connection to the data source to obtain and store data *Makes the code independent of the data sources and data vendors (e.g. plain-text, xml, LDAP, MySQL, Oracle, DB2)



Example (here Customer is the main entity):

```
public class Customer {
private final String id;
private String contactName;
private String phone;
public void setId(String id) { this.id = id; }
public String getId() { return this.id; }
public void setContactName(String cn) { this.contactName = cn;} public String getContactName() { return
this.contactName; } public void setPhone(String phone) { this.phone = phone; } public String getPhone()
{ return this.phone; }
}

public interface CustomerDAO {
public void addCustomer(Customer c) throws DataAccessException; public Customer getCustomer(String
id) throws DataAccessException; public List getCustomers() throws DataAccessException; public void
removeCustomer(String id) throws DataAccessException; public void modifyCustomer(Customer c) throws
DataAccessException; }
```

QUESTION 59
Given:

```
public class Test {

Integer x; // line 2

public static void main(String[] args) {

new Test().go(5);

}

void go(Integer i) { // line 6

System.out.print(x + ++i); // line 7

}

}
```

What is the result?

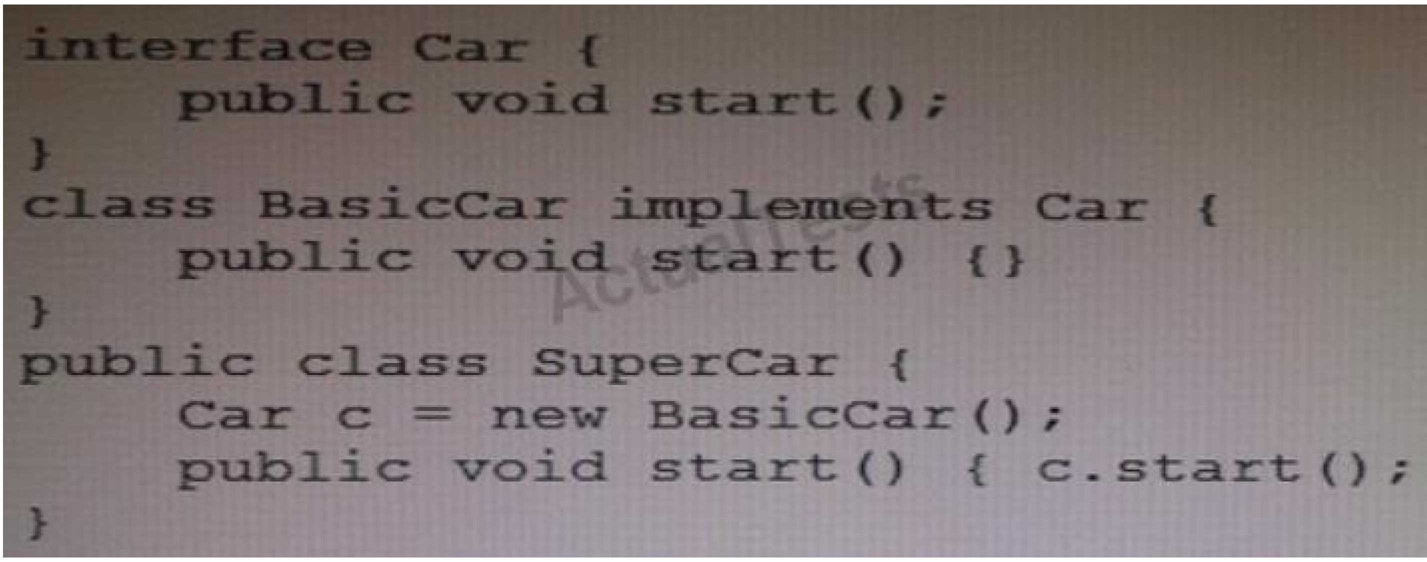
A. 5
B. 6
C. An exception is thrown at runtime
D. Compilation fails due to an error on line 6
E. Compilation fails due to an error on line 7

Correct Answer: C

Section: (none)
Explanation

Explanation/Reference:
Explanation: The code compile finem but ajava.lang.NullPointerExceptionis thrown at runtime. X has no value. The code would run if line 2 was changed to:
Integer x = 3;

QUESTION 60
Given:



```
interface Car {
    public void start();
}
class BasicCar implements Car {
    public void start() {}
}
public class SuperCar {
    Car c = new BasicCar();
    public void start() { c.start();
}
```

Which three are true?

A. BasicCar uses composition.
B. SuperCar uses composition.
C. BasicCar is-a Car.
D. SuperCar is-a Car.
E. SuperCar takes advantage of polymorphism
F. BasicCar has-a Car

Correct Answer: BCE
Section: (none)
Explanation

Explanation/Reference:
Explanation: B: The relationship modeled by composition is often referred to as the "has-a" relationship. Here SuperCar has-a Car.
C:The relationship modeled by inheritance is often referred to as the "is-a" relationship. Modeling an is-a relationship is called inheritance because the subclass inherits the interface and, by default, the implementation of the superclass. Inheritance of interface guarantees that a subclass can accept all the same messages as its superclass. A subclass object can, in fact, be used anywhere a superclass object is called for.
E:The polymorphic method call allows one type to express its distinction from another, similar type, as long as they're both derived from the same base type. This distinction is expressed through differences in behavior of the methods that you can call through the base class.

QUESTION 61
Given:

import java.util.Map;

import java.util.Set;

```
import java.util.TreeMap;
public class MapClass {

public static void main(String[] args) {

Map <String, String> partList = new TreeMap<>();

partList.put("P002", "Large Widget");

partList.put("P001", "Widget");

partList.put("P002", "X-Large Widget");

Set<String> keys = partList.keySet();

for (String key:keys) {

System.out.println(key + " " + partList.get(key));

}

}

}
```

What is the result?

A. p001 Widget
   p002 X-Large Widget
B. p002 Large Widget
   p001 Widget
C. p002 X-large Widget
   p001 Widget
D. p001 Widget
   p002 Large Widget
E. compilation fails

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: Compiles fine. Output is:
P001 Widget
P002 X-Large Widget

Line:partList.put("P002", "X-Large Widget");
overwrites line:partList.put("P002", "Large Widget");

QUESTION 62

A valid reason to declare a class as abstract is to:

A. define methods within a parent class, which may not be overridden in a child class
B. define common method signatures in a class, while forcing child classes to contain unique method implementations
C. prevent instance variables from being accessed
D. prevent a class from being extended

E. define a class that prevents variable state from being stored when object Instances are serialized

F. define a class with methods that cannot be concurrently called by multiple threads

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: Note:An abstract method in Java is something like a pure virtual function in C++ (i.e., a virtual function that is declared = 0). In C++, a class that contains a pure virtual function is called an abstract class and cannot be instantiated. The same is true of Java classes that contain abstract methods.

Any class with an abstract method is automatically abstract itself and must be declared as such.

An abstract class cannot be instantiated.

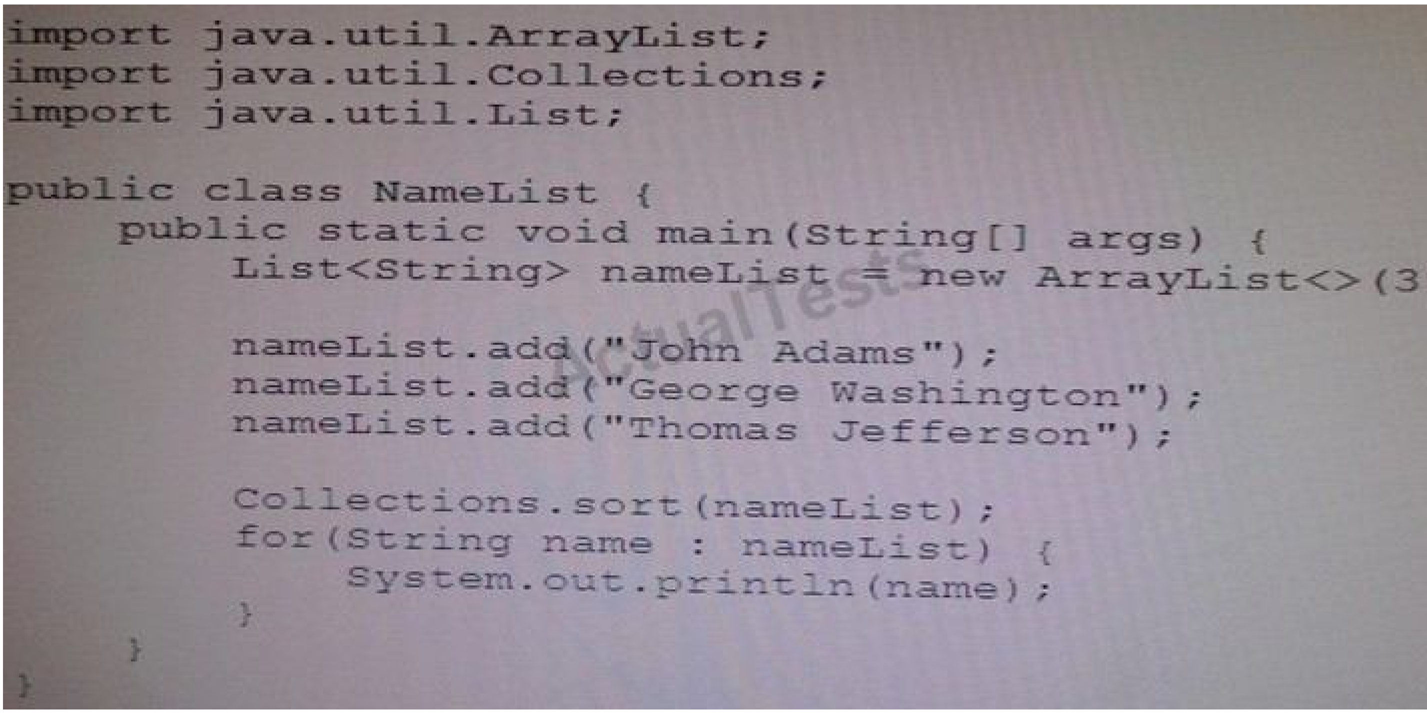
A subclass of an abstract class can be instantiated only if it overrides each of the abstract methods of its superclass and provides an implementation (i.e., a method body) for all of them. Such a class is often called a concrete subclass, to emphasize the fact that it is not abstract.

If a subclass of an abstract class does not implement all the abstract methods it inherits, that subclass is itself abstract.

static, private, and final methods cannot be abstract, since these types of methods cannot be overridden by a subclass. Similarly, a final class cannot contain any abstract methods.

A class can be declared abstract even if it does not actually have any abstract methods. Declaring such a class abstract indicates that the implementation is somehow incomplete and is meant to serve as a superclass for one or more subclasses that will complete the implementation. Such a class cannot be instantiated.

QUESTION 63
Given:



```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class NameList {
    public static void main(String[] args) {
        List<String> nameList = new ArrayList<>(3

        nameList.add("John Adams");
        nameList.add("George Washington");
        nameList.add("Thomas Jefferson");

        Collections.sort(nameList);
        for(String name : nameList) {
            System.out.println(name);
        }
    }
}
```

What is the result?

A. John Adams
George Washington

Thomas Jefferson

B. George Washington
John Adams
Thomas Jefferson

C. Thomas Jefferson
John Adams
George Washington

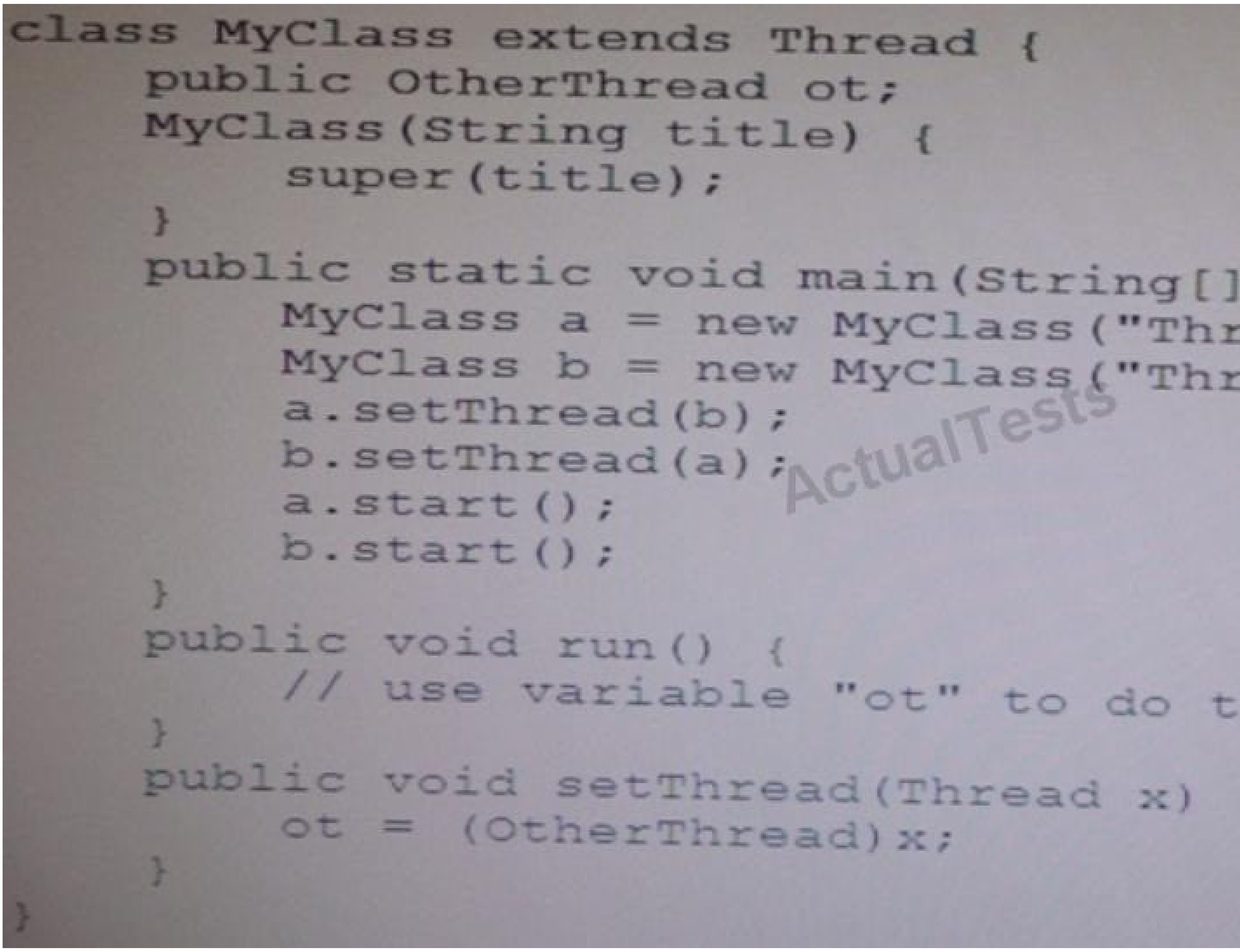
D. An exception is thrown at runtime

E. Compilation fails

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The program compiles and runs fine.
At runtime the NameList is built and then sorted.

QUESTION 64
Given the fragment:



```
class MyClass extends Thread {
    public OtherThread ot;
    MyClass(String title) {
        super(title);
    }
    public static void main(String[]
        MyClass a = new MyClass("Thr
        MyClass b = new MyClass("Thr
        a.setThread(b);
        b.setThread(a);
        a.start();
        b.start();
    }
    public void run() {
        // use variable "ot" to do t
    }
    public void setThread(Thread x)
        ot = (OtherThread)x;
    }
}
```

If thread a and thread b are running, but not completing, which two could be occurring?

