1z0-804.exam.87q

Number: 1z0-804
Passing Score: 800
Time Limit: 120 min

https://www.gratisexam.com/

1z0-804

Java SE 7 Programmer II

Exam A

QUESTION 1
Given:

```
interface Event {
   String type = "Event";
   public void details();
}

class Quiz {
   static String type = "Quiz";
}

public class PracticeQuiz 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: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 2
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 3

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:
getName() is missing in John, hence Pupils getName() is invoked and the String in Pupils scope returned.

QUESTION 4
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 5
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: The compilation fails at line start();
Erstellen eines statischen Verweises auf die nicht statische Methode start() vom Typ Runner nicht möglich. Exception in thread "main" java.lang.RuntimeException: Uncompilable source code - non-static method start() cannot be referenced from a static context

QUESTION 6
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 enum can 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 7
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:

QUESTION 8
Which represents part of a DAO design pattern?

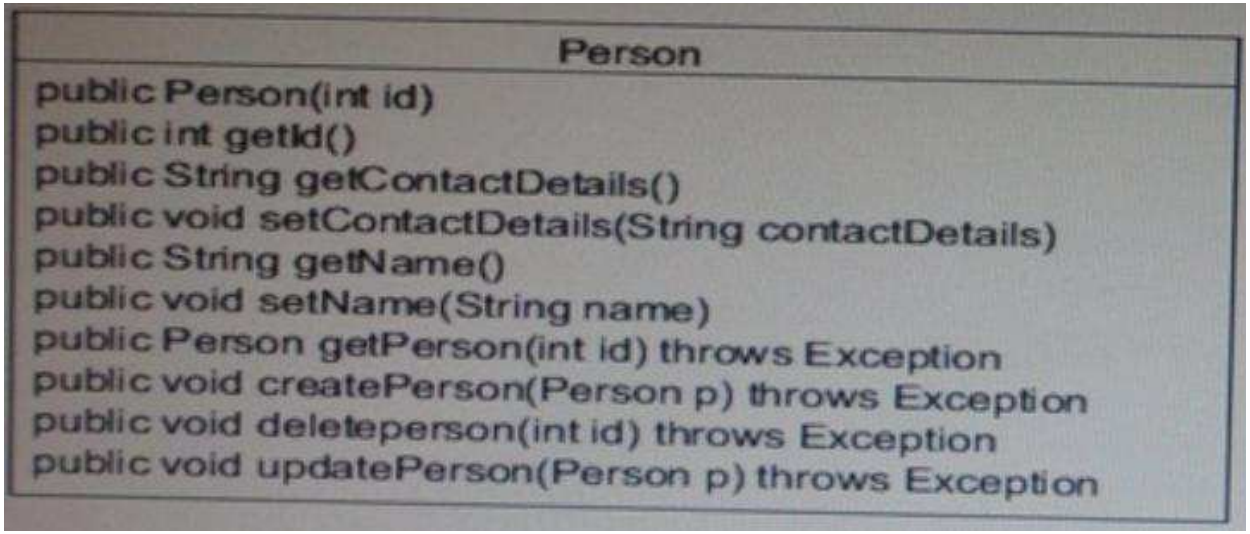
A. interface EmployeeDAO {
int getID();
Employee findByID (intid);
void update();
void delete();
}
B. class EmployeeDAO {
int getID() { return 0;}
Employee findByID (int id) { return null;}
void update () {}
void delete () {}
}
C. class EmployeeDAO {
void create (Employee e) {}
void update (Employee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

QUESTION 9
Given:



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)

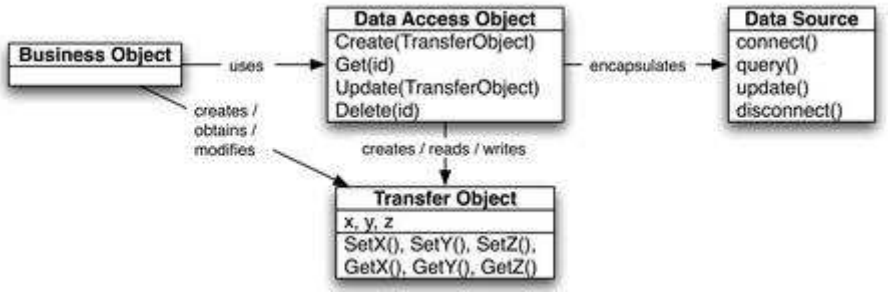
B. public int getId ()
public String getContractDetails()
public String getName()
public Person getPerson(int id) throws Exception

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.
CRUD
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; }
57

QUESTION 10
Which is a key aspect of composition?

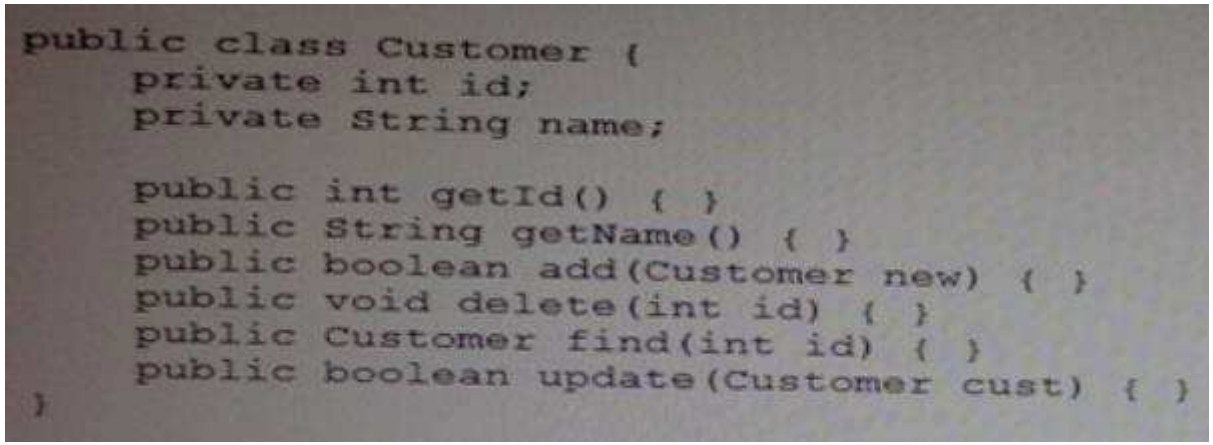
A. Using inheritance
B. Method delegation
C. Creating abstract classes
D. Implementing the composite interface

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
Note 2: 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 11
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?

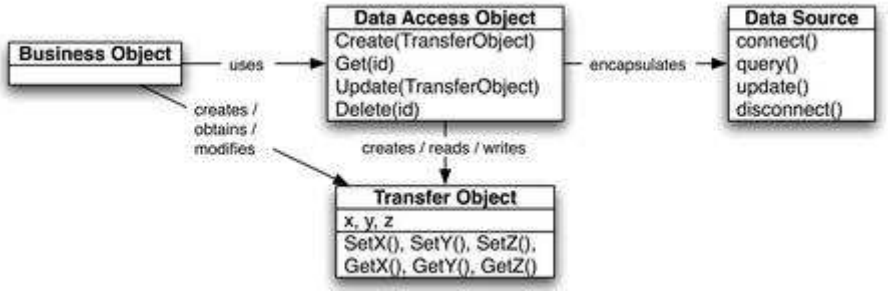
A. Make the Customer class abstract.
B. Make the customer class an interface.
C. Move the add, delete, find, and update methods into their own implementation class.
D. Create an interface that defines the signatures of the add, delete, find, and update methods.
E. Make the add, delete, and find, and 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; }
102

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 12
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: CE
Section: (none)
Explanation

Explanation/Reference:

QUESTION 13
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 14
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: AD
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
C: The constructor should be private
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.
OPTION A == SHOW THE LAZY initialization WITHOUT DOUBLE CHECKED LOCKING TECHNIQUE ,BUT
