

PDFVCE



Choose the version that fits your needs	PDF Version	Desktop Test Engine	Online Test Engine
Latest and Up-to-Date exam dumps with real exam questions answers.	✓	✓	✓
Get 12-Months free updates without any extra charges.	✓	✓	✓
Experience same exam environment before appearing in the certification exam.	✗	✓	✓
100% exam passing guarantee in the first attempt.	✓	✓	✓
20% discount on more than one license and 30% discount on 5+ license purchases.	✗	✓	✓
100% secure purchase on SSL.	✓	✓	✓
Completely private purchase without sharing your personal info with anyone.	✓	✓	✓

<http://www.pdfvce.com>

Highly Efficiently Exam Tool and Effective Exam Practice Materials

Exam : **1z0-809**

Title : **Java SE 8 Programmer II**

Vendor : **Oracle**

Version : **DEMO**

NO.1 Which two reasons should you use interfaces instead of abstract classes? (Choose two.)

- A.** You expect that classes that implement your interfaces have many common methods or fields, or require access modifiers other than public.
- B.** You expect that unrelated classes would implement your interfaces.
- C.** You want to share code among several closely related classes.
- D.** You want to declare non-static on non-final fields.
- E.** You want to take advantage of multiple inheritance of type.

Answer: B,E

NO.2 Given the definition of the Vehicle class:

```

Class Vehicle {
int distance;//line n1
Vehicle (int x) {
this distance = x;
}
public void increSpeed(int time) {//line n2
int timeTravel = time;//line n3
class Car {
int value = 0;
public void speed () {
value = distance /timeTravel;
System.out.println ("Velocity with new speed"+value+"kmph");
}
}
new Car().speed();
}
}

```

and this code fragment:

```

Vehicle v = new Vehicle (100);
v.increSpeed(60);

```

What is the result?

- A.** Velocity with new speed 1 kmph
- B.** A compilation error occurs at line n1.
- C.** A compilation error occurs at line n2.
- D.** A compilation error occurs at line n3.

Answer: D

NO.3 Given:

```

class Vehicle implements Comparable<Vehicle>{
int vno;
String name;
public Vehicle (int vno, String name) {
this.vno = vno,;
this.name = name;
}
}

```

```
public String toString () {
return vno + ":" + name;
}
public int compareTo(Vehicle o) {
return this.name.compareTo(o.name);
}
```

and this code fragment:

```
Set<Vehicle> vehicles = new TreeSet <> ();
vehicles.add(new Vehicle (10123, "Ford"));
vehicles.add(new Vehicle (10124, "BMW"));
System.out.println(vehicles);
```

What is the result?

- A. [10123:Ford, 10124:BMW]
- B. [10124:BMW, 10123:Ford]
- C. A compilation error occurs.
- D. A ClassCastException is thrown at run time.

Answer: A

NO.4 Given:

```
public class Foo<K, V> {
    private K key;
    private V value;

    public Foo(K key, V value) { this.key = key; this.value = value; }

    public static <T> Foo<T, T> twice(T value) { return new Foo<T, T>(value, value); }

    public K getKey() { return key; }
    public V getValue() { return value; }
}
```

Which option fails?

- A. Foo<String, Integer> mark = new Foo<String, Integer> ("Steve", 100);
- B. Foo<String, String> pair = Foo.<String>twice ("Hello World!");
- C. Foo<Object, Object> percentage = new Foo<String, Integer>("Steve", 100);
- D. Foo<String, String> grade = new Foo <> ("John", "A");

Answer: A

NO.5 Given the code fragment:

```
Stream<Path> files = Files.list(Paths.get(System.getProperty("user.home"))); files.forEach (fName ->
{//line n1 try { Path aPath = fName.toAbsolutePath();//line n2 System.out.println(fName + ":"
+ Files.readAttributes(aPath, Basic.File.Attributes.class).creationTime ());
} catch (IOException ex) {
ex.printStackTrace();
});
```

What is the result?

- A. All files and directories under the home directory are listed along with their attributes.
- B. A compilation error occurs at line n1.
- C. The files in the home directory are listed along with their attributes.

D. A compilation error occurs at line n2.

Answer: C

NO.6 Given:

```
class FuelNotAvailException extends Exception { }
class Vehicle {
void ride() throws FuelNotAvailException { //line n1
System.out.println("Happy Journey!");
}
}
class SolarVehicle extends Vehicle {
public void ride () throws FuelNotAvailException { //line n2
super ride ();
}
}
```

and the code fragment:

```
public static void main (String[] args) throws Exception {
Vehicle v = new SolarVehicle ();
v.ride();
}
```

Which modification enables the code fragment to print Happy Journey!?