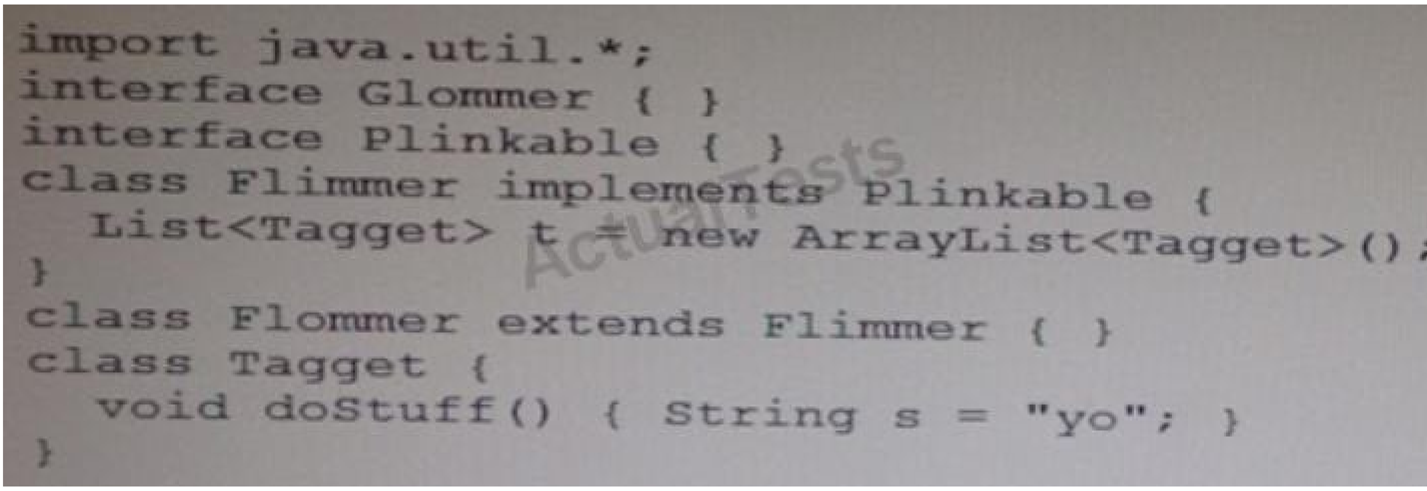
A. livelock

B. deadlock
C. starvation
D. loose coupling
E. cohesion

Correct Answer: AB
Section: (none)
Explanation

Explanation/Reference:
Explanation: A:A thread often acts in response to the action of another thread. If the other thread's action is also a response to the action of another thread, then livelock may result. A thread often acts in response to the action of another thread. If the other thread's action is also a response to the action of another thread, then livelock may result.
B:Deadlock describes a situation where two or more threads are blocked forever, waiting for each other.

QUESTION 65
Given:



```
import java.util.*;
interface Glommer { }
interface Plinkable { }
class Flimmer implements Plinkable {
  List<Tagget> t = new ArrayList<Tagget>();
}
class Flommer extends Flimmer { }
class Tagget {
  void doStuff() { String s = "yo"; }
}
```

Which three statements concerning the OO concepts "is-a" and "has-a" are true?

A. Flimmer is-a Plinkable
B. Flommer has-a Tagget
C. Flommer is-a Glommer
D. Tagget has-a String
E. Flommer is-a Plinkable
F. Flimmer is-a Flommer
G. Tagget is-a Plinkable

Correct Answer: ADF
Section: (none)
Explanation

Explanation/Reference:
Explanation: A: Flimmer implements Plinkable.
Flimmer is-a plinkable.
D:The relationship modeled by composition is often referred to as the "has-a" relationship.
HereTaggethas-aString.
F: Flommer extends Flimmer
So there is an "is-a relationship between Flommer and Flimmer .

Note: Thehas-a relationship has anencapsulation feature (like private or protected modifier used before each member field or method).

QUESTION 66

Given:

```
import java.util.*;

public class CompareTest {

public static void main(String[] args) {

TreeSet<String> set1 = new TreeSet<String>(

new Comparator<String>() {

public boolean compare(String s1, String s2) {

return s1.length() > s2.length();

}

});

set1.add("peach");
set1.add("orange");

set1.add("apple");

for (String n: set1) {

System.out.println(n);

}

}

}
```

What is the result?

A. peach
orange
apple

B. peach
orange

C. apple
orange

D. The program does not compile.

E. The program generates an exception at runtime.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:

Explanation: The compiler has a problem with the line:

```
public boolean compare(String s1, String s2) {
return s1.length() > s2.length();
```

```
error: <anonymous comparetest.CompareTest$1> is not abstract and does not override abstract method
compare(String,String) in Comparator
new Comparator<String>() {
```

```
Error: compare(String,String) in <anonymous comparetest.CompareTest$1> cannot implement compare
```

```
(T,T) in Comparator
public boolean compare(String s1, String s2) {
return type boolean is not compatible with int
where T is a type-variable:
T extends Object declared in interface Comparator
```

QUESTION 67

Given the code fragment:

```
public class ReadFile01 {

public static void main(String[] args) {

String fileName = "myfile.txt";

try (BufferedReader buffIn = // Line 4

new BufferedReader(new FileReader(fileName))) {

String line = ""; int count = 1;

line = buffIn.readLine(); // Line 7

do {

line = buffIn.readLine();

System.out.println(count + ": " + line);

} while (line != null);

} catch (IOException | FileNotFoundException e) {

System.out.println("Exception: " + e.getMessage());

}

}

}
```

What is the result, if the file myfile.txt does not exist?

A. A runtime exception is thrown at line 4
B. A runtime exception is thrown at line 7
C. Creates a new file and prints no output
D. Compilation fails

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: There will be a FileNotFoundException at line 4.

QUESTION 68

Which two forms of abstraction can a programmer use in Java?

A. enums
B. interfaces
C. primitives
D. abstract classes
E. concrete classes
F. primitive wrappers

Correct Answer: BD
Section: (none)
Explanation

Explanation/Reference:
Explanation: *When To Use Interfaces

An interface allows somebody to start from scratch to implement your interface or implement your interface in some other code whose original or primary purpose was quite different from your interface. To them, your interface is only incidental, something that have to add on to the their code to be able to use your package. The disadvantage is every method in the interface must be public. You might not want to expose everything.

*When To Use Abstract classes
An abstract class, in contrast, provides more structure. It usually defines some default implementations and provides some tools useful for a full implementation. The catch is, code using it must use your class as the base. That may be highly inconvenient if the other programmers wanting to use your package have already developed their own class hierarchy independently. In Java, a class can inherit from only one base class.

*When to Use Both
You can offer the best of both worlds, an interface and an abstract class. Implementors can ignore your abstract class if they choose. The only drawback of doing that is calling methods via their interface name is slightly slower than calling them via their abstract class name.

Reference:http://mindprod.com/jgloss/interfacevsabstract.html

QUESTION 69
Which three statements are correct about thread's sleep method?

A. The sleep (long) method parameter defines a delay in milliseconds.
B. The sloop (long) method parameter defines a delay in microseconds.
C. A thread is guaranteed to continue execution after the exact amount of time defined in the sleep (long) parameter.
D. A thread can continue execution before the amount of time defined in the sleep (long) parameter.
E. A thread can continue execution after the amount of time defined in the sleep(long) parameter
F. Only runtime exceptions are thrown by the sleep method.
G. A thread loses all object monitors (lock flags) when calling the sleep method.

Correct Answer: ACE
Section: (none)
Explanation

Explanation/Reference:
Explanation: public static void sleep(long millis)
throws InterruptedException
Causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds(A, not B). The thread does not lose ownership of any monitors(not G).
Parameters:
millis - the length of time to sleep in milliseconds.

Throws:
InterruptedException - if another thread has interrupted the current thread. The interrupted status of the current thread is cleared when this exception is thrown.

QUESTION 70
Which two compile?

A. interface Compilable {
void compile();
}
B. interface Compilable {
final void compile();
}
C. interface Compilable {
static void compile();
}
D. interface Compilable {
abstract void compile));
}
E. interface Compilable {
protected abstract void compile ();
}

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 71
Given this code fragment:

```
public static void main(String[] args) {

try {

String query = "SELECT * FROM Item";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query);

ResultSetMetaData rsmd = rs.getMetaData(); // Line 14

int colCount = rsmd.getColumnCount();

while (rs.next()) {

for (int i = 1; i <= colCount; i++) {

System.out.print(rs.getObject(i) + " "); // Line 17

}

System.out.println();

}
} catch (SQLException se) {
System.out.println("Error");
}
```

Assume that the SQL query returns records. What is the result?

A. Compilation fails due to error at line 17
B. The program prints Error
C. The program prints each record
D. Compilation fails at line 14

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 72
Given:

```
public class Test {
void display(String[] arr) {
try {
System.out.print(arr[2]);
} catch(ArrayIndexOutOfBoundsException |
NullPointerException e) {
e = new Exception();
throw e;
}
}
public static void main(String[] args) throws Exception {
try {
String[] arr = {"Unix","Solaris",null};
new Test().display(arr);
} catch(Exception e) {
System.err.print(e.getClass());
}
}
}
```

What is the result?

A. Null
B. class java.lang.ArrayIndexOutOfBoundsException
C. class java.lang.NullPointerException
D. class java.lang.Exception
E. Compilation fails.

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation: error: incompatible types
e = new Exception();
required: RuntimeException
found: Exception

QUESTION 73
Which is a key aspect of composition?

A. Using inheritance
B. Method delegation
C. Creating abstract classes
D. Implementing the composite interface

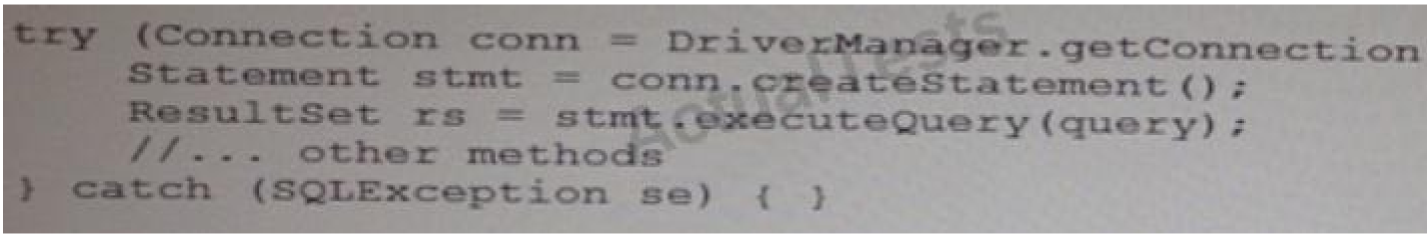
Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: In the composition approach, the subclass becomes the "front-end class," and the superclass becomes the "back-end class." With inheritance, a subclass automatically inherits an implemenation of any non-private superclass method that it doesn't override. With composition, by contrast, the front-end class must explicitly invoke a corresponding method in the back-end class from its own implementation of the method. This explicit call is sometimes called "forwarding" or "delegating" the method invocation to the back-end object.

Note:Composition means the same as:
* contains
* is part of

Note2:As you progress in an object-oriented design, you will likely encounter objects in the problem domain that contain other objects. In this situation you will be drawn to modeling a similar arrangement in the design of your solution. In an object-oriented design of a Java program, the way in which you model objects that contain other objects is with composition, the act of composing a class out of references to other objects. With composition, references to the constituent objects become fields of the containing object. To use composition in Java, you use instance variables of one object to hold references to other objects.

QUESTION 74
Given the code fragment:



```
try (Connection conn = DriverManager.getConnection
     Statement stmt = conn.createStatement();
     ResultSet rs = stmt.executeQuery(query);
     //... other methods
} catch (SQLException se) { }
```

What change should you make to apply good coding practices to this fragment?

A. Add nested try-with-resources statements for the statement and Resulset declarations.
B. Add the statement and Resulset declarations to the cry-with-resources statement.
C. Add a finally clause after the catch clause.
D. Rethrow SQLException.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The finally block always executes when the try block exits. This ensures that the finally block is executed even if an unexpected exception occurs. But finally is useful for more than just exception handling -- it allows the programmer to avoid having cleanup code accidentally bypassed by a return, continue, or break. Putting cleanup code in a finally block is always a good practice, even when no exceptions are anticipated.

QUESTION 75
Which statement creates a low overhead, low-contention random number generator that is isolated to thread to generate a random number between 1 and 100?

A. int i = ThreadLocalRandom.current().nextInt(1, 101);
B. int i = ThreadSafeRandom.current().nextInt(1, 101);
C. int i = (int) Math.random()*100+1;
D. int i = (int) Math.random(1, 101);
E. int i = new random().nextInt(100)+1;

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: public class ThreadLocalRandom
extends Random'

A random number generator isolated to the current thread. Like the global Random generator used by the Math class, a ThreadLocalRandom is initialized with an internally generated seedthat may not otherwise be modified. When applicable, use of ThreadLocalRandom rather than shared Random objects in concurrent programs will typically encounter much less overhead and contention. Use of ThreadLocalRandom is particularly appropriate when multiple tasks (for example, each a ForkJoinTask) use random numbers in parallel in thread pools. Usages of this class should typically be of the form: ThreadLocalRandom.current().nextX(...) (where X is Int, Long, etc). When all usages are of this form, it is never possible to accidently share a ThreadLocalRandom across multiple threads.

This class also provides additional commonly used bounded random generation methods.

Reference:Class ThreadLocalRandom

QUESTION 76
Given:

What is the result?

A. fast slow
B. fast goes
C. goes goes
D. fast fast
E. fast followed by an exception
F. Compilation fails

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:
Explanation: Line:Vehicle v = new Sportscar();
causes compilation failure:

```
error: cannot find symbol
Vehicle v = new Sportscar();
symbol: class Sportscar
location: class VehicleTest
```

QUESTION 77
Given:

```
class Plant {

abstract String growthDirection();

}

class Embryophyta extends Plant {

String growthDirection() { return "Up " }

}

public class Garden {

public static void main(String[] args) {

Embryophyta e = new Embryophyta();

Embryophyta c = new Carrot();

System.out.print(e.growthDirection() +
```

A. growthDirection());
}
}
What is the result?
B. Up Down
C. Up Up
D. Up null

E. Compilation fails
F. An exception is thrown at runtime

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation: Exception in thread "main" java.lang.ExceptionInInitializerError at garden.Garden.main

Caused by: java.lang.RuntimeException: Uncompilable source code - garden.Plant is not abstract and does not override abstract method growthDirection() in garden.Plant

QUESTION 78
Given three resources bundles with these values set for menu1: (the default resource bundle in US English.)

English US Resource Bundle

Menu1 = small

French Resource Bundle

Menu1 = petit

Chinese Resource Bundle

Menu1 =

And given the code fragment:

Locale.setDefault(new Locale("es", "ES")); // Set default to Spanish and Spain

Locale loc1 = Locale.getDefault();

ResourceBundle message = ResourceBundle.getBundle("MessageBundle", loc1);

System.out.println(message.getString("menu1"));

What is the result?

A. No message is printed
B. Petit
C.
D. A runtime error is produced

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: There is no Spanish resource bundle. The following runtime error will occur:

Exception in thread "main" java.util.MissingResourceException: Can't find bundle for base name MessageBundle, locale es_ES

QUESTION 79
Given:

```
import java.util.concurrent.atomic.AtomicInteger;

public class Incrementor {

public static void main(String[] args) {

AtomicInteger[] var = new AtomicInteger[5];

for (int i = 0; i < 5; i++) {

var[i] = new AtomicInteger();

}

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

var[i].incrementAndGet();

if (i ==2)

var[i].compareAndSet(2,4);

System.out.print(var[i] + " ");

}

}

}
```

What is the result?

A. 1 1 1 1 1
B. 1 2 3 4 5
C. 0 1 2 3 4
D. 0 1 4 3 4

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 80
Given:

```
public class SuperThread extends Thread {

public void run(String name) {

System.out.print("Thread");

}

public void run(Runnable r) {

r = new runnable() {
```

```
public void run() {

System.out.print("Runnable");

}

};

}

public static void main(String[] args) {

Thread t = new SuperThread();
```

A. start();
   }
   }
   Which two are true?
B. Thread is printed
C. Runnable is printed
D. No output is produced
E. No new threads of execution are started within the main method
F. One new thread of execution is started within the main method
G. Two new threads of exclusion are started within the main method

Correct Answer: CE
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 81
Give:

```
class Fibonacci extends RecursiveTask<Integer> {

final int n;

Fibonacci(int n) { this.n = n; }

Integer compute() {

if (n <= 1)

return n;

Fibonacci f1 = new Fibonacci(n - 1);

f1.fork();

Fibonacci f2 = new Fibonacci(n - 2);

return f2.compute() + f1.join(); // Line X

}
```

}

Suppose that line X is replace with:

return f1.join()+f2.compute() ; // Line X

What is the likely result?