ITS CORRECT
OPTION D == Serialzation and thraead-safety guaranteed and with couple of line of code enum Singleton pattern is best way to create Singleton in Java 5 world.
AND THERE ARE 5 WAY TO CREATE SINGLETON CLASS IN JAVA
1>>LAZY LOADING (initialization) USING SYCHRONIZATION
2>>CLASS LOADING (initialization) USING private static final Singleton instance = new Singleton();
3>>USING ENUM
4>>USING STATIC NESTED CLASS
5>>USING STATIC BLOCK
AND MAKE CONSTRUCTOR PRIVATE IN ALL 5 WAY.

QUESTION 15
Given:

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

interface Glommer { }
interface Plinkable { }

public class Flimmer implements Plinkable {
   List<Tagget> t = new ArrayList<Tagget>();
}
 class Flommer extends Flimmer {
    String s = "hey";
 }

 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. Tagget has-a Glommer.
D. Flimmer is-a ArrayList.
E. Tagget has-a doStuff()
F. Tagget is-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: The has-a relationship has an encapsulation feature (like private or protected modifier used before each member field or method).
Here Tagget has-a method doStuff()
F: Tagget implements Glommer.
Tagget is-a Glommer.
Note: The has-a relationship has an encapsulation feature (like private or protected modifier used before each member field or method).

QUESTION 16
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. 4
   1
B. 1
   2
C. 32
D. 21
E. 2
   3

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
binarySearch
public static <T> int binarySearch(List<? extends Comparable<? super T>> list, T key)
Searches the specified list for the specified object using the binary search algorithm.
The list must be sorted into ascending order according to the natural ordering of its elements (as by the sort (List) method) prior to making this call. If it is not sorted, the results are undefined.
Parameters:
list - the list to be searched.
key - the key to be searched for.
Returns:
the index of the search key, if it is contained in the list; otherwise, (-(insertion point) - 1).

QUESTION 17
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:
Example:
public class Pocket<T>
{
private T value;
public Pocket() {}
public Pocket(T value) { this.value = value; }
public void set(T value) { this.value = value; }
public T get() { return value; }
public boolean isEmpty() { return value != null; }
public void empty() { value = null; }
}

QUESTION 18
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:
push
void push(E e)
Pushes an element onto the stack represented by this deque (in other words, at the head of this deque) if it is possible to do so immediately without violating capacity restrictions, returning true upon success and throwing an IllegalStateException if no space is currently available.
This method is equivalent to addFirst(E).
pop
E pop()
Pops an element from the stack represented by this deque. In other words, removes and returns the first element of this deque.
This method is equivalent to removeFirst().
Returns:
the element at the front of this deque (which is the top of the stack represented by this deque)
Throws:
NoSuchElementException - if this deque is empty

QUESTION 19
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:
add
boolean add(E e)