- A.** Replace line n1 with public void ride() throws FuelNotAvailException {
- B.** Replace line n1 with protected void ride() throws Exception {
- C.** Replace line n2 with void ride() throws Exception {
- D.** Replace line n2 with private void ride() throws FuelNotAvailException {

Answer: C

NO.7 Given the content of /resources/Message.properties:

welcome1="Good day!"

and given the code fragment:

```
Properties prop = new Properties ();
FileInputStream fis = new FileInputStream ("/resources/Message.properties"); prop.load(fis);
System.out.println(prop.getProperty("welcome1"));
System.out.println(prop.getProperty("welcome2", "Test")); //line n1
System.out.println(prop.getProperty("welcome3")); What is the result?
```

- A.** Good day!Testfollowed by an Exception stack trace
- B.** Good day!followed by an Exception stack trace
- C.** Good day!Testnull
- D.** A compilation error occurs at line n1.

Answer: C

NO.8 Given the code fragment:

```
List<String> listVal = Arrays.asList("Joe", "Paul", "Alice", "Tom");
System.out.println (
// line n1
```

);

Which code fragment, when inserted at line n1, enables the code to print the count of string elements whose length is greater than three?

- A. `listVal.stream().filter(x -> x.length()>3).count()`
- B. `listVal.stream().map(x -> x.length()>3).count()`
- C. `listVal.stream().peek(x -> x.length()>3).count().get()`
- D. `listVal.stream().filter(x -> x.length()>3).mapToInt(x -> x).count()`

Answer: A

NO.9 Given the code fragment:

```
List<String> empDetails = Arrays.asList("100, Robin, HR", "200, Mary, AdminServices", "101, Peter, HR");  
empDetails.stream()  
.filter(s-> s.contains("r"))  
.sorted()  
.forEach(System.out::println); //line n1
```

What is the result?

- A. 100, Robin, HR101, Peter, HR
- B. E. A compilation error occurs at line n1.
- C. 101, Peter, HR200, Mary, AdminServices
- D. 100, Robin, HR200, Mary, AdminServices101, Peter, HR

Answer: D

NO.10 Which class definition compiles?

A. `public class ProductCode {
private String code;
public ProductCode(String code) {
this.code = code;
}
}`

B. `public class ProductCode {
private String code;
private ProductCode(String code) {
this.code = code;
}
}`

C. `public class ProductCode {
private String code;
public ProductCode() {
this.code = "default";
}
}`

C. `public class ProductCode {
private String code;
public ProductCode(String code) {`

```
if (code == null) {  
    throw new IllegalArgumentException("Code cannot be null");  
}  
this.code = code;  
}  
}
```

Answer: B

NO.11 Given the code fragments:

```
interface CourseFilter extends Predicate<String> {  
    public default boolean test (String str) {  
        return str.equals ("Java");  
    }  
}  
and  
List<String> strs = Arrays.asList("Java", "Java EE", "Java ME");  
Predicate<String> cf1 = s -> s.length() > 3;  
Predicate cf2 = new CourseFilter() { //line n1  
    public boolean test (String s) {  
        return s.contains ("Java");  
    }  
};  
long c = strs.stream()  
    .filter(cf1)  
    .filter(cf2//line n2  
    .count();  
System.out.println(c);  
What is the result?
```

- A. 2
- B. 3
- C. A compilation error occurs at line n1.
- D. A compilation error occurs at line n2.

Answer: B

NO.12 Which two statements are true about localizing an application? (Choose two.)

- A. Support for new regional languages does not require recompilation of the code.
- B. Textual elements (messages and GUI labels) are hard-coded in the code.
- C. Language and region-specific programs are created using localized data.
- D. Resource bundle files include data and currency information.
- E. Language codes use lowercase letters and region codes use uppercase letters.

Answer: A,E

References:

NO.13 Given:

```
interface Interfacel {
    public default void sayHi() {
        System.out.println("Hi Interface-1");
    }
}

interface Interface2 {
    public default void sayHi() {
        System.out.println("Hi Interface-2");
    }
}

public class MyClass implements Interfacel, Interface2 {
    public static void main(String[] args) {
        Interfacel obj = new MyClass();
        obj.sayHi();
    }
    public void sayHi() {
        System.out.println("Hi MyClass");
    }
}
```

What is the result?

- A. Hi Interface-2
- B. A compilation error occurs.
- C. Hi Interface-1
- D. Hi MyClass

Answer: D

NO.14 Given the code fragment:

```
Path source = Paths.get ("/data/december/log.txt");
```

```
Path destination = Paths.get("/data");
```

```
Files.copy (source, destination);
```

and assuming that the file /data/december/log.txt is accessible and contains:

```
10-Dec-2014 - Executed successfully
```

What is the result?

- A. A file with the name log.txt is created in the /data directory and the content of the /data/december/log.txt file is copied to it.
- B. The program executes successfully and does NOT change the file system.
- C. A FileNotFoundException is thrown at run time.
- D. A FileAlreadyExistsException is thrown at run time.

Answer: D

NO.15 Given:

```
class Book {
```

```
int id;
String name;
public Book (int id, String name) {
this.id = id;
this.name = name;
}
public boolean equals (Object obj) { //line n1
boolean output = false;
Book b = (Book) obj;
if (this.name.equals(b.name))
output = true;
}
return output;
}
}
```

and the code fragment:

```
Book b1 = new Book (101, "Java Programing");
Book b2 = new Book (102, "Java Programing");
System.out.println (b1.equals(b2)); //line n2
```

Which statement is true?