A. The program produces the correct result, with similar performance to the original.
B. The program produces the correct result, with performance degraded to the equivalent of being single-threaded.
C. The program produces an incorrect result.
D. The program goes into an infinite loop.
E. An exception is thrown at runtime.
F. The program produces the correct result, with better performance than the original.

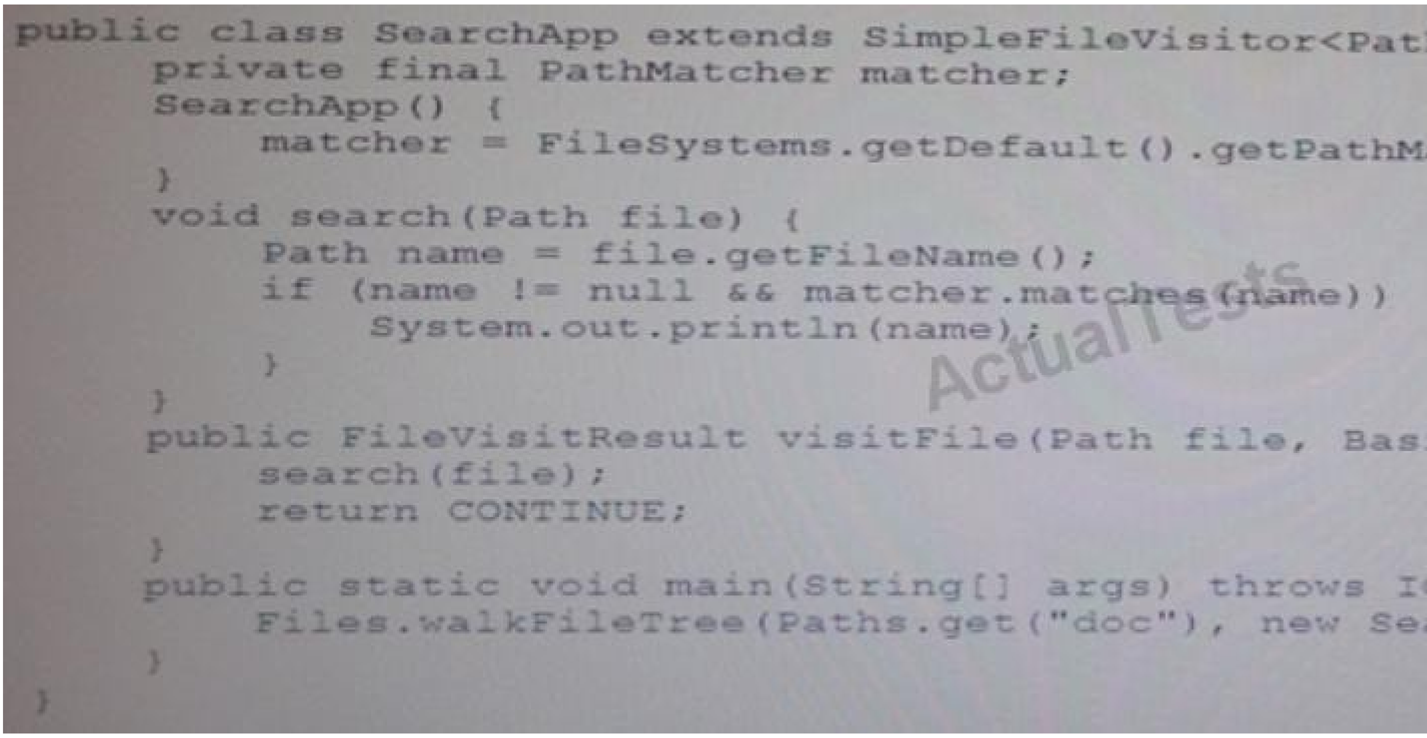
Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: join()does not proceed until the task's result has been computed. Here we start to wait before doing the computing. The code will not finish.

QUESTION 82
Given the following files in doc directory:

- Index.htm
- Service.html
- Logo.gif
- Title.jpg

And the code fragment:



```
public class SearchApp extends SimpleFileVisitor<Pat
     private final PathMatcher matcher;
     SearchApp() {
          matcher = FileSystems.getDefault().getPathMa
     }
     void search(Path file) {
          Path name = file.getFileName();
          if (name != null && matcher.matches(name))
               System.out.println(name);
          }
     }
     public FileVisitResult visitFile(Path file, Bas
          search(file);
          return CONTINUE;
     }
     public static void main(String[] args) throws I
          Files.walkFileTree(Paths.get("doc"), new Se
     }
 }
```

What is the result, if doc is present in the current directory?

A. No output is produced.

B. index.htm

C. index.htm
userguide.txt
logo.gif

D. index.htm
service.html
userguide.txt
logo.gif

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The Glob search expression is defined through "glob:*.htm, html, xml" Only the file name index.htm matches this pattern.

QUESTION 83
Given:

```
public class Test {

public static void main(String[] args) {

String[] arr = {"SE","ee","ME"};

for(String var : arr) {

try {

switch(var) {

case "SE":

System.out.println("Standard Edition");

break;

case "EE":

System.out.println("Enterprise Edition");

break;

default: assert false;

}

} catch (Exception e) {

System.out.println(e.getClass()); }

}

}

}
```

And the commands:

```
javac Test.java
```

```
java ea Test
```

What is the result?

A. Compilation fails

B. Standard Edition
Enterprise Edition
Micro Edition

C. Standard Edition
class java.lang.AssertionError
Micro Edition

D. Standard Edition is printed and an Assertion Error is thrown

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The command line :
javac Test.java
willcompilethe program.

As for command line:
java ea Test
First the code will produce the output:
Standard Edition
See Note below.

The ea option will enable assertions.
This will make the following line in the switch statement to be run:
default: assert false;
This will thow an assertion error. This error will be caught. An the class of the assertion error (class java.lang.AssertionError) will be printed by the following line:
System.out.println(e.getClass());

Note:The java tool launches a Java application. It does this by starting a Java runtime environment, loading a specified class, and invoking that class's main method. The method declaration must look like the following:
public static void main(String args[])

Paramater ea:
-enableassertions[:<package name>"..." | :<class name>] -ea[:<package name>"..." | :<class name>]
Enable assertions. Assertions are disabled by default. With no arguments, enableassertions or -ea enables assertions.

Note 2:
An assertion is a statement in the JavaTM programming language that enables you to test your assumptions about your program.
Each assertion contains a boolean expression that you believe will be true when the assertion

executes. If it is not true, the system will throw an error.

public classAssertionError
extendsError
Thrown to indicate that an assertion has failed.

Note 3:
The javac command compiles Java source code into Java bytecodes. You then use the Java interpreter - the java command - to interprete the Java bytecodes.

Reference:java - the Java application launcher

Reference:java.langClass AssertionError

QUESTION 84
Given the classes:

```
class Pupil {

String name = "unknown";

public String getName() {return name;}

}

class John extends Pupil {

String name = "John";

}

class Harry extends Pupil {

String name = "Harry";

public String getName() {return name;}

}

public class Director {
public static void main(String[] args) {

Pupil p1 = new John();

Pupil p2 = new Harry();

System.out.print(p1.getName() + " ");

System.out.print(p2.getName());

}

}
```

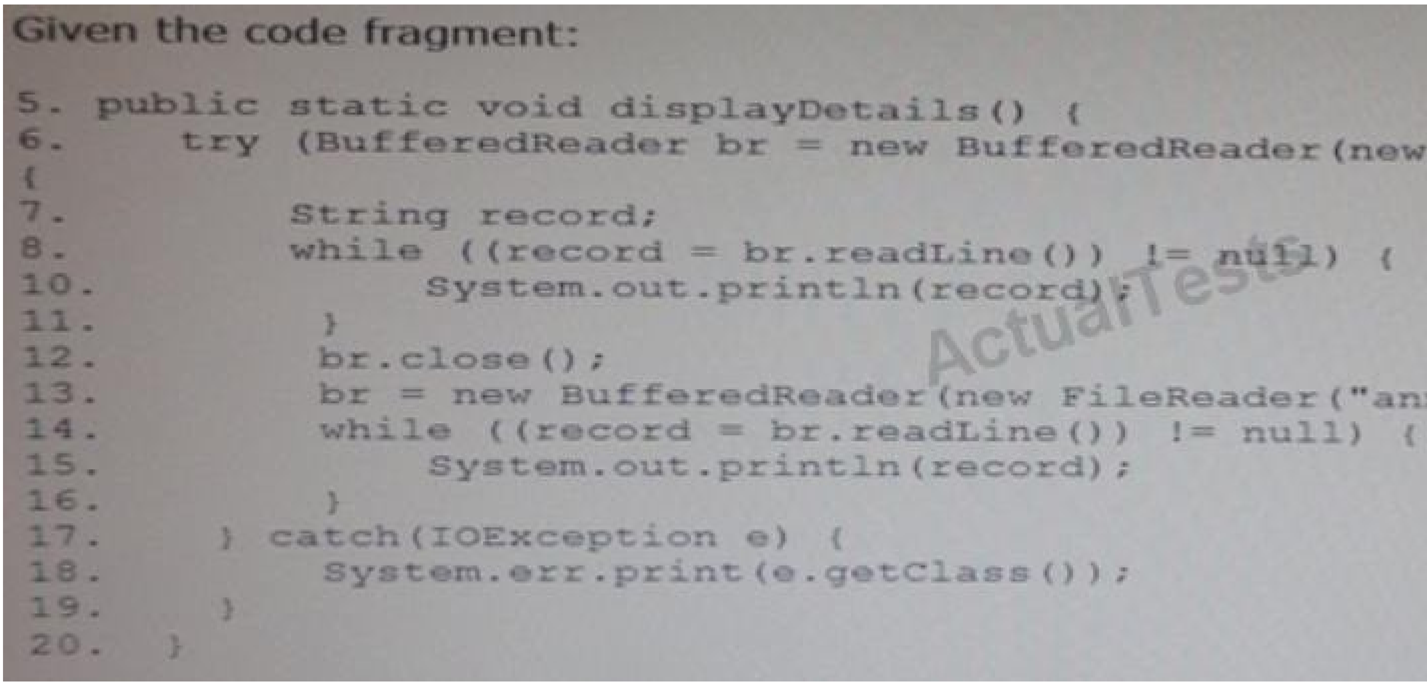
What is the result?

A. John Harry
B. unknown Harry
C. john unknown
D. unknown unknown
E. Compilation fails.
F. An exception is thrown at runtime.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 85
Given the code fragment:



Given the code fragment:

```
5. public static void displayDetails() {
6.     try (BufferedReader br = new BufferedReader(new
{
7.          String record;
8.          while ((record = br.readLine()) != null) {
10.              System.out.println(record);
11.          }
12.          br.close();
13.          br = new BufferedReader(new FileReader("an
14.          while ((record = br.readLine()) != null) {
15.              System.out.println(record);
16.          }
17.     } catch(IOException e) {
18.          System.err.print(e.getClass());
19.     }
20.  }
```

What is the result, if the file salesreport.dat does not exist?

A. Compilation fails only at line 6
B. Compilation fails only at line 13
C. Compilation fails at line 6 and 13
D. Class java.io.IOException
E. Class java.io.FileNotFoundException

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Compilation works fine.
There will be a runtime error at line 6 (as the file salesreport.dat) does not exist.
This will be caught as an IOException at line 17.

QUESTION 86
Given:

import java.util.ArrayDeque;

import java.util.Deque;

public class Counter {

public static void main(String[] args) {

Deque<String> deq = new ArrayDeque<String>(2);

deq.addFirst("one");

```
deq.addFirst("two");

deq.addFirst("three"); // Line 9

System.out.print(deq.pollLast());

System.out.print(deq.pollLast());

System.out.print(deq.pollLast()); // Line 12

}

}
```

What is the result?

A. An exception is thrown at runtime on line 9.
B. An exception is thrown at runtime on line 12
C. one two null
D. one two three
E. two one null
F. three two one

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 87
Which class safely protects the doIt () method from concurrent thread access?

A.
```
class SafeMethod {
Static int ID = 0;
Public static void doIt(String s) {
Synchronized (s) {
System.out.println("Name:"+ s +"ID:"+ id++);
}
}
}
```

B.
```
class SafeMethod {
Static int ID = 0;
Public static void doIt(String s) {
Synchronized (new object () ) {
System.out.println("Name:"+ s +"ID:"+ id++);
}
}
}
```

C.
```
class SafeMethod {
Static int ID = 0;
Public static void doIt(String s) {
Synchronized (this) {
System.out.println("Name:"+ s +"ID:"+ id++);
}
}
}
```

D.
```
class SafeMethod {
Static int ID = 0;
Public static void doIt(String s) {
```

```
Synchronized (SafeMethod.class) {
System.out.println("Name:"+ s +"ID:"+ id++);
}
}
}
```

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: It should be pointed out that:

```
public void blah() {
synchronized (this) {
// do stuff
}
}
```

is semantically equivalent to:

```
public synchronized void blah() {
// do stuff
}
```

Incorrect answer:
B: A constructor cannot be synchronized.

QUESTION 88
Given:

```
interface Event {

String getCategory();

}

public class CueSports {

public String getCategory() {

return "Cue sports";

}

}

public class Snooker extends CueSports implements Event { // Line 9

public static void main(String[] args) {

Event obj1 = new Snooker(); // Line 11
CueSports obj2 = new Snooker(); // Line 12

System.out.print(obj1.GetCategory() + ", " + obj2.getCategory());

}

}
```

What is the result?

A. Cue sports, Cue sports

B. Compilation fails at line 9
C. Compilation fails at line 11
D. Compilation fails at line 12
E. Compilation fails at line 13

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: Class Snooker is public. Should be declared in a separate file.

QUESTION 89
Which is a factory method from the java.text.NumberFormat class?

A. format (long number)
B. getInstance()
C. getMaxiraumFractionDigits ()
D. getAvailableLocales ()
E. isGroupingUsed()

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: To obtain a NumberFormat for a specific locale, including the default locale, call one of NumberFormat's factory methods, such as getInstance().

Reference:java.textClass DecimalFormat

QUESTION 90
Given:

```
class InvalidAgeException extends IllegalArgumentException { }
public class Tracker {

void verify (int age) throws IllegalArgumentException {

if (age < 12)

throw new InvalidAgeException ();

if (age >= 12 && age <= 60)

System.out.print("General category");

else

System.out.print("Senior citizen category");

}

public static void main(String[] args) {

int age = Integer.parseInt(args[1]);

try {

new Tracker().verify(age);
}

catch (Exception e) {

System.out.print(e.getClass());

}

}

}
```

And the command-line invocation:

Java Tracker 12 11

What is the result?

A. General category
B. class invalidAgeException
C. class java.lang.IllegalArgumentntException
D. class java.lang.RuntimeException

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The second argument 11 makes the program to throw an InvalidAgeException due to the line:

if (age < 12)
throw new InvalidAgeException ();

QUESTION 91
Which code fragment correctly appends "Java 7" to the end of the file /tmp/msg.txt?

A. Filewriter w = new FileWriter("/tmp/msg.txt");
B. append("Java 7");
C. close();
D. Filewriter w = new FileWriter("/tmp/msg.txt", true);
E. append("Java 7");
F. close();
G. Filewriter w = new FileWriter("/tmp/msg.txt", FileWriter.MODE_APPEND);
H. append("Java 7");
I. close();
J. Filewriter w = new FileWriter("/tmp/msg.txt", Writer.MODE_APPEND);
K. append("Java 7");
L. close();

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: FileWriter(File file, boolean append)
Constructs a FileWriter object given a File object.
If the second argument is true, then bytes will be written to the end of the file rather than the beginning.
Parameters:

file - a File object to write to
append - if true, then bytes will be written to the end of the file rather than the beginning

QUESTION 92
Given the code format:
SimpleDateFormat sdf;

Which code statements will display the full text month name?