Inserts the specified element into the queue represented by this deque (in other words, at the tail of this deque) if it is possible to do so immediately without violating capacity restrictions, returning true upon success and throwing an IllegalStateException if no space is currently available. When using a capacity-restricted deque, it is generally preferable to use offer.
This method is equivalent to addLast(E).
remove
E remove()
Retrieves and removes the head of the queue represented by this deque (in other words, the first element of This deque). This method differs from poll only in that it throws an exception if this deque is empty.
This method is equivalent to removeFirst().
Returns:
The head of the queue represented by this deque
Class ArrayDeque

QUESTION 20
Which concept allows generic collections to interoperate with java code that defines collections that use raw types?

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

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:
The type erasure of its leftmost bound, or type Object if no bound was specified.
Examples:

type parameters type erasure
List<String> List
Map.Entry<String,Long> Map.Entry
<T extends Cloneable & Comparable<T>> Cloneable
<T extends Object & Comparable<T>> Object
<T> T[] toArray(T[] a) Object[] toArray(Object[] a)
The type erasure process can be imagined as a translation from generic Java source code back into regular Java code. In reality the compiler is more efficient and translates directly to Java byte code. But the byte code created is equivalent to the non-generic Java code.

QUESTION 21
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 fine but java.lang.NullPointerException is thrown at runtime.
x has no value. The code would run if line 2 was changed to:
Integer x = 3;

QUESTION 22
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");
put
V put(K key, V value)
Associates the specified value with the specified key in this map (optional operation). If the map previously contained a mapping for the key, the old value is replaced by the specified value. (A map m is said to contain a mapping for a key k if and only if m.containsKey(k) would return true.)

Parameters:
key - key with which the specified value is to be associated
value - value to be associated with the specified key
Returns the previous value associated with key, or null if there was no mapping for key. (A null return can also indicate that the map previously associated null with key, if the implementation supports null values.)

QUESTION 23
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:
Strings are immutable, therefore no change at line: svar.replace(svar,"convertibles");
Format String Syntax:
%[argument_index$][flags][width][.precision]conversion
The optional argument_index is a decimal integer indicating the position of the argument in the argument list.
The first argument is referenced by "1$", the second by "2$", etc.
The optional flags is a set of characters that modify the output format. The set of valid flags depends on the conversion.
's', 'S' general
'd' integral The result is formatted as a decimal / integer

QUESTION 24
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:
regex: Java / one or more anything !!! / ends with a digit
so it is the source string

QUESTION 25
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);
   s.useDelimiter(",\\s*");
   while (s.hasNext()) {
      System.out.println(s.next());
    }
    s.close();
}
}
```