- A.** The program prints true.
- B.** The program prints false.
- C.** A compilation error occurs. To ensure successful compilation, replace line n1 with:`boolean equals (Book obj) {`
- D.** A compilation error occurs. To ensure successful compilation, replace line n2 with:`System.out.println (b1.equals((Object) b2));`

Answer: A

NO.16 Given that version.txt is accessible and contains:

1234567890

and given the code fragment:

```
try (FileInputStream fis = new FileInputStream("version.txt");
    InputStreamReader isr = new InputStreamReader(fis);
    BufferedReader br = new BufferedReader(isr);) {
    if (br.markSupported()) {
        System.out.print((char) br.read());
        br.mark(2);
        System.out.print((char) br.read());
        br.reset();
        System.out.print((char) br.read());
    }
} catch (Exception e) {
    e.printStackTrace();
}
```

What is the result?

- A.** 121

- B. 122
- C. 135
- D. The program prints nothing.

Answer: B

NO.17 Given:

```
public class Foo<K, V> {
    private K key;
    private V value;

    public Foo(K key, V value) { this.key = key; this.value = value; }

    public static <T> Foo<T, T> twice(T value) { return new Foo<T, T>(value, value); }

    public K getKey() { return key; }
    public V getValue() { return value; }
}
```

Which option fails?

- A. `Foo<String, Integer> mark = new Foo<Object, Object>("Steve", 100);`
- B. `Foo<String, String> pair = Foo.<String>twice("Hello World!");`
- C. `Foo<Object, Object> percentage = new Foo<Object, Object>("Steve", 100);`
- D. `Foo<String, String> grade = new Foo <>("John", "A");`

Answer: C

NO.18 Given:

```
class Student {
    String course, name, city;
    public Student (String name, String course, String city) {
        this.course = course; this.name = name; this.city = city;
    }
    public String toString() {
        return course + ":" + name + ":" + city;
    }
    public String getCourse() {return course;}
    public String getName() {return name;}
    public String getCity() {return city;}
}
```

and the code fragment:

```
List<Student> stds = Arrays.asList(
    new Student ("Jessy", "Java ME", "Chicago"),
    new Student ("Helen", "Java EE", "Houston"),
    new Student ("Mark", "Java ME", "Chicago"));
stds.stream()
    .collect(Collectors.groupingBy(Student::getCourse))
    .forEach(src, res) -> System.out.println(res));
```

What is the result?

- A. A compilation error occurs.
- B. Java EEJava ME
- C. [Java EE: Helen:Houston][Java ME: Jessy:Chicago, Java ME: Mark:Chicago]

D. [Java ME: Jessy:Chicago, Java ME: Mark:Chicago][Java EE: Helen:Houston]

Answer: B

NO.19 Given:

Book.java:

```
public class Book {  
private String read(String bname) { return "Read" + bname }  
}
```

EBook.java:

```
public class EBook extends Book {  
public class String read (String url) { return "View" + url }  
}
```

Test.java:

```
public class Test {  
public static void main (String[] args) {  
Book b1 = new Book();  
b1.read("Java Programing");  
Book b2 = new EBook();  
b2.read("http://ebook.com/ebook");  
}  
}
```

What is the result?

- A. Read Java ProgrammingView http:/ ebook.com/ebook
- B. Read Java ProgrammingRead http:/ ebook.com/ebook
- C. The EBook.java file fails to compile.
- D. The Test.java file fails to compile.

Answer: D

NO.20 Given the code fragment:

```
try {  
    Properties prop = new Properties();  
    prop.put("user", userName);  
    prop.put("password", passWord);  
    Connection conn = DriverManager.getConnection(dbURL, prop);  
    if(conn != null){  
        System.out.print("Connection Established");  
    }  
} catch (Exception e) {  
    System.out.print(e);  
}
```

and the information:

- * The required database driver is configured in the classpath.
- * The appropriate database is accessible with the dbURL, username, and passWord exists.

What is the result?

- A. A ClassNotFoundException is thrown at runtime.

- B. The program prints nothing.
- C. The program prints Connection Established.
- D. A SQLException is thrown at runtime.

Answer: C

NO.21 Given:

```
class Vehicle {  
    int vno;  
    String name;  
    public Vehicle (int vno, String name) {  
        this.vno = vno;  
        this.name = name;  
    }  
    public String toString () {  
        return vno + ":" + name;  
    }  
}
```

and this code fragment:

```
Set<Vehicle> vehicles = new TreeSet <> ();  
vehicles.add(new Vehicle (10123, "Ford"));  
vehicles.add(new Vehicle (10124, "BMW"));  
System.out.println(vehicles);
```

What is the result?

- A. 10123 Ford10124 BMW
- B. 10124 BMW10123 Ford
- C. A compilation error occurs.
- D. A ClassCastException is thrown at run time.

Answer: D