A. sdf = new SimpleDateFormat ("mm", Locale.UK);
System.out.println("Result:", sdf.format(new date()));
B. sdf = new SimpleDateFormat ("MM", Locale.UK);
System.out.println("Result:", sdf.format(new date()));
C. sdf = new SimpleDateFormat ("MMM", Locale.UK);
System.out.println("Result:", sdf.format(new date()));
D. sdf = new SimpleDateFormat ("MMMM", Locale.UK);
System.out.println("Result:", sdf.format(new date()));

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Typical output would be
Current Month in M format : 2
Current Month in MM format : 02
Current Month in MMM format : Feb
Current Month in MMMM format : February

QUESTION 93
Given:

class Car implements TurboVehicle, Steerable {

// Car methods > interface Convertible

{

// Convertible methods

}

public class SportsCar extends Car implements Convertible {

}

Which statement is true?

A. SportsCar must implement methods from TurboVehicle and steerable
B. SportsCar must override methods defined by car.
C. SportsCar must implement methods define by convertible.
D. Instances of car can invoke convertible methods.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: To declare a class that implements an interface, you include an implements clause in the class declaration.
By convention, theimplements clause follows the extends clause, if there is one.

Here are the some point that must be considered while implementing an interface (or interfaces) into a java class.

A class implementing an interface must either implement all the methods of that interface otherwise known as the abstract class.
A class in java may extend at most one superclass because java does not allow multiple inheritance, by it may implement more than one interface. Multiple inheritance in java is achieved through the interfaces. When a class implements more than one interface then implement statement requires a comma-separated list of interfaces to be implement by that class.

QUESTION 94
Given:

```
public class Dog {

protected String bark() {return "woof "; }

}

public class Beagle extends Dog {

private String bark() { return "arf "; }

}

public class TestDog {

public static void main(String[] args) {

Dog[] dogs = {new Dog(), new Beagle()};

for(Dog d: dogs)

System.out.print(d.bark());

}

}
```

What is the result?

A. woof arf
B. woof woof
C. arf arf
D. A RuntimeException is generated
E. The code fails to compile

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 95
Given:

```
1. interface Writable {

2. void write (String s);
}

abstract class Writer implements Writable {
// Line ***
}
```

Which two statements are true about the writer class?

A. It compiles without any changes.
B. It compiles if the code void write (String s); is added at line***.
C. It compiles if the code void write (); is added at line ***.
D. It compiles if the code void write (string s) { } is added at line ***.
E. It compiles if the code write () {}is added at line ***.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: An abstract class does not need to implement the interface methods.

QUESTION 96
Given the class?

```
public class Name implements Comparable<Name> {

String first, last;

public Name(String first, String last) {

this.first = first;

this.last = last;

}

public int compareTo (Name n) {

int cmpLast = last.compareTo(n.last);

return cmpLast != 0 ? cmpLast : first.compareTo(n.first);

}

public String toString() {

return first + " " + last;

}

}
```

and the code fragment:

```
ArrayList<Name> list = new ArrayList<Name>();

list.add (new Name("Joe","Shmoe"));
list.add (new Name("John","Doe"));

list.add (new Name("Jane","Doe"));

Collections.sort(list);
for (Name n : list) {

System.out.println(n);

}
```

What is the result?

A. Jane Doe
John Doe
Joe Shmoe

B. John Doe
Jane Doe
Joe Shmoe

C. Joe Shmoe
John Doe
Jane Doe

D. Joe Shmoe
Jane Doe
John Doe

E. Jane Doe
Joe Shmoe
John Doe

F. John Doe
Joe Shmoe
Jane Doe

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The list will be sortedalpabethically.
Output will be:
Jane Doe
John Doe
Joe Shmoe

QUESTION 97
Given the code fragment:

```
public void infected() {

System.out.print("before ");

try {
int i = 1/0;

System.out.print("try ");

} catch(Exception e) {

System.out.print("catch ");

throw e;

} finally {
```

```
System.out.print("finally ");

}

System.out.print("after ");

}
```

What is the result when infected() is invoked?

A. before try catch finally after
B. before catch finally after
C. before catch after
D. before catch finally
E. before catch

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The following line throws and exception:
int i = 1/0;

This exception is caught by:
catch(Exception e) {
System.out.print("catch ");
throw e;

Lastly, the finally statement is run as the finally block always executes when the try block exits. This ensures that the finally block is executed even if an unexpected exception occurs.

Reference: Java Tutorial,The finally Block

QUESTION 98
Given the two Java classes:

```
public class Word {

private Word(int length) {}

protected Word(String w) {}

}

public class Buzzword extends Word {

public Buzzword() {

// Line ***, add code here

}

public Buzzword(String s) {

super(s);

}

}
```

Which two code snippets, added independently at line ***, can make the Buzzword class compile?

A. this ();
B. this (100);
C. this ("Buzzword");
D. super ();
E. super (100);
F. super ("Buzzword");

Correct Answer: AF
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 99
Given:

```
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Test {

private static String REGEX = "\\Sto\\S|\\bo\\b";

private static String INPUT = "Nice to see you,to,be fine.";

private static String REPLACE =",";

public static void main(String[] args) {

Pattern p = Pattern.compile(REGEX);

Matcher m = p.matcher(INPUT);

INPUT = m.replaceAll(REPLACE);

System.out.println(INPUT);

}

}
```

What is the result?

A. Nice to see you, be fine
B. Nice, see you, be fine
C. Nice, see you, to, be fine
D. Nice, see you, be fine
E. Nice to see y, u, be fine

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The text to is removed (replaced by the empty string).

QUESTION 100

Given the following code fragment:

```
10. p1 = paths.get("report.txt");

11. p2 = paths.get("company");
12. / / insert code here
```

Which code fragment, when inserted independently at line 12, move the report.txt file to the company directory, at the same level, replacing the file if it already exists?

A. Files.move(p1, p2, StandardCopyOption.REPLACE_EXISTING, StandardCopyOption.ATOMIC_MOVE);
B. Files.move(p1, p2, StandardCopyOption.REPLACE_Existing, LinkOption.NOFOLLOW_LINKS);
C. Files.move (p1, p2, StandardCopyOption.REPLACE_EXISTING, LinkOption.NOFOLLOW_LINKS);
D. Files.move(p1, p2, StandardCopyOption.REPLACE_EXISTING, StandardCopyOption.copy_ATTRIBUTES, StandrardCopyOp)
E. Files.move (p1, p2 StandardCopyOption.REPLACE_EXISTING, StandardCopyOption.copy_ATTRIBUTES, LinkOption.NOF)

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: Moving a file is equally as straight forward move(Path source, Path target, CopyOption... options);
The available StandardCopyOptions enums available are:
StandardCopyOption.REPLACE_EXISTING
StandardCopyOption.ATOMIC_MOVE
If Files.move is called with StandardCopyOption.COPY_ATTRIBUTES an UnsupportedOperationException is thrown.

QUESTION 101
Given:

```
import java.util.*;

public class SearchText {

public static void main(String[] args) {

Object[] array1 = new Object[3];

array1[0] = "foo";
array1[0] = 1;

array1[0] = 'a';

int index = Arrays.binarySearch(array1, "bar");

System.out.println(index);

}

}
```

What is the result?

A. 1
B. 0
C. 2
D. Compilation fails
E. An exception is thrown at runtime

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code compiles fine.
An exception is thrown at runtime due to data type comparison mismatch:

```
Exception in thread "main" java.lang.ClassCastException: java.lang.String cannot be cast to
java.lang.Integer
at java.lang.Integer.compareTo(Integer.java:52)
at java.util.Arrays.binarySearch0(Arrays.java:1481)
at java.util.Arrays.binarySearch(Arrays.java:1423)
at searchtext.SearchText.main(SearchText.java:22)
```

Note:binarySearch

```
public static int binarySearch(char[] a,
char key)
```

Searches the specified array of chars for the specified value using the binary search algorithm. The array must be sorted (as by the sort method, above) prior to making this call. If it is not sorted, the results are undefined. If the array contains multiple elements with the specified value, there is no guarantee which one will be found.
Parameters:
a - the array to be searched.
key - the value to be searched for.
Returns:
index of the search key, if it is contained in the list; otherwise, (-(insertion point) - 1). The insertion point is defined as the point at which the key would be inserted into the list: the index of the first element greater than the key, or list.size(), if all elements in the list are less than the specified key. Note that this guarantees that the return value will be >= 0 if and only if the key is found.

QUESTION 102
Give:

```
public class Test {

public static void main(String[] args) {

String svar= "sports cars";

svar.replace(svar,"convertibles");

System.out.printf("There are %3$s %2$s and %d trucks.",5,svar,2+7);

}

}
```

What is the result?

A. There are 27 sports cars and 5 trucks

B. There are 27 convertibles and 5 trucks
C. There are 9 sports cars and 5 trucks
D. There are 9 convertibles and 5 trucks
E. IllegalFormatConversionException is thrown at runtime

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 103
Which two statements are true about Rowset subinterfaces?

A. A JdbcRowSet object provides a JavaBean view of a result set.
B. A cachedRowset provides a connected view of the database.
C. A FilteredRowSet object filter can be modified at any time.
D. A webRowset returns JSON-formatted data.

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation: A:a JdbcRowSet object can be one of the Beans that a tool makes available for composing an application. Because a JdbcRowSet is a connected rowset, that is, it continually maintains its connection to a database using a JDBC technology-enabled driver, it also effectively makes the driver a JavaBeans component.
C:The FilteredRowSet range criterion can be modified by applying a new Predicate object to the FilteredRowSet instance at any time. This is possible if no additional references to the FilteredRowSet object are detected. A new filter has has an immediate effect on criterion enforcement within the FilteredRowSet object, and all subsequent views and updates will be subject to similar enforcement.

Reference: javax.sql Interface RowSet

QUESTION 104
Given two classes in separate files:

```
package a.b;

/ / import statement

public class parent

child c = new child();

package a.b.c;

public class child{

}
```

Which two import statements can make the a.b.parent class compliable?

A. import a.b.c.Parent;
B. import a.b.c.Child;
C. import a.b.c.*;
D. import a.b.*;

E. import a.*;

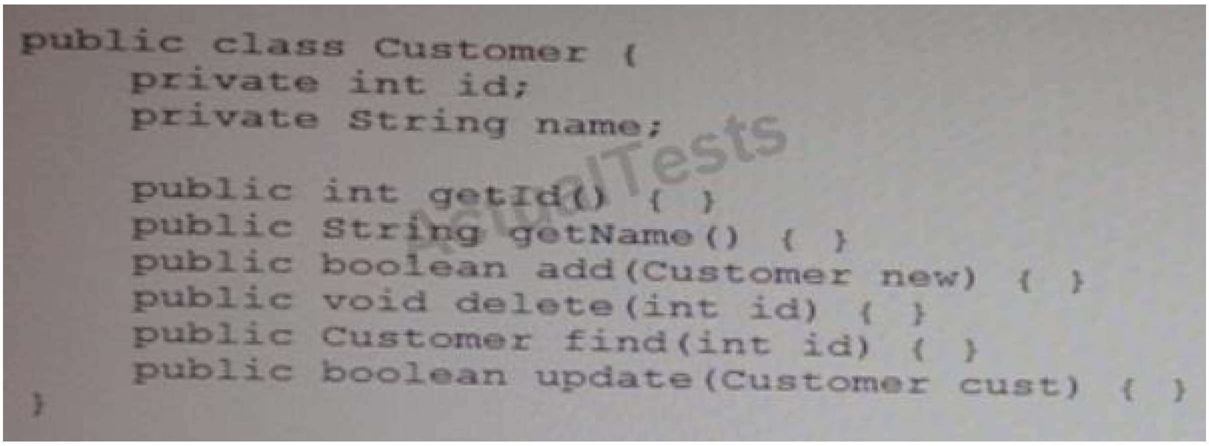
Correct Answer: BC
Section: (none)
Explanation

Explanation/Reference:
Explanation: B:To import a specific member into the current file, put an import statement at the beginning of the file before any type definitions but after the package statement, if there is one.

C:To import all the types contained in a particular package, use the import statement with the asterisk (*) wildcard character.

Reference: The Java Tutorials,Using Package Members

QUESTION 105
Given:



```
public class Customer {
    private int id;
    private String name;

    public int getId() { }
    public String getName() { }
    public boolean add(Customer new) { }
    public void delete(int id) { }
    public Customer find(int id) { }
    public boolean update(Customer cust) { }
}
```

What two changes should you make to apply the DAO pattern to this class?

A. Make the Customer class abstract.
B. Make the customer class an interface.
C. Move the add, delete, find, and update methods into their own implementation class.
D. Create an interface that defines the signatures of the add, delete, find, and update methods.
E. Make the add, delete, and find, and update methods private for encapsulation.
F. Make the getName and getID methods private for encapsulation.

Correct Answer: CD
Section: (none)
Explanation

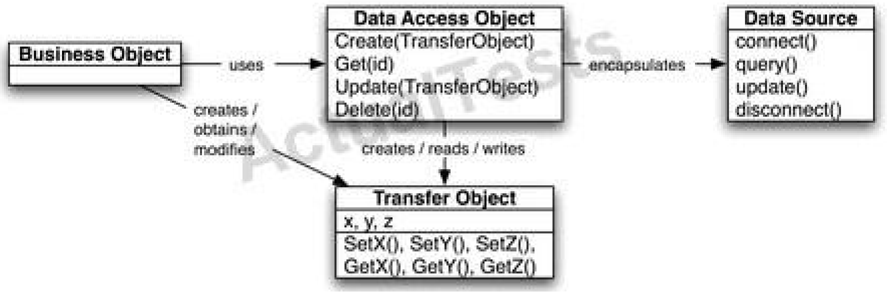
Explanation/Reference:
Explanation: C:The methods related directly to the entity Customer is moved to a new class.
D: Example (here Customer is the main entity):
public class Customer {
private final String id;
private String contactName;
private String phone;
public void setId(String id) { this.id = id; }

public String getId() { return this.id; }
public void setContactName(String cn) { this.contactName = cn;} public String getContactName() { return

this.contactName; } public void setPhone(String phone) { this.phone = phone; } public String getPhone()
{ return this.phone; }
}

public interface CustomerDAO {
public void addCustomer(Customer c) throws DataAccessException; public Customer getCustomer(String id) throws DataAccessException; public List getCustomers() throws DataAccessException; public void removeCustomer(String id) throws DataAccessException; public void modifyCustomer(Customer c) throws DataAccessException; }

Note:DAO Design Pattern
*Abstracts and encapsulates all access to a data source *Manages the connection to the data source to obtain and store data *Makes the code independent of the data sources and data vendors (e.g. plain-text, xml, LDAP, MySQL, Oracle, DB2)



QUESTION 106
Which code fragment is required to load a JDBC 3.0 driver?

A. DriverManager.loadDr iver ("org.xyzdata.jdbc.NetworkDriver");
B. Class.forName("org.xyzdata.jdbc-NetworkDriver");
C. Connection con = Connection.getDriver
("jdbc:xyzdata://localhost:3306/EmployeeDB");
D. Connection con = DriverManager.getConnection
("jdbc:xyzdata://localhost:3306/EmployeeDB");

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: In previous versions(prior to 4.0)of JDBC, to obtain a connection, you first had to initialize your JDBC driver by calling the method Class.forName. This methods required an object of type java.sql.Driver.

Note:
DriverManager: This fully implemented class connects an application to a data source, which is specified by a database URL. When this class first attempts to establish a connection, it automatically loads any JDBC 4.0 drivers found within the class path. Note that your application must manually load any JDBC drivers prior to version 4.0.