What is the result?

A. Pastel
   Enamel
   Fresco
   Gouache
B. Pastel
   *Enamel
   Fresco
   *Gouache
C. Pastel
   Enamel

Fresco
Gouache

D. Pastel
Enamel, Fresco
Gouache

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
regex explanation:
, = ,
\ = masks the following
\s = A whitespace character: [\t \n \x0B \f \r]
* = Greedy Quantifier: zero or more times
Delimiter: comma + zero or more whitespace characters

QUESTION 26
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
```

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: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
javac Test.java
will compile the 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 throw 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 class AssertionError extends Error
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 27
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.IllegalArgumentException

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 28
Given the code fragment:

```
static 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: D
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 29
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 the FileInputStream will be read (see ** below).
The content of the FileInputStream will 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 30
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);
      }
   }
}
```

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: D
Section: (none)
Explanation

Explanation/Reference:
Explanation:
!! Compilation fails if FileNotFoundException is tried to catch (Line 12)
(The exception FileNotFoundException is already caught by the alternative IOException)
---
if this is removed will be thrown a FileNotFoundException at line 4.

QUESTION 31
Given the code fragment:

```
public static void displayDetails() {
   try ( BufferedReader br = new BufferedReader(new FileReader("salesreport.dat")) ) { // Line6

      String record;
      while ( (record = br.readLine()) != null) {
         System.out.println(record);
      }
      br.close();
      br = new BufferedReader(new FileReader("annualreport.dat")) ; // Line13
      while ( (record = br.readLine()) != null) {
         System.out.println(record);
      }
   } catch (IOException e) {
      System.err.println(e.getClass());
   }
}
```

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: B
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Compilation fails Line 13 : The resource br of a try-with-resources statement cannot be assigned resources are final in try-with-resources statements

QUESTION 32

Given the code fragment:

```
class Base {
   public void process() throws IOException {
      FileReader fr = new FileReader("userguide.txt");
      BufferedReader br = new BufferedReader(fr);
      String record;
      while ((record = br.readLine()) != null) {
         System.out.println(record);
      }
   }
}

class Derived extends Base {
   public void process() throws Exception {
      super.process();
      System.out.println("Success");
   }

   public static void main(String[] args) {
      try {
         new Derived().process();
      } catch (Exception e) {
         System.err.println(e.getClass());
      }
   }
}
```

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: E
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Compilation fails : Exception Exception is not compatible with throws clause in Base.process() IOException die Exception in der Derived Class Methode muss zur Base Class Methode passen.

QUESTION 33
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:
BufferedReader: mark() : Marks the present position in the stream. Subsequent calls to reset() will attempt to reposition the stream to this point.
reset() : Resets the stream to the most recent mark.
!! After last Line is read (readLine()), a trial to reset() throws IOException : Mark invalid

QUESTION 34
Given the code fragment:

```
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:
A: Compilation fails: Exception IOException is not compatible with throws clause in Base.process()
B: Compilation fails: Exception IOException is not compatible with throws clause in Base.process()
C: Compilation fails: Exception Exception is not compatible with throws clause in Base.process()
D: Compilation fails: Exception FileNotFoundException has already been caught by the alternative IOException
Alternatives in a multi-catch statement cannot be related to subclassing Alternative
java.io.FileNotFoundException is a subclass of alternative java.io.IOException
E: compiles ...

QUESTION 35
Given the code fragment:

```
static public void otherMethod() {
      printFile("");
      }

   static public void printFile(String file) {
      try  ( FileInputStream fis = new FileInputStream(file) ) {
         System.out.println (fis.read());
      } catch (IOException e) {
         printStackTrace();
      }
   }
```

Why is there no output when otherMethod is called?

A. An exception other than IOException is thrown.
B. Standard error is not mapped to the console.
C. There is a compilation error.
D. The exception is suppressed.

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
C: wenn printStackTrace() ohne Referenz auf das Exception object aufgerufen
A : java.io.FileNotFoundException: wenn e.printStackTrace();
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 36
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 37
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: AC
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 38
Given the code fragment:

```
    public static boolean isContentSame() throws IOException {
        Path p1 = Paths.get("D:\\Test\\faculty\\report.txt");
        Path p2 = Paths.get("E:\\Test\\student\\report.txt");

        Files.copy(p1,p2,StandardCopyOption.REPLACE_EXISTING,StandardCopyOption.COPY_ATTRIBUTES,
 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: B
Section: (none)
Explanation

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

QUESTION 39
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 WatchService API takes advantage of this support where available. However, when a file system does not support this mechanism, the WatchService 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 40
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:
p.getName(int location) = returns path' name element by index/location (starts with 0)
Example:
path = "project//doc//index.html"
p.getName(0) = project
p.getName(1) = doc
p.getName(2) = index.html

QUESTION 41
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.printf("get.Name(0): %s \n", path.getName(0));
        System.out.printf("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 42
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: C

Section: (none)
Explanation

Explanation/Reference:
Explanation:
Tricky:
Compiles successfully ! Not B !
D is correct, if Column Quantity instead of Quality
Table Item Column Quality --- System.out.println("Quantity: " + rs.getInt("Quantity"));
wenn jedoch so gewollt: die Zeile gibt Error aus (die anderen funktionieren) !!!
The connection conn is not defined. The code will not compile.

QUESTION 43
Which code fragment demonstrates the proper way to handle JDBC resources?

A.
```
try {
ResultSet rs = stmt.executeQuery (query);
statement stmt = con.createStatement();
while (rs.next()) (/* . . . */)
} catch (SQLException e) {}
```
B.
```
try {
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery (query);
while (rs.next()) (/* . . . */)
} catch (SQLException e) {}
```
C.
```
try {
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery (query);
while (rs.next()) (/* . . . */)
} finally {
rs.close();
stmt.close();
}
```
D.
```
try {
ResultSet rs = stmt.executeQuery (query);
Statement stmt = con.createStatement();
while (rs.next()) (/* . . . */)
} finally {
rs.close();
stmt.close();
}
```

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 44
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')");
        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 (e.g. conn.rollback(save1);) 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 45
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: C

Section: (none)
Explanation

Explanation/Reference:
Explanation:
!!! The given Code prints Error -- the second query clears the ResultSet !? ErrorMessage: Operation not allowed after ResultSet closed
It would print A, if second Query i set to rs = stmt.executeQuery("SELECT ID FROM Customer"); // Line ***
It would print B, if Line *** is missing. //
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 used in 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 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: if 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 An example provided by the standard Java SE API is the ServiceLoader. Among others, the JDBC 4.0 compatible drivers implement this. This way just dropping the JDBC driver JAR file folder will automatically load the driver class during Java application's startup/initialization without the need for any manual Class.forName ("com.example.Driver") line in your code.

QUESTION 47
Given the code fragment:

```
try (Connection conn = DriverManager.getConnection( url )) {
        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 ResultSet declarations.
B. Add the statement and ResultSet declarations to the try-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 48
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 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 49
Given:

```
class Counter extends Thread {
   int i = 10;
   public synchronized void display ( Counter obj) {
      try {
         Thread.sleep (5) ;
         obj.increment (this) ;
         System.out.println (i) ;
      } catch (InterruptedException ex) {  }
   }
   public synchronized void increment (Counter obj) {
      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(obj2) ;
         }
      }).start() ;
      new Thread( new Runnable() {
         public void run() {
           obj2.display(obj1) ;
         }
      }).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, then livelock 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 50
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 simultaneously is called thread safe. If a piece of code is thread safe, then it contains no race conditions. Race condition only occur when multiple threads 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.

A, B, C : false: wrong Implementation (missing int getNextId();)
E: false: synchronized (mutex Object! not Simple Data Type)

QUESTION 51
Given the code fragment:

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.concurrent.CopyOnWriteArraySet;

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:

QUESTION 52
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.
40

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

QUESTION 53
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
  }
}
```

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 singlethreaded.
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 54
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 55
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.high = high;
   }