QUESTION 107
Given:

public class CowArray extends Thread {

static List<Integer> myList = new CopyOnWriteArrayList<Integer>();

public static void main(String[] args) {

myList.add(11);

```
myList.add(22);
myList.add(33);
myList.add(44);
new CowArray().start();
for(Integer i: myList) {
try { Thread.sleep(1000); }
catch (Exception e) { System.out.print("e1 "); }
System.out.print(" " +i);
}
}
public void run() {
try { Thread.sleep(500); }
catch (Exception e) { System.out.print("e2 "); }
myList.add(77);
System.out.print("size: " + myList.size() + ", elements:");
}
}
```

What is the most likely result?

A. size: 4, elements: 11 22 33 44
B. size: 5, elements: 11 22 33 44
C. size: 4, elements: 11 22 33 44 77
D. size: 5, elements: 11 22 33 44 77
E. a ConcurrentModification Exception is thrown

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 108
Which two are true about Singletons?

A. A Singleton must implement serializable.
B. A Singleton has only the default constructor.
C. A Singleton implements a factory method.
D. A Singleton improves a class's cohesion.
E. Singletons can be designed to be thread-safe.

Correct Answer: BE
Section: (none)

Explanation

Explanation/Reference:
Explanation: B:The Singleton Pattern is one of the commonly used design templates when there needs to be a control on how an object can be created. This design pattern proposes that at any time there can only be one instance of a singleton (object) created by the JVM. The Singleton class's default constructor is made private, which prevents the direct instantiation of the object by others (Other Classes).
E:The singletonpattern makes unit testing far more difficult,as it introduces global state into an application. It should also be noted that this pattern reduces the potential for parallelism within a

program, because access to the singleton in a multi-threaded context must be serialised, e.g., by locking.

QUESTION 109
Given:

```
public enum Direction {

NORTH, EAST, SOUTH, WEST

}
```

Which statement will iterate through Direction?

A. for (Direction d : Direction.values()){
   //
   }
B. for (Direction d : Direction.asList()){
   //
   }
C. for (Direction d : Direction.iterator()){
   //
   }
D. for (Direction d : Direction.asArray()){
   //
   }

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The static values() method of an enum type returns an array of the enum values. The foreach loop is a good way to go over all of them.

```
//... Loop over all values.
for (Direction d : Direction.values()){
System.out.println(d); // PrintsNORTH, EAST, ...
}
```

QUESTION 110
What are two differences between Callable and Runnable?

A. A callable can return a value when executing, but a Runnable cannot.
B. A callable can be executed by a ExecutorService, but a Runnable cannot.
C. A Callable can be passed to a Thread, but a Runnable cannot.
D. A callable can throw an Exception when executing, but a Runnable cannot.

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:

Explanation: The Callable interface is similar to Runnable, in that both are designed for classes whose instances are potentially executed by another thread. A Runnable, however, does not return a result and cannot throw a checked exception.

QUESTION 111
Which three are true?

A. A setAutoCommit (False) method invocation starts a transaction context.
B. An instance of savepoint represents a point in the current transaction context.
C. A rollback () method invocation rolls a transaction back to the last savepoint.
D. A rollback () method invocation releases any database locks currently held by this connection object.
E. After calling rollback (mysavepoint), you must close the savepoint object by calling mySavepoint.close () .

Correct Answer: ABC
Section: (none)
Explanation

Explanation/Reference:
Explanation: A:The way to allow two or more statements to be grouped into a transaction is to disable the auto-commit mode.After the auto-commit mode is disabled, no SQL statements are committed until you call the method commit explicitly. All statements executed after the previous call to the method commit are included in the current transaction and committed together as a unit.

Note:When a connection is created, it is in auto-commit mode. This means that each individual SQL statement is treated as a transaction and is automatically committed right after it is executed. (To be more precise, the default is for a SQL statement to be committed when it is completed, not when it is executed. A statement is completed when all of its result sets and update counts have been retrieved. In almost all cases, however, a statement is completed, and therefore committed, right after it is executed.)

B:The method Connection.setSavepoint, sets a Savepoint object within the current transaction. The Connection.rollback method is overloaded to take a Savepoint argument.

C:calling the method rollback terminates a transaction and returns any values that were modified to their previous values. If you are trying to execute one or more statements in a transaction and

get a SQLException, call the method rollback to end the transaction and start the transaction all over again.

QUESTION 112
Given:

String s = new String("3");

System.out.print(1 + 2 + s + 4 + 5);

What is the result?

A. 12345
B. 3345
C. 1239
D. 339
E. Compilation fails.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: 1 and 2 are added. Then the string s is concatenated.
Finally 3 and 4 are concatenated as strings.

QUESTION 113
Which two properly implement a Singleton pattern?

A. class Singleton {
private static Singleton instance;
private Singleton () {}
public static synchronized Singleton getInstance() {
if (instance = = null) {
instance = new Singleton ();
}
return instance;
}
}

B. class Singleton {
private static Singleton instance = new Singleton();
protected Singleton () {}
public static Singleton getInstance () {

return Instance;
}
}

C. class Singleton {
Singleton () {}
private static class SingletonHolder {
private static final Singleton INSTANCE = new Singleton (); }
public static Singleton getInstance () {
return SingletonHolder.INSTANCE;
}
}

D. enum Singleton {
INSTANCE;
}

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: A: Here the method for getting the reference to the SingleTon object is correct.
B: The constructor should be private such as:

private static Singleton instance = new Singleton();

Note: Java has several design patterns Singleton Pattern being the most commonly used. Java Singleton pattern belongs to the family of design patterns, that govern the instantiation process. This design pattern proposes that at any time there can only be one instance of a singleton (object) created by the JVM.

The class's default constructor is made private, which prevents the direct instantiation of the object by others (Other Classes). A static modifier is applied to the instance method that returns the object as it then makes this method a class level method that can be accessed without creating an object.

QUESTION 114
Given the code fragment:

public class Employee {

String name;

transient String companyName;

}

public class Manager extends Employee implements Serializable {

String mgrId;

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

Manager mgr = new Manager();

What is the result?

A. M001, ,
B. M001, null, null
C. M001, Sam,
D. M001, Sam, null
E. M001, Sam, ABC Inc
F. Compilation fails
G. A NullPointerException is thrown at runtime

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 115
Select four examples that initialize a NumberFormat reference using a factory.

A. NumberFormat nf1 = new DecimalFormat();
B. NumberFormat nf2 = new DecimalFormat("0.00") ;
C. NumberFormat nf3 = NumberFormat.getInstance();
D. NumberFormat nf4 = NumberFormat.getIntegerInstance();
E. NumberFormat nf5= DecimalFormat.getNumberInstance ();
F. NumberFormat nf6 = Number Format.getCurrecyInstance () ;

Correct Answer: CDEF
Section: (none)
Explanation

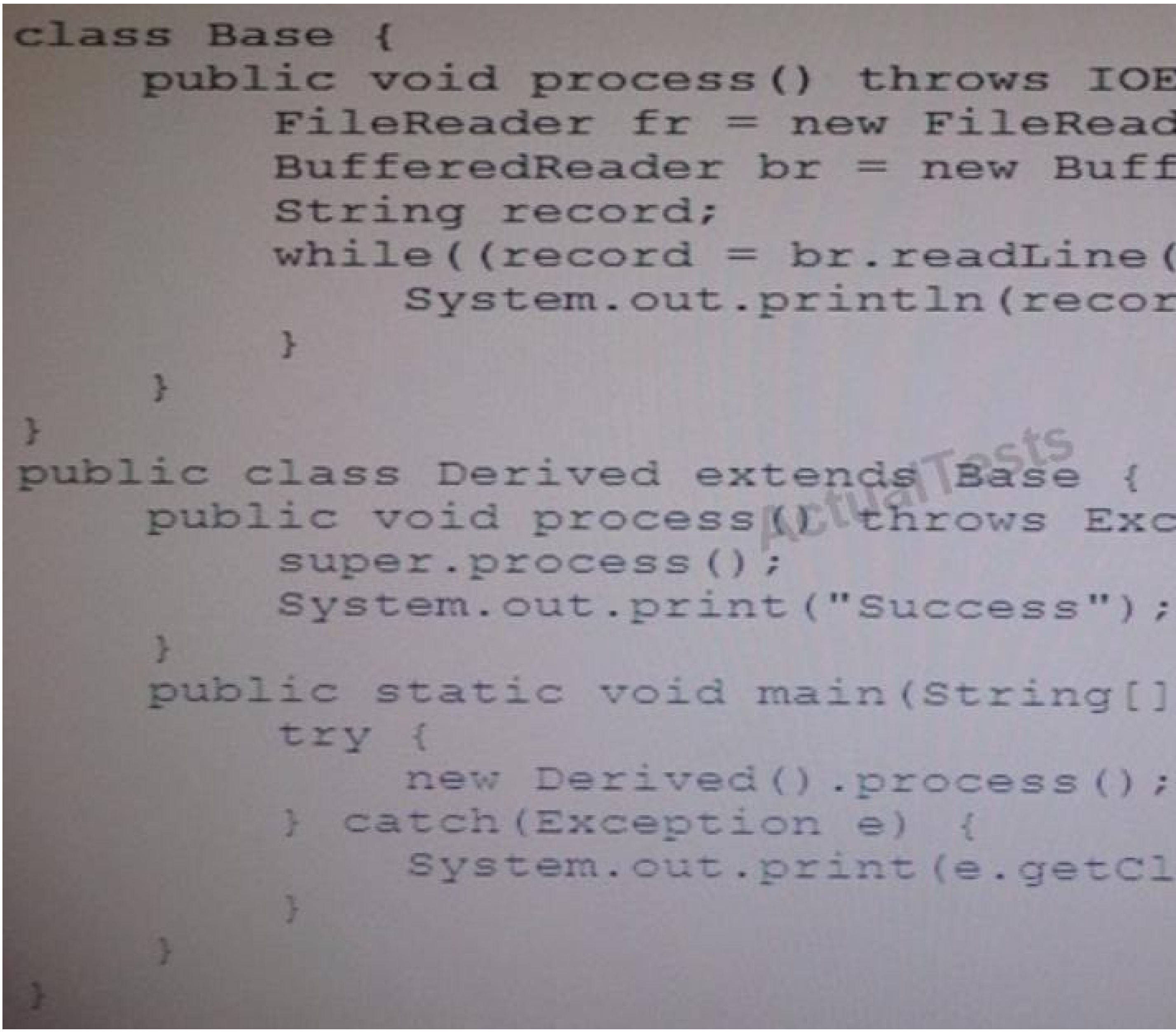
Explanation/Reference:
Explanation: getInstance
public static finalNumberFormatgetInstance()
Returns the default number format for the current default locale. The default format is one of the styles provided by the other factory methods: getNumberInstance(E), getIntegerInstance(D), getCurrencyInstance(F)or getPercentInstance. Exactly which one is locale dependant.

C: To obtain a NumberFormat for a specific locale, including the default locale, call one of NumberFormat's factory methods, such as getInstance().

E:To obtain standard formats for a given locale, use the factory methods on NumberFormat such as getNumberInstance. These factories will return the most appropriate sub-class of NumberFormat for a given locale.
F:To obtain standard formats for a given locale, use the factory methods on NumberFormat such as getInstance or getCurrencyInstance.

Reference:java.textClass NumberFormat

QUESTION 116
Given the code fragment:



```
class Base {
    public void process() throws IOE
        FileReader fr = new FileRead
        BufferedReader br = new Buff
        String record;
        while((record = br.readLine(
            System.out.println(recor
        }
    }
}
public class Derived extends Base {
    public void process() throws Exc
        super.process();
        System.out.print("Success");
    }
    public static void main(String[]
        try {
            new Derived().process();
        } catch(Exception e) {
            System.out.print(e.getCl
        }
    }
}
```

If the file userguide.txt does not exist, what is the result?

A. An empty file is created and success is printed.
B. class java.io.FileNotFoundException.
C. class java.io.IOException.
D. class java.lang.Exception.
E. Compilation fails.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: Note:public FileReader(String fileName)
throws FileNotFoundException
Creates a new FileReader, given the name of the file to read from.

Parameters:
fileName - the name of the file to read from
Throws:
FileNotFoundException - if the named file does not exist, is a directory rather than a regular file, or for some other reason cannot be opened for reading.

Reference:java.io Class FileReader

QUESTION 117
Given:

```
public class Counter {

public static int getCount(String[] arr) {

int count =0 ;

for(String var:arr) {

if(var!=null) count++;

}

return count;

}

public static void main(String[] args) {

String[] arr =new String[4];

arr[1] = "C";

arr[2] = "";

arr[3] = "Java";

assert (getCount(arr) < arr.length);

System.out.print(getCount(arr));

}

}
```

And the commands:

```
javac Counter.java

java ea Counter
```

What is the result?

A. 2
B. 3
C. NullPointException is thrown at runtime
D. AssertionError is thrown at runtime
E. Compilation fails

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The command line javac Counter.java
will compile the code.
The command line java ea Counter
will run the cod with assertions enabled.

The following line:
assert (getCount(arr) < arr.length);
where the Boolean expressiongetCount(arr) < arr.lengthwill evaluate to false, will ensure that anAssertionError is thrown at runtime.

Note:The javac command compiles Java source code into Java bytecodes. You then use the Java interpreter - the java command - to interprete the Java bytecodes.

Note 2:The java tool launches a Java application. It does this by starting a Java runtime environment, loading a specified class, and invoking that class's main method. The method declaration must look like the following:
public static void main(String args[])

Paramater ea:
-enableassertions[:<package name>"..." | :<class name>] -ea[:<package name>"..." | :<class name>]
Enable assertions. Assertions are disabled by default. With no arguments, enableassertions or -ea enables assertions.

Note 3:
An assertion is a statement in the JavaTM programming language that enables you to test your assumptions about your program.
Each assertion contains a boolean expression that you believe will be true when the assertion executes. If it is not true, the system will throw an error.

QUESTION 118
Given the code fragment:

```
String s = "Java 7, Java 6";

Pattern p = Pattern.compile("Java.+\\d");

Matcher m = p.matcher(s);

while (m.find()) {

System.out.println(m.group());

}
```

What is the result?

A. Java 7
B. Java 6
C. Java 7, Java 6
D. Java 7
java 6
E. Java

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 119
Given:

```
import java.util.concurrent.atomic.AtomicInteger;

public class AtomicCounter {

private AtomicInteger c = new AtomicInteger(0);

public void increment() {

// insert code here

}

}
```

Which line of code, inserted inside the increment () method, will increment the value of c?

A. c.addAndGet();
B. c++;
C. c = c+1;
D. c.getAndIncrement ();

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: getAndIncrement
public final int getAndIncrement()
Atomically increment by one the current value.

Reference:java.util.concurrent.atomic

QUESTION 120
Given:

```
public class Runner {

public static String name = "unknown";

public void start() {

System.out.println(name);

}
public static void main(String[] args) {

name = "Daniel";

start();

}

}
```

What is the result?

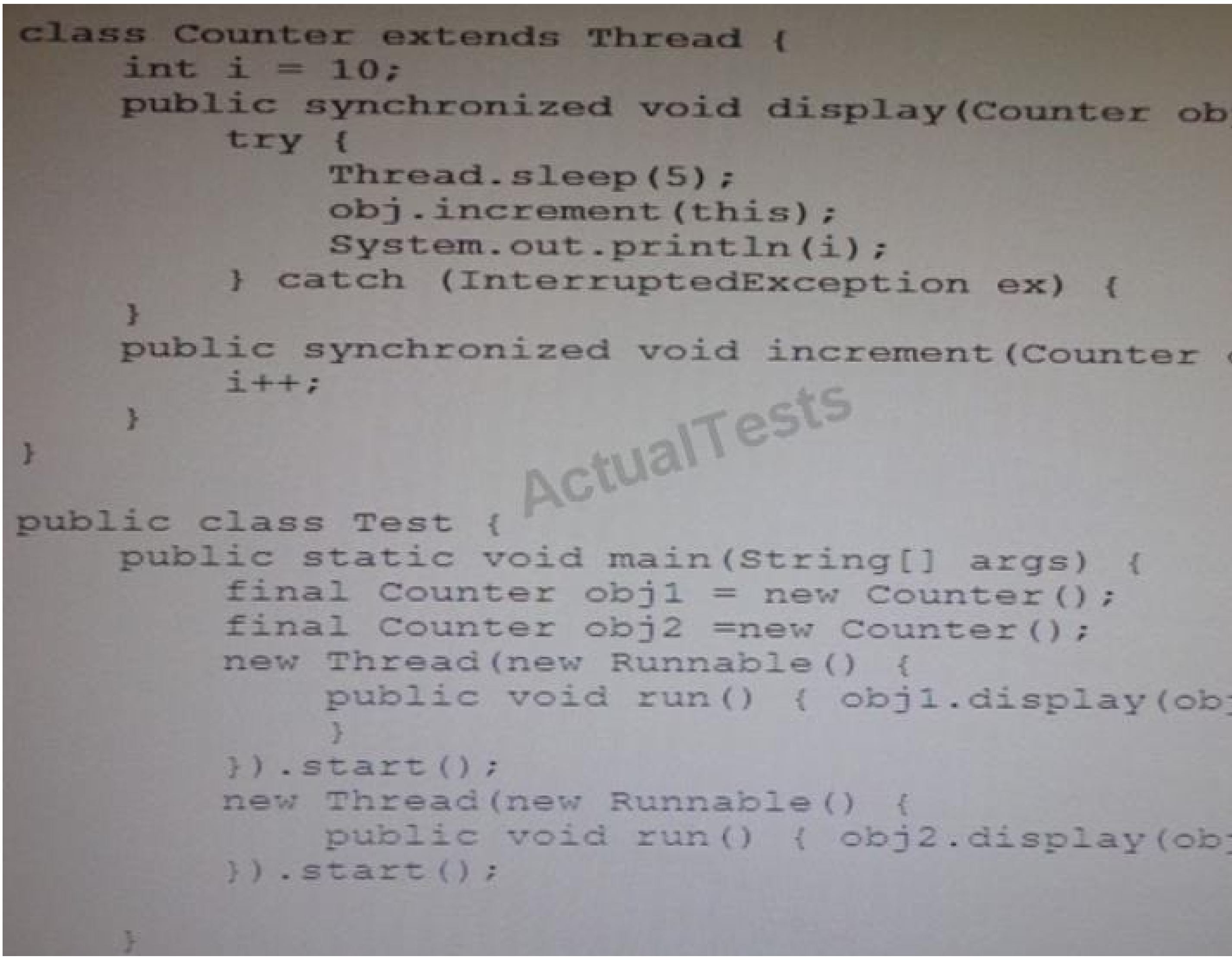
A. Daniel
B. Unknown
C. It may print"unknown"or"Daniel"depending on the JVM implementation.
D. Compilation fails.
E. An exception is thrown at runtime.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Thecompilation fails at line start();

Exception in thread "main" java.lang.RuntimeException: Uncompilable source code - non-static method start() cannot be referenced from a static context

QUESTION 121
Given:



```
class Counter extends Thread {
    int i = 10;
    public synchronized void display(Counter ob
        try {
            Thread.sleep(5);
            obj.increment(this);
            System.out.println(i);
        } catch (InterruptedException ex) {
    }
    public synchronized void increment(Counter
        i++;
    }
}

public class Test {
    public static void main(String[] args) {
        final Counter obj1 = new Counter();
        final Counter obj2 =new Counter();
        new Thread(new Runnable() {
            public void run() { obj1.display(ob
            }
        }).start();
        new Thread(new Runnable() {
            public void run() { obj2.display(ob
        }).start();

    }
```

From what threading problem does the program suffer?

A. deadlock
B. livelock
C. starvation
D. race condition

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: A thread often acts in response to the action of another thread. If the other thread's action is also a response to the action of another thread, thenlivelock may result. As with deadlock, livelocked threads are unable to make further progress. However, the threads are not blocked -- they are simply too busy responding to each other to resume work. This is comparable to two people attempting to pass each other in a corridor: Alphonse moves to his left to let Gaston pass, while Gaston moves to his right to let Alphonse pass. Seeing that they are still blocking each other, Alphone moves to his right, while Gaston moves to his left. They're still blocking each other, so...

QUESTION 122
Given the code fragment:

public class IsContentSame {

```
public static boolean isContentSame() throws IOException {

Path p1=Paths.get("D:\\faculty\\report.txt");

Path p2=Paths.get("C:\\student\\report.txt");

Files.copy(p1,p2,StandardCopyOption.REPLACE_EXISTING,StandardCopyOption.COPY_ATTRI
BUTES,LinkOption.NOFOLLOW_LINKS);

if(Files.isSameFile(p1,p2)) {

return true;

} else {

return false;

}

}

public static void main(String[] args) {
try {

boolean flag = isContentSame();

if(flag)

System.out.println("Equal");

else

System.out.println("Not equal");

} catch (IOException e) {

System.err.println("Caught IOException: " + e.getMessage());

}

}

}
```

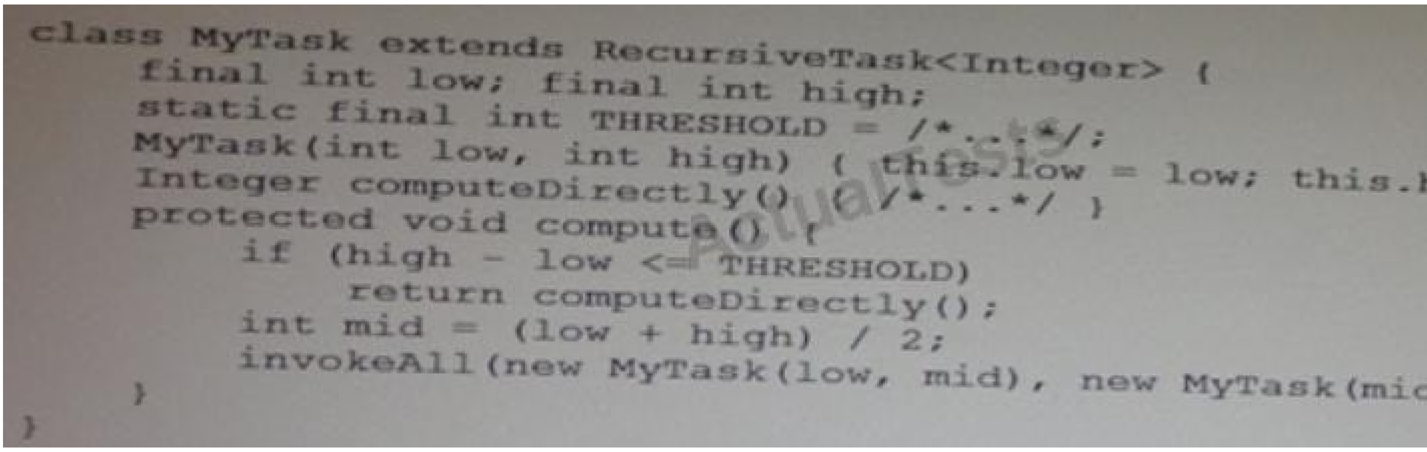
What is the result when the result.txt file already exists in c:\student?

A. The program replaces the file contents and the file's attributes and prints Equal.
B. The program replaces the file contents as well as the file attributes and prints Not equal.
C. An unsupportedoperationException is thrown at runtime.
D. The program replaces only the file attributes and prints Not equal.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: Assuming there is a fileD:\\faculty\\report.txtthen this file will be copied and will be replacing C:\\student\\report.txt.

QUESTION 123
Given the following incorrect program:



```
class MyTask extends RecursiveTask<Integer> {
    final int low; final int high;
    static final int THRESHOLD = /*...*/;
    MyTask(int low, int high) { this.low = low; this.
    Integer computeDirectly() { /*...*/ }
    protected void compute() {
        if (high - low <= THRESHOLD)
            return computeDirectly();
        int mid = (low + high) / 2;
        invokeAll(new MyTask(low, mid), new MyTask(mi
    }
}
```

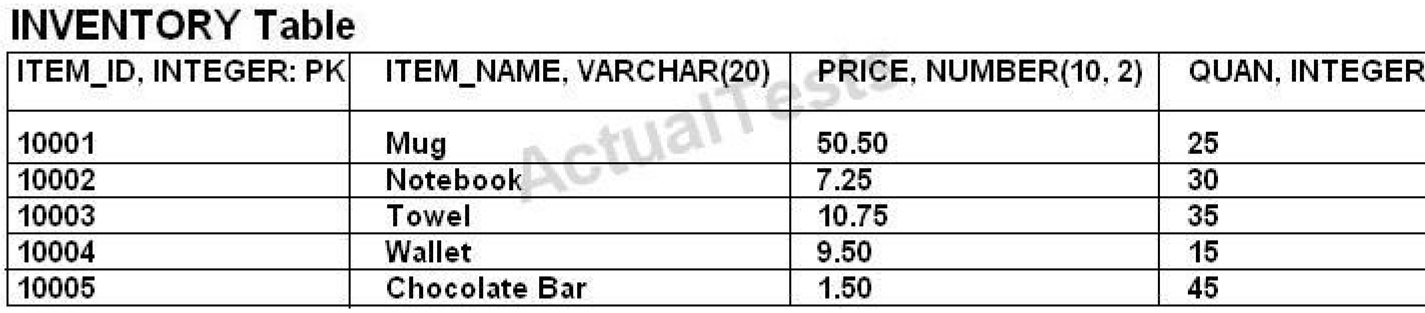
Which two changes make the program work correctly?

A. Results must be retrieved from the newly created MyTask instances and combined.
B. The threshold value must be increased so that the overhead of task creation does not dominate the cost of computation.
C. The midpoint computation must be altered so that it splits the workload in an optimal manner.
D. The compute () method must be changed to return an Integer result.
E. The compute () method must be enhanced to (fork) newly created tasks.
F. The myTask class must be modified to extend RecursiveAction instead of RecursiveTask

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Incorrect answer:
F: ARecursiveActionis arecursive resultless ForkJoinTask.Here we need a result, so we should use aRecursiveTask.

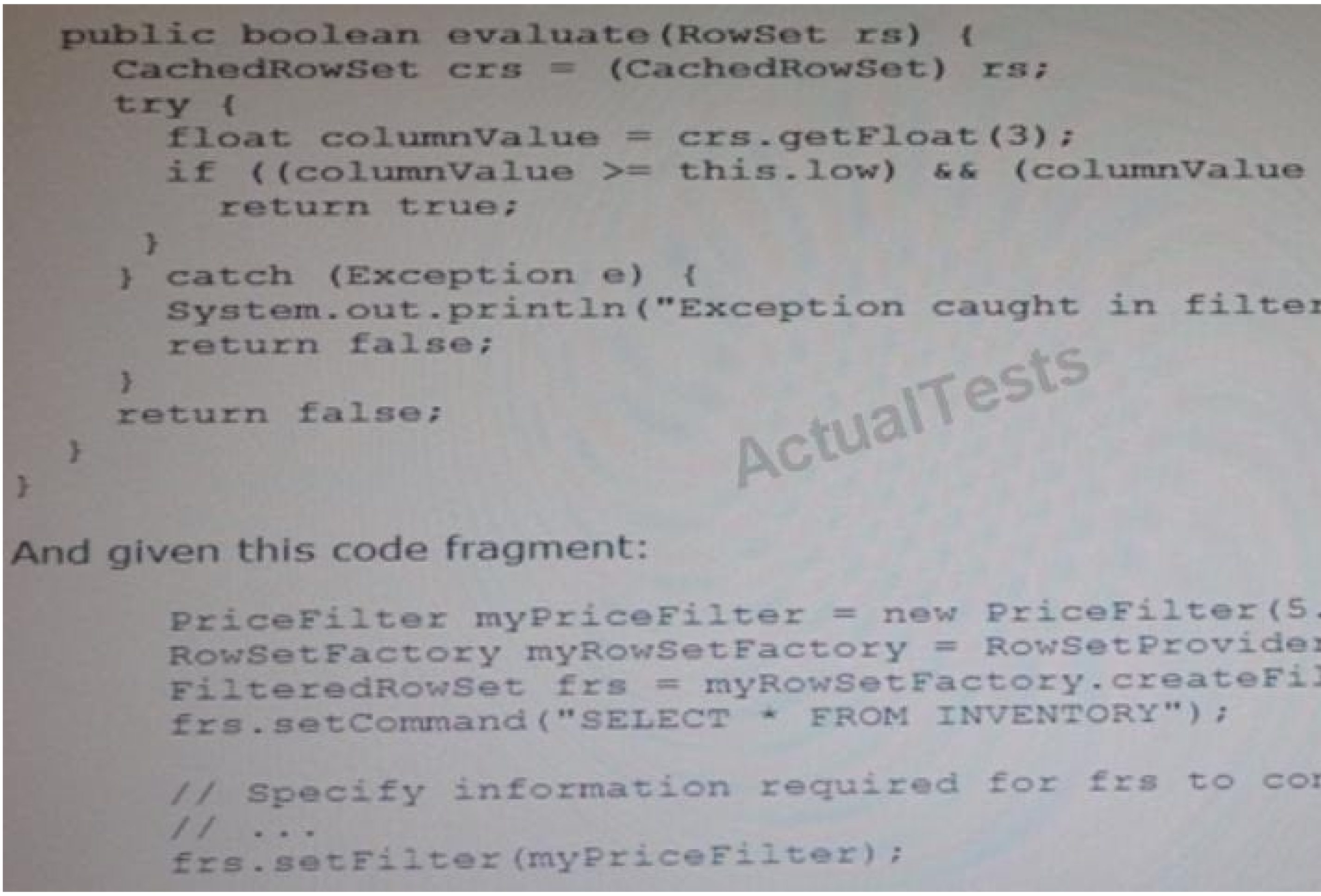
QUESTION 124
Given the database table:



INVENTORY Table

ITEM_ID, INTEGER: PK	ITEM_NAME, VARCHAR(20)	PRICE, NUMBER(10, 2)	QUAN, INTEGER
10001	Mug	50.50	25
10002	Notebook	7.25	30
10003	Towel	10.75	35
10004	Wallet	9.50	15
10005	Chocolate Bar	1.50	45

And given this class:



```
  public boolean evaluate(RowSet rs) {
     CachedRowSet crs = (CachedRowSet) rs;
     try {
        float columnValue = crs.getFloat(3);
        if ((columnValue >= this.low) && (columnValue
           return true;
       }
     } catch (Exception e) {
        System.out.println("Exception caught in filte
        return false;
     }
     return false;
  }
}
```

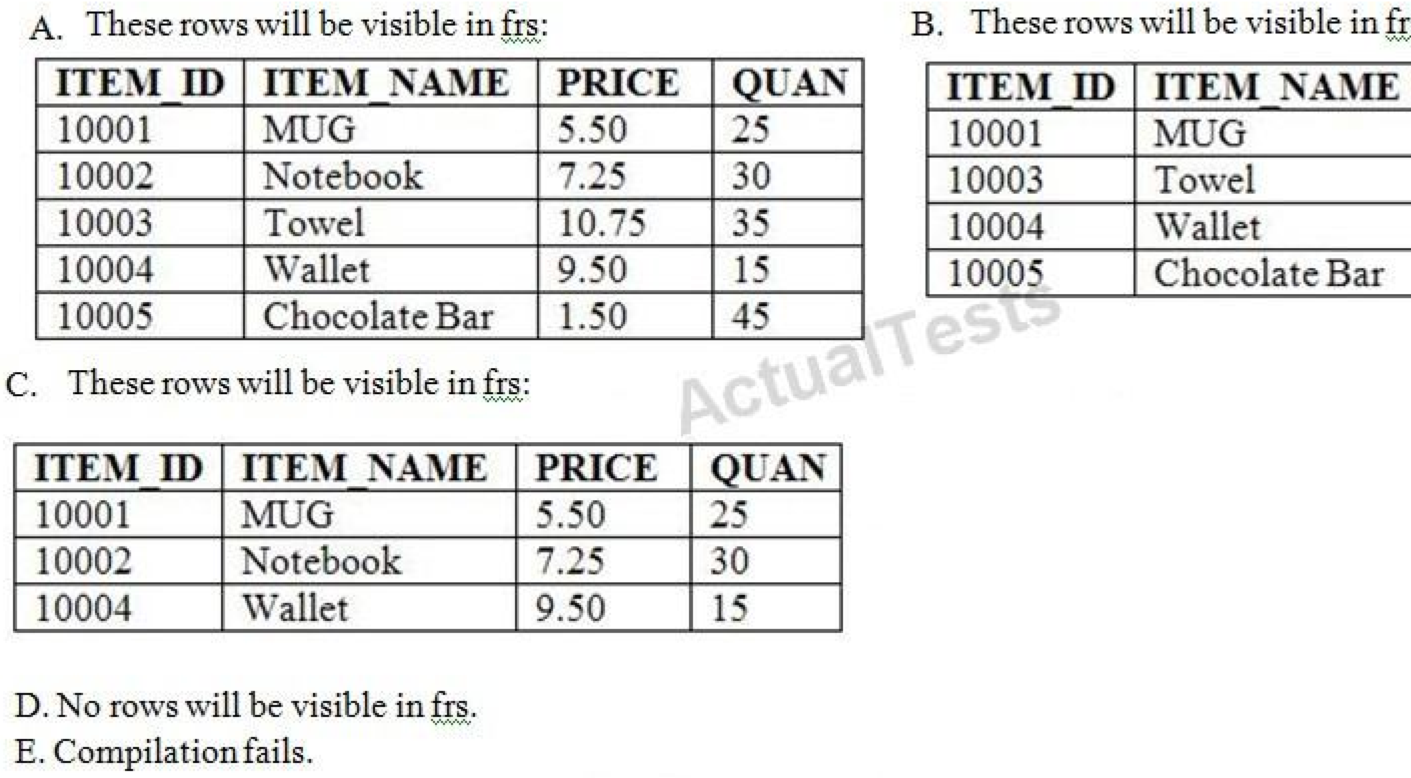
And given this code fragment:

```
        PriceFilter myPriceFilter = new PriceFilter(5.
        RowSetFactory myRowSetFactory = RowSetProvide
        FilteredRowSet frs = myRowSetFactory.createFi
        frs.setCommand("SELECT * FROM INVENTORY");

        // Specify information required for frs to co
        // ...
        frs.setFilter(myPriceFilter);
```

Assume that the SQL integer queries are valid. What is the result of compiling and executing this code fragment?

Refer to the exhibit.



A. These rows will be visible in frs:

ITEM_ID	ITEM_NAME	PRICE	QUAN
10001	MUG	5.50	25
10002	Notebook	7.25	30
10003	Towel	10.75	35
10004	Wallet	9.50	15
10005	Chocolate Bar	1.50	45

B. These rows will be visible in fr

ITEM_ID	ITEM_NAME
10001	MUG
10003	Towel
10004	Wallet
10005	Chocolate Bar

C. These rows will be visible in frs:

ITEM_ID	ITEM_NAME	PRICE	QUAN
10001	MUG	5.50	25
10002	Notebook	7.25	30
10004	Wallet	9.50	15

D. No rows will be visible in frs.

E. Compilation fails.

A. A
B. B
C. C
D. D
E. E

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code will compile now.
The three rows with PRICE in between 5.5 and 9.5 will be displayed.

QUESTION 125
Given that myFile.txt contains:

First

Second

Third

And given:

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class ReadFile04 {

```
public static void main(String[] args) {
try (BufferedReader buffIn =
new BufferedReader(new FileReader("D:\\faculty\\myfile.txt"))) {
String line = "";
int count = 1;
buffIn.mark(1);
line = buffIn.readLine();
System.out.println(count + ": " + line);
line = buffIn.readLine();
count++;
System.out.println(count + ": " + line);
buffIn.reset();
line = buffIn.readLine();
count++;
System.out.println(count + ": " + line);
} catch (IOException e) {
System.out.println("IOException");
}
}
}
```

What is the result?

A. 1: First
2: Second
3: Third

B. 1 : First
2: Second
3: First

C. 1: First
2: First
3: First

D. IOExcepton

E. Compilation fails

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 126

Given:

```
public class Print01 {

public static void main(String[] args) {

double price = 24.99;

int quantity = 2;

String color = "Blue";

// insert code here. Line ***

}

}
```

Which two statements, inserted independently at line ***, enable the program to produce the following output:

```
We have 002 Blue pants that cost $24.99.
```

A. System.out.printf("We have %03d %s pants that cost $%3.2f.\n",quantity, color, price);
B. System.out.printf("We have$03d$s pants that cost $$3.2f.\n",quantity, color, price);
C. String out = String.format ("We have %03d %s pants that cost $%3.2f.\n",quantity, color, price);
   System.out.println(out);
D. String out = System.out.format("We have %03d %s pants that cost $%3.2f.",quantity, color, price);
   System.out.println(out);
E. System.out.format("We have %s%spants that cost $%s.\n",quantity, color, price);

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 127
Given the cache class:

```
public class Cache {

private T t;

public void setValue (T t) { this.t=t; }

public T getValue() {return t; }

}
```

What is the result of the following?

```
Cache<> c = new Cache<Integer>(); // Line 1
```

A. SetValue(100); // Line 2
   System.out.print(c.getValue().intValue() +1); // Line 3
B. 101
C. Compilation fails at line 1.
D. Compilation fails at line 2.

E. Compilation fails at line 3.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: Compilation failure at line:
illegal start of type
type cache.Cache does not take parameters.

QUESTION 128
Given the code fragment:

```
public class Base {
BufferedReader br;

String record;

public void process() throws FileNotFoundException {

br = new BufferedReader(new FileReader("manual.txt"));

}

}

public class Derived extends Base {

// insert code here. Line ***

public static void main(String[] args) {

try {

new Derived().process();

} catch (Exception e) { }

}

}
```

Which code fragment inserted at line ***, enables the code to compile?

A. public void process () throws FileNotFoundException, IOException { super.process ();
```
while ((record = br.readLine()) !=null) {
System.out.println(record);
}
}
```

B. public void process () throws IOException {
```
super.process ();
while ((record = br.readLine()) != null) {
System.out.println(record);
}
}
```

C. public void process () throws Exception {
```
super.process ();
while ((record = br.readLine()) !=null) {
System.out.println(record);
}
}
```

D. public void process (){
```
try {
super.process ();
while ((record = br.readLine()) !=null) {
System.out.println(record);
}
} catch (IOException | FileNotFoundException e) { }
}
```
E. public void process (){
```
try {
super.process ();
while ((record = br.readLine()) !=null) {
System.out.println(record);
}
} catch (IOException e) {}
}
```

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Incorrect answer:
D: exception java.io.FileNotFoundException has already been caught

Alternatives in a multi-catch statement cannot be related to subclassing Alternative java.io.FileNotFoundExceptionis a subclass of alternative java.io.IOException

QUESTION 129
An application is waiting for notification of changes to a tmp directory using the following code statements:

```
Path dir = Paths.get("tmp")

WatchKey key = dir.register (watcher, ENTRY_CREATE, ENTRY_DELETE, ENTRY_MODIFY) ;
```

In the tmp directory, the user renames the file testA to testB,

Which statement is true?

A. The events received and the order of events are consistent across all platforms.
B. The events received and the order of events are consistent across all Microsoft Windows versions.
C. The events received and the order of events are consistent across all UNIX platforms.
D. The events received and the order of events are platform dependent.

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: Most file system implementations have native support for file change notification. The Watch Service API takes advantage of this support where available. However, when a file system does not support this mechanism, the Watch Service will poll the file system, waiting for events.

Note:WatchKey : When a Watchable entity is registered with a WatchService a key which is a WatchKey is generated. Initially the key is in ready state waiting to be notified of any events on the Watchable entity. Once an event occurs the key goes into signaled state and allows to access the events using its pollEvents method. After processing the poll events the key has to be reset by invoking its reset method.

Reference: The Java Tutorials,Watching a Directory for Changes

QUESTION 130

Given:

```
StringBuffer b = new StringBuffer("3");

System.out.print(5+4+b+2+1);
```

What is the result?

A. 54321
B. 9321
C. 5433
D. 933
E. Output is Similar to: 9java.lang.StringBuffer@100490121.
F. Compilation fails.

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code will not compile.
The print function cannot handle the mixture of integers and strings.

```
Exception in thread "main" java.lang.RuntimeException: Uncompilable source code - Erroneous tree type
```

QUESTION 131

Which four are true about enums?

A. An enum is typesafe.
B. An enum cannot have public methods or fields.
C. An enum can declare a private constructor.
D. All enums implicitly implement Comparable.
E. An enumcan subclass another enum.
F. An enum can implement an interface.

Correct Answer: ACDF
Section: (none)
Explanation

Explanation/Reference:
Explanation: C:The constructor for an enum type must be package-private or private access.

Reference: Java Tutorials,Enum Types

QUESTION 132

Given:

```
abstract class Boat {

String doFloat() { return "floating"; }

abstract void doDock();

}

class Sailboat extends Boat {

public static void main(String[] args) {
```

```
Boat b = new Sailboat(); // Line A

Boat b2 = new Boat(); // Line B

}

String doFloat() { return "slow float"; } // Line C

void doDock() { } // Line D

}
```

Which two are true about the lines labeled A through D?

A. The code compiles and runs as is.
B. If only line A is removed, the code will compile and run.
C. If only line B is removed, the code will compile and run.
D. If only line D is removed, the code will compile and run.
E. Line C is optional to allow the code to compile and run.
F. Line C is mandatory to allow the code to compile andrun.

Correct Answer: AE
Section: (none)
Explanation

Explanation/Reference:
Explanation: A: The code will compile. The abstract method doDock() is implemented fine, and doFloat() is overridden.
E: Line C overrides the implementation of doFloat(). This is optional.

QUESTION 133
Given the code fragment:

```
try {

String query = "SELECT * FROM Employee WHERE ID=110";

Statement stmt = conn.createStatement();

ResultSet rs = stmt.executeQuery(query);

System.out.println("Employee ID: " + rs.getInt("ID"));

} catch (Exception se) {

System.out.println("Error");

}
```

Assume that the SQL query matches one record. What is the result of compiling and executing this code?

A. The code prints Error.
B. The code prints the employee ID.
C. Compilation fails due to an error at line 13.
D. Compilation fails due to an error at line 14.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code compiles fine.
The code will rune fine.

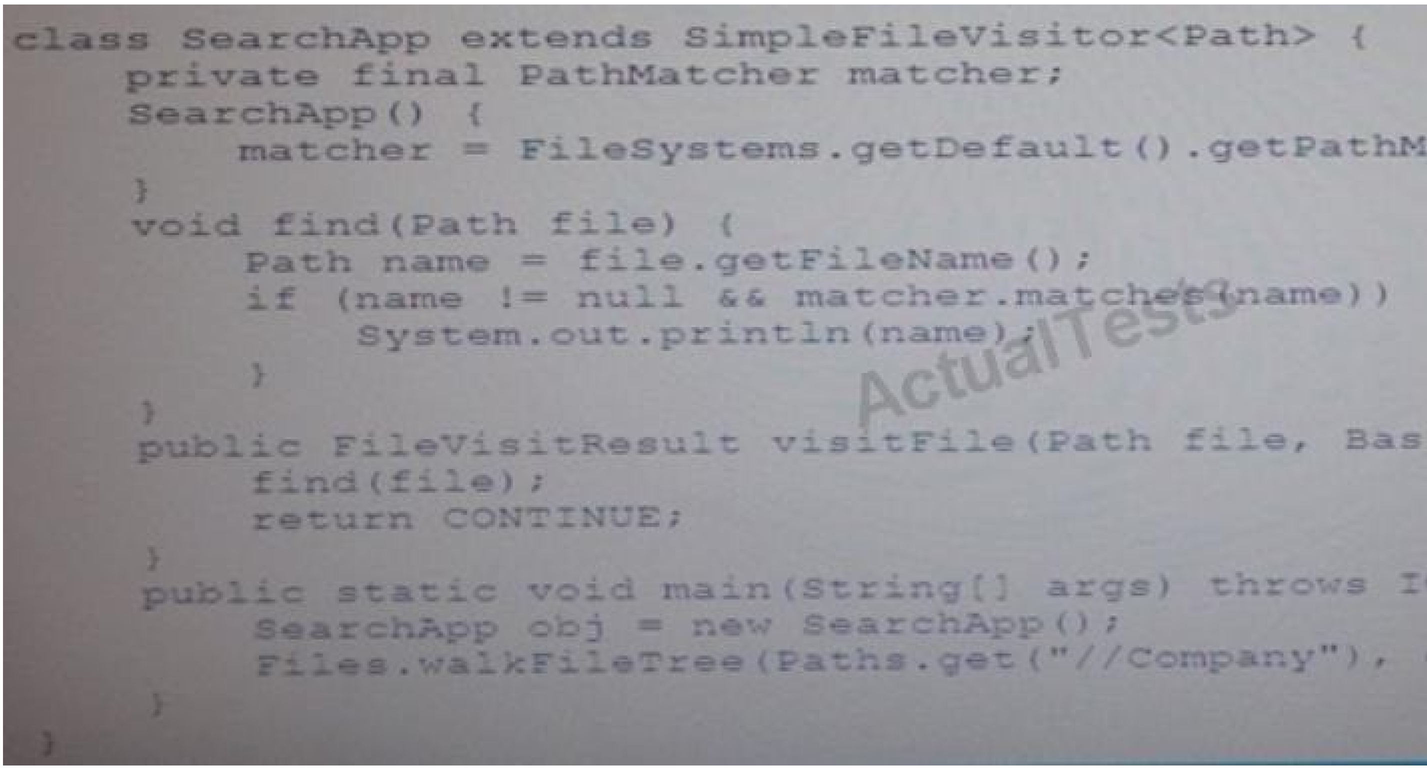
public int getInt(String columnName)
throws SQLException
Retrieves the value of the designated column in the current row of this ResultSet object as an int in the Java programming language

QUESTION 134
Given the directory structure that contains three directories: company, Salesdat, and Finance:

Company

- Salesdat
- Target.dat
- Finance
- Salary.dat
- Annual.dat

And the code fragment:



```
class SearchApp extends SimpleFileVisitor<Path> {
    private final PathMatcher matcher;
    SearchApp() {
        matcher = FileSystems.getDefault().getPathM
    }
    void find(Path file) {
        Path name = file.getFileName();
        if (name != null && matcher.matches(name))
            System.out.println(name);
        }
    }
    public FileVisitResult visitFile(Path file, Bas
        find(file);
        return CONTINUE;
    }
    public static void main(String[] args) throws I
        SearchApp obj = new SearchApp();
        Files.walkFileTree(Paths.get("//Company"),
    }
}
```

If Company is the current directory, what is the result?

A. Prints only Annual.dat
B. Prints only Salesdat, Annual.dat
C. Prints only Annual.dat, Salary.dat, Target.dat
D. Prints at least Salesdat, Annual.dat, Salary.dat, Target.dat

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The pattern *dat will matchthe directory name Salesdat and it will also match the file Annual.dat.

It will not be matched to Target.dat which is in a subdirectory.

QUESTION 135
Given:

import java.util.Scanner;

public class Painting {

public static void main(String[] args) {

String input = "Pastel, *Enamel, Fresco, *Gouache";

Scanner s = new Scanner(input);

A. useDelimiter(",\\s*");
while (s.hasNext()) {
System.out.println(s.next());
}
}
}
What is the result?

B. Paste1
Ename1
Fresco
Gouache

C. Paste1
*Ename1
Fresco
*Gouache

D. Pastel
Ename1
Fresco
Gouache

E. Pastel
Ename1, Fresco
Gouache

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 136
Which type of ExecutorService supports the execution of tasks after a fixed delay?

A. DelayedExecutorService
B. ScheduledExecutorService
C. TimedExecutorService
D. FixedExecutorService
E. FutureExecutorService

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation: The ScheduledExecutorService interface supplements the methods of its parent

ExecutorService with schedule, which executes a Runnable or Callable task after a specified delay. In addition, the interface defines scheduleAtFixedRate and scheduleWithFixedDelay, which executes specified tasks repeatedly, at defined intervals.

Note:The java.util.concurrent package defines three executor interfaces:

*Executor, a simple interface that supports launching new tasks. *ExecutorService, a subinterface of Executor, which adds features that help manage the lifecycle, both of the individual tasks and of the executor itself. *ScheduledExecutorService, a subinterface of ExecutorService, supports future and/or periodic execution of tasks.

Reference: The Java Tutorials,Executor Interfaces

QUESTION 137
How many Threads are created when passing task to an Executor instance?

A. A new Thread is used for each task.
B. A number of Threads equal to the number of CPUs Is used to execute tasks.
C. A single Thread Is used to execute all tasks.
D. A developer-defined number of Threads is used to execute tasks.
E. A number of Threads determined by system load is used to execute tasks.
F. The method used to obtain the Executor determines how many Threads are used to execute tasks.

Correct Answer: F
Section: (none)
Explanation

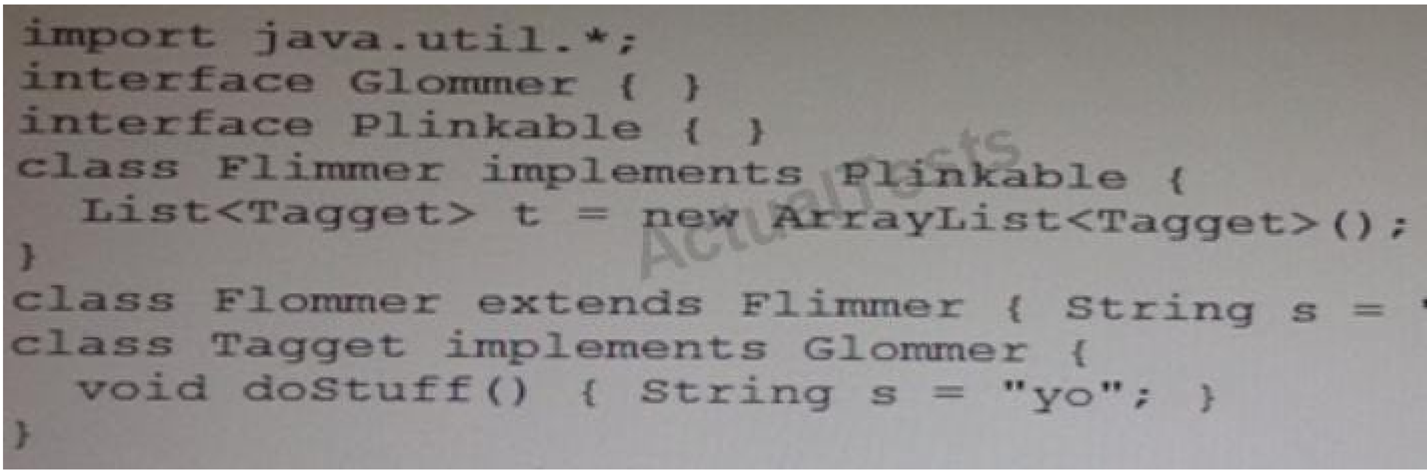
Explanation/Reference:
Explanation: The Executor interface provides a single method, execute, designed to be a drop-in replacement for a common thread-creation idiom. If r is a Runnable object, and e is an Executor object you can replace

(new Thread(r)).start();
with

e.execute(r);
However, the definition of execute is less specific. The low-level idiom creates a new thread and launches it immediately. Depending on the Executor implementation, execute may do the same thing, but is more likely to use an existing worker thread to run r, or to place r in a queue to wait for a worker thread to become available.

Reference: The Java Tutorial,The Executor Interface

QUESTION 138
Given:



```
import java.util.*;
interface Glommer { }
interface Plinkable { }
class Flimmer implements Plinkable {
  List<Tagget> t = new ArrayList<Tagget>();
}
class Flommer extends Flimmer { String s = "
class Tagget implements Glommer {
  void doStuff() { String s = "yo"; }
}
```

Which two statements concerning the OO concepts "is-a" and "has-a" are true?

A. Flimmer is-a Glommer.
B. Flommer has-a String.
C. Taggethas-a Glommer.
D. Flimmer is-a ArrayList.
E. Taggethas-a doStuff()
F. Taggetis-a Glommer.

Correct Answer: BF
Section: (none)
Explanation

Explanation/Reference:
Explanation: B: The relationship modeled by composition is often referred to as the "has-a" relationship. Here Flommer has-a String.
E: Thehas-a relationship has anencapsulation feature (like private or protected modifier used before each member field or method).
Here Tagget has-a method doStuff()
F: TaggetimplementsGlommer.
Taggetis-aGlommer.

Note: Thehas-a relationship has anencapsulation feature (like private or protected modifier used before each member field or method).

QUESTION 139
Given the Greetings.properties file, containing:

```
HELLO_MSG = Hello, everyone!

GOODBYE_MSG = Goodbye everyone!
```

And given:

```
import java.util.Enumeration;

import java.util.Locale;

import java.util.ResourceBundle;

public class ResourceApp {

public void loadResourceBundle() {

ResourceBundle resource = ResourceBundle.getBundle("Greetings", Locale.US);

System.out.println(resource.getObject(1));

}
public static void main(String[] args) {

new ResourcesApp().loadResourceBundle();

}

}
```

What is the result?

A. Compilation fails

B. HELLO_MSG
C. GOODGYE_NSG
D. Hello, everyone!
E. Goodbye everyone!

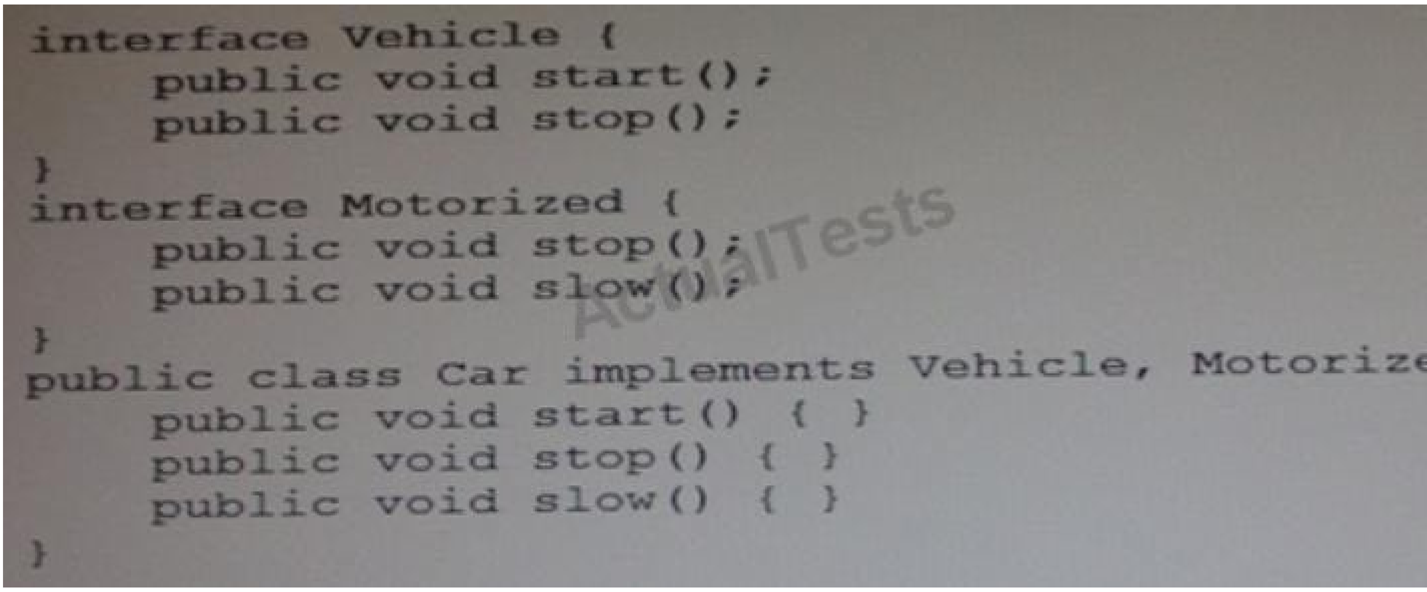
Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code will not compile.
The problem is the following line:
System.out.println(resource.getObject(1));

In particular getObject(1) throws the following error:
Exception in thread "main" java.lang.RuntimeException: Uncompilable source code - Erroneous sym type: <any>.loadResourceBundle

Note:getObject(String key)
Gets an object for the given key from this resource bundle or one of its parents.

QUESTION 140
Given:



```
interface Vehicle {
    public void start();
    public void stop();
}
interface Motorized {
    public void stop();
    public void slow();
}
public class Car implements Vehicle, Motorize
    public void start() { }
    public void stop() { }
    public void slow() { }
}
```

What is the result of invoking Car's scop method?

A. Both vehicles and Motorized's stop methods are invoked.
B. Vehicles stop method is invoked.
C. Motorized's stop method is invoked-
D. The implementation of the Car's stop determines the behavior.
E. Compilation fails.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The Car class is implementing the methods.
Methods are not implemented in interfaces.

QUESTION 141

Given:

```
interface Rideable {

public String ride() { return "riding "; }

}

class Horse implements Rideable {

public String ride() { return "cantering "; }

}

class Icelandic extends Horse implements Rideable {

public String ride() { return "tolting "; }

}

class Test {

public static void main(String[] args) {

Rideable r1 = new Icelandic();

Rideable r2 = new Horse();

Horse h1 = new Icelandic();

System.out.println(r1.ride() + r2.ride() + h1.ride());
}

}
```

What is the result?

A. riding riding tolting
B. riding riding cantering
C. tolting cantering tolting
D. tolting cantering cantering
E. Compilation fails.
F. An exception is thrown at runtime.

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation: The compilation fails at:

```
interface Rideable {
public String ride() { return "riding ";}
}
```

Error due to: interface methods cannot have body.

QUESTION 142
You have been asked to create a ResourceBundle file to localize an application.

Which code example specifies valid keys menu1 and manu2 with values of File Menu and View Menu?

A. <key name ="menu1">File Menu</key>
<key name ="menu1">View Menu</key>

B. <key> menu1</key><File Menu>File Menu </value>
<key> menu1</key><File Menu>View Menu </value>

C. menu1m File menu, menu2, view menu

D. menu1 = File Menu
menu2 = View Menu

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: A properties file is a simple text file. You can create and maintain a properties file with just about any text editor.

You should always create a default properties file. The name of this file begins with the base name of your ResourceBundle and ends with the .properties suffix. In the PropertiesDemo program the base name is LabelsBundle. Therefore the default properties file is called LabelsBundle.properties. The following examplefile contains the following lines:

```
# This is the default LabelsBundle.properties file
s1 = computer
s2 = disk
s3 = monitor
s4 = keyboard
```

Note that in the preceding file the comment lines begin with a pound sign (#). The other lines contain key-value pairs. The key is on the left side of the equal sign and the value is on the right. For instance, s2 is the key that corresponds to the value disk. The key is arbitrary. We could have called s2 something else, like msg5 or diskID. Once defined, however, the key should not change because it is referenced in the source code. The values may be changed. In fact, when your localizers create new properties files to accommodate additional languages, they will translate the values into various languages.

QUESTION 143
Given:

```
interface Rideable {

String ride() ;

}

class Horse implements Rideable {

String ride() { return "cantering "; }

}

class Icelandic extends Horse {

String ride() { return "tolting "; }

}

class Test {
public static void main(String[] args) {

Rideable r1 = new Icelandic();
Rideable r2 = new Horse();

Horse h1 = new Icelandic();

System.out.println(r1.ride() + r2.ride() + h1.ride());

}

}
```

What is the result?

A. tolting cantering cantering
B. cantering cantering cantering
C. compilation fails
D. an exception is thrown at runtime

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 144
Given:

```
public class SleepOtherThread {

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

Runnable r = new Runnable() {

public void run() {

System.out.print(Thread.currentThread().getName());

}

};

Thread t1 = new Thread(r, "One ");

t1.start();

t1.sleep(2000);
Thread t2 = new Thread(r, "Two ");

t2.start();

t2.sleep(1000);

System.out.print("Main ");

}

}
```

What is the most likely result?

A. Main One Two

B. Main Two One
C. One Two Main
D. One Main Two
E. Two Main One

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 145
Which two are valid initialization statements?

A. Map<String, String> m = newSortedMap<String, String>();
B. Collection m = new TreeMap<Object, Object>();
C. HashMap<Object, Object> m = newSortedMap<Object, Object>();
D. SortedMap<Object, Object> m = new TreeMap<Object, Object> ();
E. Hashtablem= new HashMap();
F. Map<List, ArrayList> m = new Hashtable<List, ArrayList>();

Correct Answer: DF
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 146
Which four are syntactically correct?

A. package abc;
package def;
import Java.util . * ;
public class Test { }
B. package abc;
import Java.util.*;
import Java.util.regex.* ;
public class Test { }
C. package abc;
public class Test {}
import Java.util.* ;
D. import Java.util.*;
package abc;
public class Test {}
E. package abc;
import java.util. *;
publicclassTest{}
F. publicclass Test{}
package abc;
importjava.util.*{}
G. import java.util.*;
public class Test{}
H. package abc;
public class test

Correct Answer: BEGH

Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 147
Given the code fragment:

```
public class Test {

public static void main(String[] args) {

Path dir = Paths.get("D:\\company");

//insert code here. Line ***

for (Path entry: stream) {

System.out.println(entry.getFileName());
} catch (IOException e) {

System.err.println("Caught IOException: " + e.getMessage());

}

}
```

Which two try statements, when inserted at line ***, enable you to print files with the extensions.java, .htm, and .jar.

A. try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir,"*.{java, htm,jar}")){
B. try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir,"*. [java, htm, jar]")) {
C. try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir,"*.{java*, htm*, jar*}")) {
D. try (DirectoryStream<Path> stream = Files.newDirectoryStream(dir,"**.{java, htm, jar}")) {

Correct Answer: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation: "*. {java, htm, jar}and "**. {java, htm, jar}will match any file with file endings java, htm, or jar.

QUESTION 148
Given:

```
import java.io.File;

import java.nio.file.Path;

public class Test12 {

static String displayDetails(String path, int location) {

Path p = new File(path).toPath();

String name = p.getName(location).toString();

return name;

}
public static void main(String[] args) {
String path = "project//doc//index.html";

String result = displayDetails(path,2);

System.out.print(result);

}

}
```

What is the result?

A. doc
B. index.html
C. an IllegalArgumentException is thrown at runtime.
D. An InvalidPthException is thrown at runtime.
E. Compilation fails.

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:

QUESTION 149
Given the code fragment:

```
public void otherMethod() {

printFile("");

}

public void printFile(String file) {

try (FileInputStream fis = new FileInputStream(file)) {

System.out.println (fis.read());

} catch (IOException e) {
```

A. printStackTrace();
   }
   Why is there no output when otherMethod is called?
B. An exception other than IOException is thrown.
C. Standard error is not mapped to the console.
D. There is a compilation error.
E. The exception is suppressed.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: The code compiles fine
The line
FileInputStream fis = new FileInputStream(file))
will fail at runtime since file is an empty string.

Note:
public void printStackTrace()
Prints this throwable and its backtrace to the standard error stream.

QUESTION 150
Given the code fragment:

```
public void ReadFile (String source) {

char[] c = new char [128];

int cLen = c.length;

try (FileReader fr = new FileReader (source)) {

int count = 0;

int read = 0;

while ((read = fr.read(c)) != -1) {

count += read;

}

System.out.println("Read: " + count + " characters.");

} catch (IOException i) {

}
```

What change should you make to this code to read and write strings instead of character arrays?

A. ChangeFileReader to Readers.
B. ChangeFileReader to DataReader.
C. ChangeFileReader to File.
D. ChangeFileReader to BufferReader.

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: public class BufferedReader
extends Reader
Read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.
The buffer size may be specified, or the default size may be used. The default is large enough for most purposes.