   Integer computeDirectly() {   /* ... */  }

   protected void compute() {
      if (high - low <= THRESHOLD)
         return computeDirectly();
         int mid = (low+high) / 2;
         invokeAll(new MyTask(low,mid), new MyTask(mid, high));
      }
   }
```

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: AD
Section: (none)
Explanation

Explanation/Reference:
Explanation:
Note 1: A RecursiveTask is a recursive result-bearing ForkJoinTask.
Note 2: The invokeAll(ForkJoinTask<?>… tasks) forks the given tasks, returning when isDone holds for each task or an (unchecked) exception is encountered, in

which case the exception is rethrown.
Note 3: Using the fork/join framework is simple. The first step is to write some code that performs a segment of the work. Your code should look similar to this:
if (my portion of the work is small enough)
do the work directly
else
split my work into two pieces
invoke the two pieces and wait for the results
Wrap this code as a ForkJoinTask subclass, typically as one of its more specialized types RecursiveTask
(which can return a result) or RecursiveAction.

QUESTION 56
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 57
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:

QUESTION 58
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 59
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: DE
Section: (none)
Explanation

Explanation/Reference:
Explanation:
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 60
Given the code fragment:

SimpleDataFormat sdf;

Which code fragment displays the three-character month abbreviation?

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

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:

QUESTION 61
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. small
D. A runtime error is produced

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
Explanation: Compiles fine, but runtime error when trying to access the Spanish Resource bundle (which does not exist):
Exception in thread "main" java.util.MissingResourceException: Can't find bundle for base name
messageBundle, locale es_ES

QUESTION 62
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 63
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 64
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 = NumberFormat.getCurrencyInstance () ;

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 65
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 ResourceApp().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) !!! String key Gets an object for the given key from this resource bundle or one of its parents.

QUESTION 66
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 67
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 68
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 5d;
}}

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 69
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 70
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 code compiles fine.
The code runs fine.
The error 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 71
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 72
Which two codes correctly represent a standard language locale code?

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

F. u8

Correct Answer: AB
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.

QUESTION 73
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: CE
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 74
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 75
Assuming the port statements are correct, which two (three?) 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(fos);
   dos.writeByte(0);
   dos.close();
D. OutputStream fos = new FileOutputStream ("/tmp/data.bin"); fos.writeByte(0);
   fos.close();

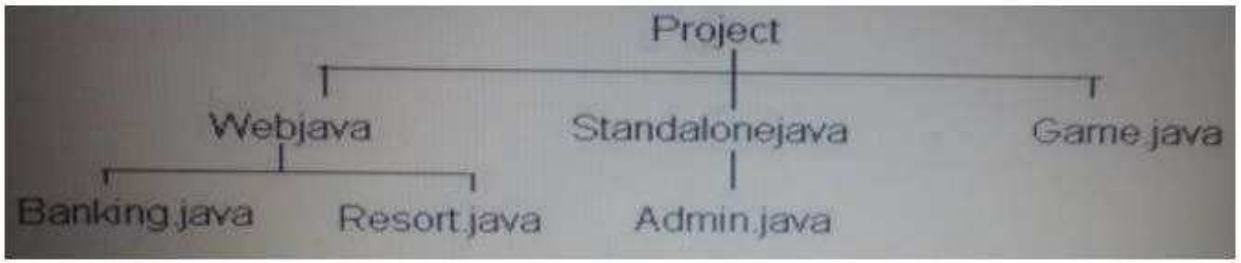
Correct Answer: ABC
Section: (none)
Explanation

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 76
View the exhibit:



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 FileVisitResult.CONTINUE; // ursprünglich stand hier nur CONTINUE !!!! ????? !!!!
   }
}

public class Visitor {
   public static void main(String[] args) throws IOException {
      Finder finder = new Finder();
      Files.walkFileTree(Paths.get("c:/_Workspace/OCP/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: C
Section: (none)
Explanation

Explanation/Reference:
Explanation:
C: 4 Falls geändert zu: return FileVisitResult.CONTINUE sonst A: weil CONTINUE als Konstante unbekannt
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 77
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 78
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 regarding SuperX' to determine the level of cohesion.

QUESTION 79
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 80
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:
int compare(T obj1, T obj2)
0 if equal
positive if obj1 greater
negative if obj2 greater
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 81
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 82
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, the implements 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 83
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.
java.lang.NullPointerException
because only one element of list is initialized : element [0]
elements [1] and [2] equals null
alte Begründung:
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 84
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:
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 85
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 (Frage unvolständig!!!)
F. Compilation fails
G. A NullPointerException is thrown at runtime

Correct Answer: E
Section: (none)
Explanation

Explanation/Reference:

QUESTION 86
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: B
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.
Assertion is true because getCount(arr) = 3 and Length of array is 4

The following line:
assert (getCount(arr) < arr.length);
where the Boolean expression getCount(arr) < arr.length will evaluate to false, will ensure that an AssertionError 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 87
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: