Oracle.1Z0-819.v2022-08-08.q109

試験コード：	1Z0-819
試験名称：	Java SE 11 Developer
認証ベンダー：	Oracle
無料問題の数：	109
バージョン：	v2022-08-08
ページの閲覧量：	513
問題集の閲覧量：	8170

https://www.jpnshiken.com/shiken/Oracle.1Z0-819.v2022-08-08.q109.html

質問: **1**

与えられた：

```
1. interface Pastry {
2.     void getIngredients();
3. }
4. abstract class Cookie implements Pastry {}
5.
6. class ChocolateCookie implements Cookie {
7.     public void getIngredients() {}
8. }
9. class CoconutChocolateCookie extends ChocolateCookie
0.     void getIngredients(int x) {}
1 }
```

どちらが正しいですか？

A. 6行目のエラーが原因でコンパイルが失敗します。

B. 9行目のエラーが原因でコンパイルが失敗します。

C. 7行目のエラーが原因でコンパイルが失敗します。

D. 2行目のエラーが原因で、コンパイルが失敗します。

E. 4行目のエラーが原因でコンパイルが失敗します。

F. コンパイルは成功します。

G. 10行目のエラーが原因でコンパイルが失敗します。

正解: **A** (コメントを発表する)

質問: **2**

与えられた：

```
1. {
2.    Iterator iter = List.of(1,2,3).iterator();
3.    while (iter.hasNext()) {
4.      foo(iter.next());
5.    }
6.    Iterator iter2 = List.of(1,2,3).iterator();
7.    while (iter.hasNext()) {
8.      bar(iter2.next());
9.    }
10. }
11.   for (Iterator iter = List.of(1,2,3).iterator(); iter.hasNext(); ) {
12.     foo(iter.next());
13.   }
14.   for (Iterator iter2 = List.of(1,2,3).iterator(); iter.hasNext(); ) {
15.     bar(iter2.next());
16.   }
```

コンパイル時エラーが発生するループはどれですか？

A. ループ開始行3

B. ループ開始行11

C. ループ開始行14

D. ループ開始行7

正解: **(**正解を表示します**)**

質問: **3**

与えられた：

```
public static void main(String[] args) {
   try (Reader reader1 = new FileReader("File1.txt");
        Reader reader2 = new FileReader("File2.txt");
        Reader reader3 = new FileReader("File3_txt")) {

   } catch (IOException ex) {
      Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
   }
   // Line 1
   System.out.println("Done");
}
```

実行して3つのファイルがすべて存在する場合、1行目の各リーダーの状態はどうなりますか？

A. コンパイルは失敗します。

B. 3つのリーダーすべてが閉じられました。

C. reader1のみが閉じられました。

D. 3つのリーダーはすべて開いたままです。

正解: **A (**コメントを発表する**)**

質問: **4**

変数番号=List.of（0,1,2,3,4,5,6,7,8,9）;

数値の平均を計算したいとします。これを達成する2つのコードはどれですか？（2つ選択してください。）

A. double avg = numbers.stream（）.parallel（）.averagingDouble（a-> a）;

B. double avg = numbers.parallelStream（）.mapToInt（m-> m）.average（）.getAsDouble（）;

C. double avg = numbers.stream（） mapToInt i-> i）.average（） parallel（）;

D. double avg = numbers.stream（） average（） getAsDouble（）;

E. double avg = numbers.stream（） collect Collectors.averagingDouble n-> n）;

正解: **B,D** (コメントを発表する)

```
import java.io.*;
import java.util.*;
class Hello {
public static void main(String[] args) {

    var numbers = List.of(0,1,2,3,4,5,6,7,8,9);
  double avg = numbers.parallelStream().mapToInt (m -> m).average().getAsDouble();

    }
}
```

質問: **5**

与えられた：

```
import java.io.*;
public class Tester {
   public static void main(String[] args) {
      try {
         doA();
         doB();
      } catch(IOException e) {
          System.out.print("c");
          return;
      } finally{
          System.out.print("d");
      }
      System.out.print("f");
   }
   private static void doA() {
      System.out.print("a");
      if (false) {
         throw new IndexOutOfBoundsException();
       }
   }
   private static void doB() throws FileNotFoundException {
      System.out.print("b");
      if (true) {
         throw new FileNotFoundException();
      }
   }
}
```

結果はどうなりますか？

A. abd

B. abcd

C. abdf

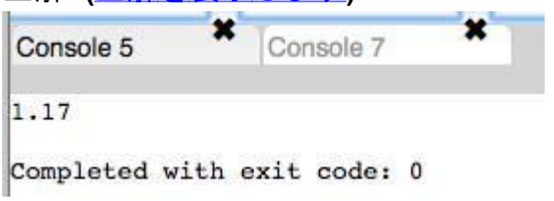
D. adf
E. コンパイルは失敗します。
正解: **B** (コメントを発表する)

質問: **6**
コードフラグメントが与えられた場合：
public void foo（Function <Integer、String> fun）{...}
どちらの2つをコンパイルしますか？ 2つ選択してください。）
A. foo（n-> Integer.toHexString（n））
B. foo（toHexString）
C. foo（n-> n + 1）
D. foo（int n-> Integer.toHexString（n））
E. foo（n-> Integer :: toHexString）
F. foo（Integer :: toHexString）
G. foo（n :: toHexString）
H. foo（int n）-> Integer.toHexString（n））
正解: (正解を表示します)
説明/参照：

質問: **7**
与えられた：

```
public class Tester {
    public static void main(String[] args) {
        byte x = 7, y = 6;
        // line 1
        System.out.println(z);
    }
}
```


1行目に追加したときに1.17の出力を生成する式はどれですか？
A. float z = （float）Math.round（（float）x / y * 100）/ 100）;
B. float z = Math.round（（int）（x / y）、2）;
C. float z = Math.round（（float）x / y、2）;
D. float z = Math.round（（float）x / y * 100）/（float）100;
正解: (正解を表示します)



質問: **8**

与えられた：

```
public class Foo {
    public <T> Collection<T> foo(Collection<T> arg) { ... }
}

and

public class Bar extends Foo { ... }
```

メソッドがBarに追加された場合、正しい2つのステートメントはどれですか。2つ選択してください。）

A. public <T> Collection <T> foo（Stream <T> arg）{...}はFoo.fooをオーバーロードします。

B. public Collection <String> foo（Collection <String> arg）{...}はFoo.fooをオーバーライドします。

C. public <T> List <T> foo（Collection <T> arg）{...}はFoo.fooをオーバーライドします。

D. public <T> Collection <T> bar（Collection <T> arg）{...}はFoo.fooをオーバーロードします。

E. public <T> Iterable <T> foo（Collection <T> arg）{...}はFoo.fooをオーバーライドします。

F. public <T> Collection <T> foo（Collection <T> arg）{...}はFoo.fooをオーバーロードします。

正解: **(**正解を表示します**)**

質問: **9**

与えられた：

```
public class Employee {
   private String name;
   private LocalDate birthday;
   // the constructors, getters, and setters methods go here
}
```

と

```
List<Employee> roster = new ArrayList<>();
// ...
Predicate<Employee> y = (Employee e) -> e.getBirthday()
    .isBefore(IsoChronology.INSTANCE.date(1989, 1, 1));
Set<String> s1 = roster.stream()
// Line 1
```

1行目のどのコードフラグメントがs1セットに、1月1日より前に生まれたすべての従業員の名前を含むようにします。

1989年？

A.
```
.collect(Collectors.partitioningBy(y))
.get(true)
.stream()
.map(Employee::getName)
.collect(Collectors.toCollection(TreeSet::new));
```

B.
```
.collect(Collectors.partitioningBy(y))
.get(true)
.map(Employee::getName)
.collect(Collectors.toSet());
```

C.
```
.collect(Collectors.partitioningBy(y, Collectors.mapping(
    Employee::getName, Collectors.toSet())));
```

D.
```
.collect(Collectors.partitioningBy(y, Collectors.groupingBy(
    Employee::getName, Collectors.toCollection(TreeSet::new))));
```

A. オプションC
B. オプションA
C. オプションD
D. オプションB
正解: **D** (コメントを発表する)

質問: **10**
与えられた：
var data = new ArrayList <> ();
data.add ("Peter");
data.add (30);
data.add ("Market Road");
data.set (1、25);
data.remove (2);
data.set (3、1000L);
System.out.print (data);
出力は何ですか？
A. [Market Road、1000]
B. [ピーター、30、マーケットロード]
C. [ピーター、25、ヌル、1000]
D. 実行時に例外がスローされます。
正解: **D** (コメントを発表する)

質問: **11**
与えられた：

```
public class Tester {
   static class Person implements /* line 1 */ {
      private String name;
      Person(String name) { this.name = name; }
      /* line 2 */
   }
   public static void main(String[] args) {
      Person[] people = {new Person("Joe"),
                         new Person("Jane"),
                         new Person("John")};
      Arrays-sort(people);
      for(Person person: people) {
         System.out.println(person.name);
      }
   }
}
```

コードで次の出力を生成する必要があります。
ジョン
ジョー
ジェーン
出力を生成するには、1行目と2行目にどのコードフラグメントを挿入する必要がありますか？
A. 1行目にComparable<Person>を挿入します。
入れる
public int compare（Person p1、Person p2）{
p1.name.compare（p2.name）;を返します。
}
2行目。
B. 1行目にComparator<Person>を挿入します。
入れる
public int compareTo（Person person）{
person.name.compareTo（this.name）;を返します。
}
2行目。
C. 1行目にComparator<Person>を挿入します。
入れる
public int compare（Person p1、Person p2）{
p1.name.compare（p2.name）;を返します。
}
2行目。

D. 1行目にComparator<Person>を挿入します。
入れる
public int compare（Person person）{
person.name.compare（this.name）;を返します。
}
2行目。
正解: **(**正解を表示します**)**

質問: **12**
与えられた：

```
var numbers = List.of(1,2,3,4,5,6,7,8,9,10);
// line 1
StringBuilder sb = new StringBuilder();
for(int a: numbers) {
    sb.append(f.apply(a));
    sb.append(" ");
}
System.out.println(sb.toString());
```

1行目のどのステートメントがこのコードのコンパイルを可能にしますか？
A. 関数<整数、整数> f = n> n * 2;
B. 関数<整数> f = n> n * 2;
C. Function <int> f = n> n * 2;
D. Function <int、int> f = n> n * 2;
E. 関数f = n> n * 2;
正解: **(**正解を表示します**)**

```
16  public class Main {
17    public static void main(String[] args) {
18      var numbers = List.of(1,2,3,4,5,6,7,8,9,10);
19      Function<Integer, Integer> f = n -> n * 2;
20      StringBuilder sb = new StringBuilder();
21      for(int a: numbers) {
22          sb.append(f.apply(a));
23          sb.append(" ");
24      }
25      System.out.println(sb.toString());
26  }
27  }
28
```

Result
CPU Time: 0.22 sec(s), Memory: 33056 kilobyte(s)
2 4 6 8 10 12 14 16 18 20 ORACLE

質問: **13**
この列挙型宣言を考えると：

```
1. enum Letter {
2.   ALPHA(100), BETA(200), GAMMA(300);
3.   int v;
4.   Letter(int v) { this.v = v; }
5.   /* Insert code here */
6. }
```

このコードを調べてください：

System.out.println（Letter.values（）[1]）;

このコードが200を出力するには、5行目にどのコードを記述する必要がありますか？

A. public String toString（）{return String.valueOf（ALPHA.v）;}

B. public String toString（）{return String.valueOf（Letter.values（）[1]）;}

C. public String toString（）{return String.valueOf（v）;}

D. String toString（）{return "200";}

正解: **C** (コメントを発表する)

```
13 public class Main {
14 enum Letter {
15     ALPHA(100), BETA(200), GAMMA(300);
16     int v;
17     Letter(int v) { this.v = v; }
18    public String toString() { return String.valueOf(v); }
19
20
21
22 }
23 public static void main (String[] args) {
24  System.out.println(Letter.values() [1]);
25 }
26 }
27
28
```

Result

compiled and executed in 1.099 sec(s)

```
200
```

質問: **14**

与えられた：

```
List<String> list = ... ;
list.forEach( x -> { System.out.println(x); } );
```

xのタイプは何ですか？

A. リスト<文字列>

B. リスト<キャラクター>

C. 文字列

D. char

正解: (正解を表示します)

質問: **15**

与えられた：

```
public class Main {
    public static void main(String[] args) {
        Optional<String> value = createValue();
        String str = value.orElse ("Duke");
        System.out.println(str);
    }
    static Optional<String> createValue() {
        String s = null;
        return Optional.ofNullable(s);
    }
}
```

出力は何ですか？

A. null

B. 実行時にNoSuchElementExceptionがスローされます。

C. デューク

D. 実行時にNullPointerExceptionがスローされます。

正解: (正解を表示します)

質問: **16**

与えられた：

```
1. interface Pastry {
2.     void getIngredients();
3. }
4. abstract class Cookie implements Pastry {}
5.
6. class ChocolateCookie implements Cookie {
7.     public void getIngredients() {}
8. }
9. class CoconutChocolateCookie extends ChocolateCookie {
10.    void getIngredients(int x) {}
11. }
```

どちらが正しいですか？

A. コンパイルは成功します。

B. 10行目のエラーが原因でコンパイルが失敗します。

C. 7行目のエラーが原因でコンパイルが失敗します。

D. 4行目のエラーが原因でコンパイルが失敗します。

E. 9行目のエラーが原因でコンパイルが失敗します。

F. 2行目のエラーが原因で、コンパイルが失敗します。

G. 6行目のエラーが原因でコンパイルが失敗します。

正解: **(**正解を表示します**)**

質問: **17**

与えられた：

```
class CustomType<T> {
   public <T> int count(T[] anArray, T element) {
      int count = 0;
      for(T e : anArray) {
        if (e.equals(element)) ++count;
      }
      return count;
   }
}
```

と

```
public class Test extends CustomType {
   public static void main(String[] args) {
      String[] words = ("banana","orange","apple","lemon");
      Integer[] numbers = {1,2,3,4,5};
      CustomType type = new CustomType();
      CustomType<String> stringType = new CustomType<>();
      System.out.println(stringType.count(words, "apple"));
      System.out.println(type.count(words, "apple"));
      System.out.println(type.count (numbers, 3));
   }
}
```

結果はどうなりますか？

A. 実行時にNullPointerExceptionがスローされます。

B. コンパイルは失敗します。

C. 1

ヌル

ヌル

D. 1

1

1

E. 実行時にClassCastExceptionがスローされます。

正解: **(**正解を表示します**)**

```
Console 4
Error: Could not find or load main class CustomType
Caused by: java.lang.ClassNotFoundException: CustomType
```

質問: **18**

与えられた：

/code/a/Test.java

含む：

```
package a;
import b.Best;
public class Test {
   public static void main(String[] args) {
      Best b = new Best();
   }
}
```

と

/code/b/Best.java

含む：

パッケージb;

パブリッククラスベスト{}

すべてのクラスのバイトコードを生成する有効な方法はどれですか？

A. java -cp / code a.Test

B. java /code/a/Test.java

C. javac -d / code /code/a/Test.java /code/b/Best.java

D. java /code/a/Test.java /code/b/Best.java

E. javac -d / code / code / a / Test
F. javac -d / code /code/a/Test.java
正解: **(**正解を表示します**)**

質問: **19**
与えられた：

```
public class Person {
   private String name;
   public void setName(String name) {
      String title = "Dr. ";
      name = title+name;
   }
   public String toString() {
      return name;
   }
}

and

public class Test {
   public static void main(String args[]) {
      Person p = new Person();
      p.setName("Who");
      System.out.println(p);
   }
}
```


結果はどうなりますか？
A. ドクター ・フー
B. ヌル博士
C. 実行時に例外がスローされます。
D. null
正解: **D (**コメントを発表する**)**

質問: **20**
与えられた：

```
import java.util.ArrayList;
import java.util.Arrays;
public class NewMain {
  public static void main(String[] args) {
    String[] fruitNames = { "apple", "orange",
        "grape", "lemon", "apricot", "watermelon" };
    var fruits = new ArrayList<>(Arrays.asList(fruitNames));
    fruits.sort((var a, var b) -> -a.compareTo(b));
    fruits.forEach(System.out::println);
  }
}
```

結果はどうなりますか？

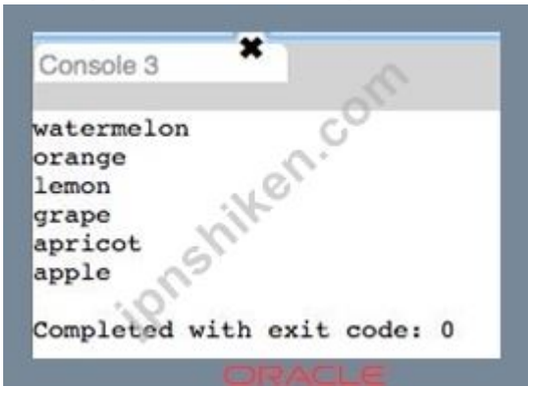
A. スイカorangelemongrapeapricotapple

B. 何もない

C. appleapricotgrapelemonorangewatermelon

D. appleorangegrapelemonapricotwatermelon

正解: **(**正解を表示します**)**



質問: **21**

与えられた：

1行目に相当するステートメントはどれですか？

A. double totalSalary = list.stream（）map e> e.getSalary（）ratio）.reduce o）.ifPresent p> p.doubleValue（）;）

B. double totalSalary = list.stream（）mapToDouble e> e.getSalary（）ratio）.sum;

C. double totalSalary = list.stream（）map Employee :: getSalary * ratio）.reduce o）.orElse 0.0）;

D. double totalSalary = list.stream（）mapToDouble e> e.getSalary（）ratio）.reduce starts、bo）;

正解: **(**正解を表示します**)**

質問: **22**
与えられた：
var Fruits = List.of（"apple"、"orange"、"banana"、"lemon"）;
文字nを含む最初の要素を調べたいと思います。どのステートメントがこれを達成しますか？
A. 文字列の結果= Fruits.stream（）filter f> f.contains（"n"））findAny（）;
B. fruits.stream（）filter f> f.contains（"n"））forEachOrdered $System.out :: print）;
C. オプション<String> result = Fruits.stream（）filter f> f.contains（"n"））findFirst（）;
D. オプション<String> result = Fruits.stream（）anyMatch f> f.contains（"n"）;
正解: **(**正解を表示します**)**

質問: **23**
与えられた：

```
public class Tester {
    public static void main(String[] args) {
        int x = 0, y = 6;
        for( ; x < y ; x++, y--) { // line 1
            if (x%2 == 0) {
                continue;
            }
            System.out.println(x+"-"+y);
        }
    }
}
```


結果はどうなりますか？

2-4

A. 0-6

B. 1-5

2-4

1-5

C. 1-5

D. 2-4

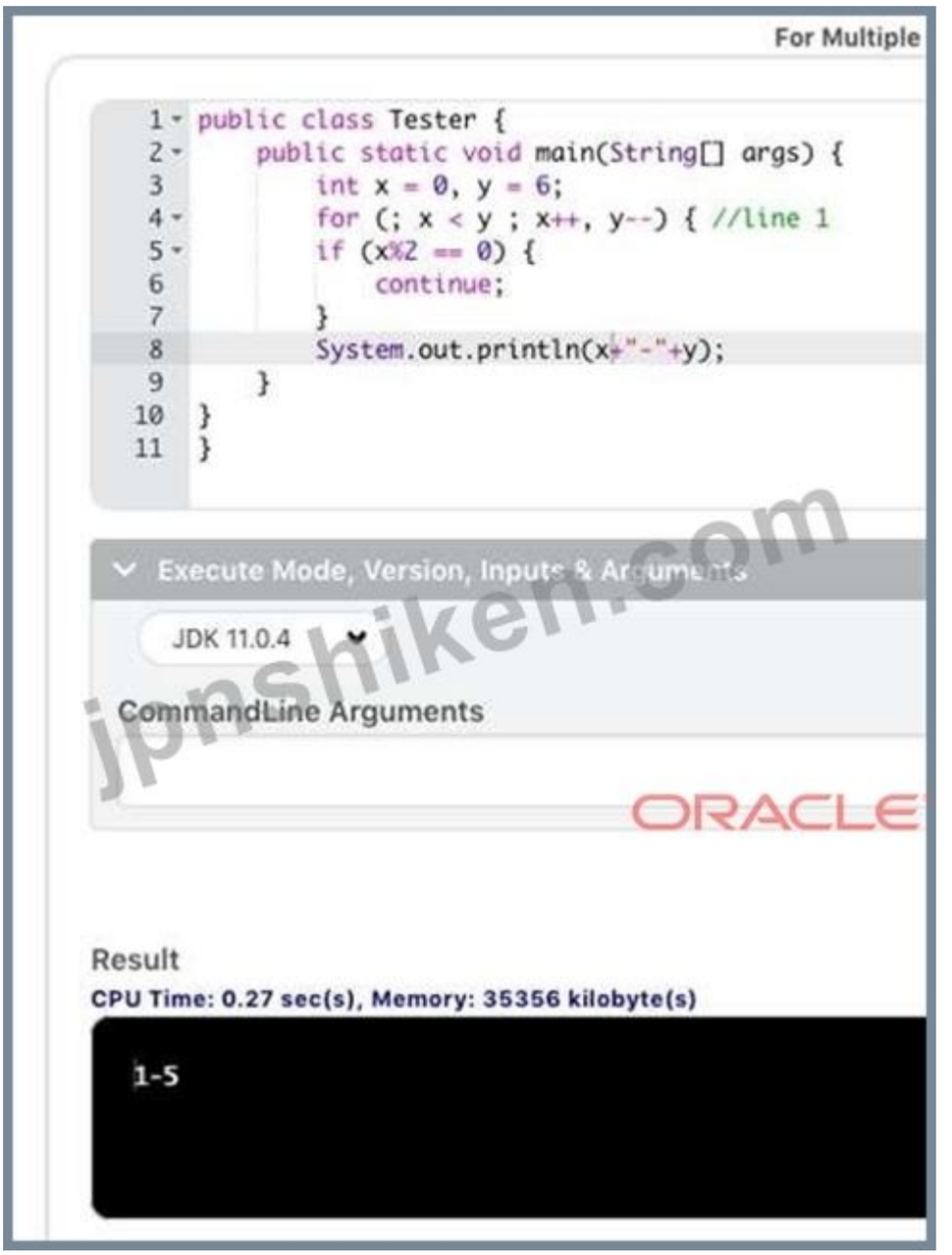
E. 1行目のエラーが原因で、コンパイルが失敗します。

0-6

F. 0-6

G. 2-4

正解: **C** (コメントを発表する)



質問: **24**
正しい3つの初期化ステートメントはどれですか？ 3つ選択してください。）
A. short sh = (short)'A';
B. 文字列contact#= " (12) 999) 283)";
C. バイトb = 10; char c = b;
D. ブール値true = (4 == 4);
E. int x = 12_34;
F. int [] [] e = {{1,1},{2,2}};
G. float x = 1.99;
正解: **(正解を表示します)**

質問: **25**
与えられた

:

```
public class Point {
    @JsonField(type=JsonField.Type.STRING, name="name")
    private String _name;

    @JsonField(type=JsonField.Type.INT)
    private int x;

    @JsonField(type=JsonField.Type.INT)
    private int y;
}
```

Pointクラスをコンパイルするjsonfieldアノテーションの正しい定義は何ですか？

A）

```
@Target(ElementType.FIELD)
@interface JsonField {
    String name() default "";
    enum Type {
        INT, STRING, BOOLEAN
    };
    Type type();
}
```

B）

```
@interface JsonField {
    String name();
    enum Type {
        INT, STRING, BOOLEAN
    };
    Type type();
}
```

C）

```
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
@interface JsonField {
    String name() default "";
    enum Type {
        INT, STRING, BOOLEAN
    };
    Type type();
```

A. オプションC

B. オプションA

C. オプションB

正解: **(**正解を表示します**)**

質問: **26**

宣言が与えられた場合：

```
@interface Resource {
    String name();
    int priority() default 0;
}
```

このコードフラグメントを調べます。
/ * Loc1 * / class ProcessOrders {...}
コードフラグメントのLoc1に適用できる2つの注釈はどれですか？ 2つ選択してください。）
A. @Resource（priority = 100）
B. @ Resource
C. @Resource（name = "Customer1"、priority = 100）
D. @Resource（priority = 0）
E. @Resource（name = "Customer1"）
正解: **(**正解を表示します**)**

質問: **27**
TripleThis.javaが与えられた場合：

```
6.  import java.util.function.*;
7.  public class TripleThis {
8.     public static void main(String[] args) {
9.        Function tripler = x -> { return (Integer) x * 3; };
10.       TripleThis.printValue(tripler, 4);
11.    }
12.    public static <T> void printValue(Function f, T num) {
13.       System.out.println(f.apply(num));
14.    }
15. }
```

TripleThis.javaをコンパイルすると、このコンパイラに警告が表示されます。
注 :TripleThis.javaは、チェックされていない、または安全でない操作を使用します。
一緒に行われた2つの置換のうち、このコンパイラの警告を削除するのはどれですか？
A. 9行目をfunction <Integer> tripler = x->-{return（Integer）X * 3;に置き換えます。}。
B. 12行目をpublic static void printValue function <Integer> f、int num）{に置き換えます。
C. 9行目をfunction <Integer>、Integer> = X-> {return（Integer）x * 3;に置き換えます。}。
D. 12行目をpublic static <T> void printValue（Function <T、T> f、T num）{、
E. 12行目をpublic static int printValue function <Integer、Integer>、f、T num {に置き換えます。
正解: **A,E (**コメントを発表する**)**

質問: **28**
与えられた：

```
public interface Builder {
    public A build(String str);
}

and

public class BuilderImpl implements Builder {
    @Override
    public B build(String str) {
        return new B(str);
    }
}
```

このコードが正しくコンパイルされると仮定すると、正しい3つのステートメントはどれですか。（3つ選択してください。）

A. Bを抽象化することはできません。
B. BはAのサブタイプです。
C. Aを抽象化することはできません。
D. Aをファイナルにすることはできません。
E. Bをファイナルにすることはできません。
F. AはBのサブタイプです。
正解: **A,B,D** (コメントを発表する)

質問: **29**
与えられた：

```
Integer[] intArray = {2, 1, 3, 4, 5};
List<Integer> list =
new ArrayList<>(Arrays.asList (intArray));
list.parallelStream()
    .forEach(e -> System.out.print(e + " "));
```

どちらが正しいですか？ （2つ選択してください。）

A. 出力は正確に2 1 345になります。
B. プログラムは1 4 2 3を出力しますが、順序は予測できません。
C. forEach（をforEachOrdered（に置き換えると、プログラムは2 1 3 4 5を出力しますが、順序は予測できません。
D. forEach（をforEachOrdered（に置き換えると、プログラムは1 2 345を出力します。
E. forEach（をforEachOrdered（に置き換えると、プログラムは2 1 345を出力します。
正解: **B,D** (コメントを発表する)

```
8  public class Secret {
9      public static void main(String[] args) {
10     Integer[] intArray = {1, 2, 3, 4, 5};
11     List<Integer> list =
12     new ArrayList<> (Arrays.asList (intArray));
13     list.parallelStream()
14     .forEachOrdered(e -> System.out.print(e + " "));
15     }
16 }
```

Result

CPU Time: 0.32 sec(s), Memory: 37040 kilobyte(s)

```
1 2 3 4 5
```

質問: **30**

与えられた：

List <String> longlist = List.of（"Hello"、"World"、"Beat"）;

List <String> shortlist = new ArrayList <>（）;

文字 "e"を含む単語の短いリストを正しく形成するコードフラグメントはどれですか？

A.
```
longList.stream()
    .filter(w -> w.indexOf('e') != -1)
    .parallel()
    .forEach(w -> shortList.add(w));
```

B.
```
longList.parallelStream()
    .filter(w -> w.indexOf('e') != -1)
    .forEach(w -> shortList.add(w));
```

C.
```
shortList = longList.stream()
    .filter(w -> w.indexOf('e') != -1)
    .parallel()
    .collect(Collectors.toList());
```

D.
```
longList.stream()
    .filter(w -> w.indexOf('e') != -1)
    .parallel()
    .collect(shortlist);
```

A. オプションD

B. オプションA

C. オプションB

D. オプションC

正解: **(**正解を表示します**)**

質問: **31**
与えられた：

```
public class Hello {
    public static void main(String[] args) {
        System.out.println(args[0]+args[1]+args[2]);
    }
}
```

コマンドを使用して実行：
java Hello "Hello World" Hello World
出力は何ですか？
A. Hello World Hello World
B. ハローワールドハローワールド
C. 実行時に例外がスローされます。
D. こんにちはWorldHelloWorld
E. HelloHello WorldHelloWorld
正解: (正解を表示します)

質問: **32**
与えられた：

```
public class FunctionalInterfaceTest {
    public static void main(String[] args) {
        List fruits = Arrays.asList("apple", "orange", "banana");
        Consumer<String> c = System.out::print;
        Consumer<String> output = c.andThen(x -> System.out.println(":" + x.toUpperCase
()));
        fruits.forEach(output);
    }
}
```

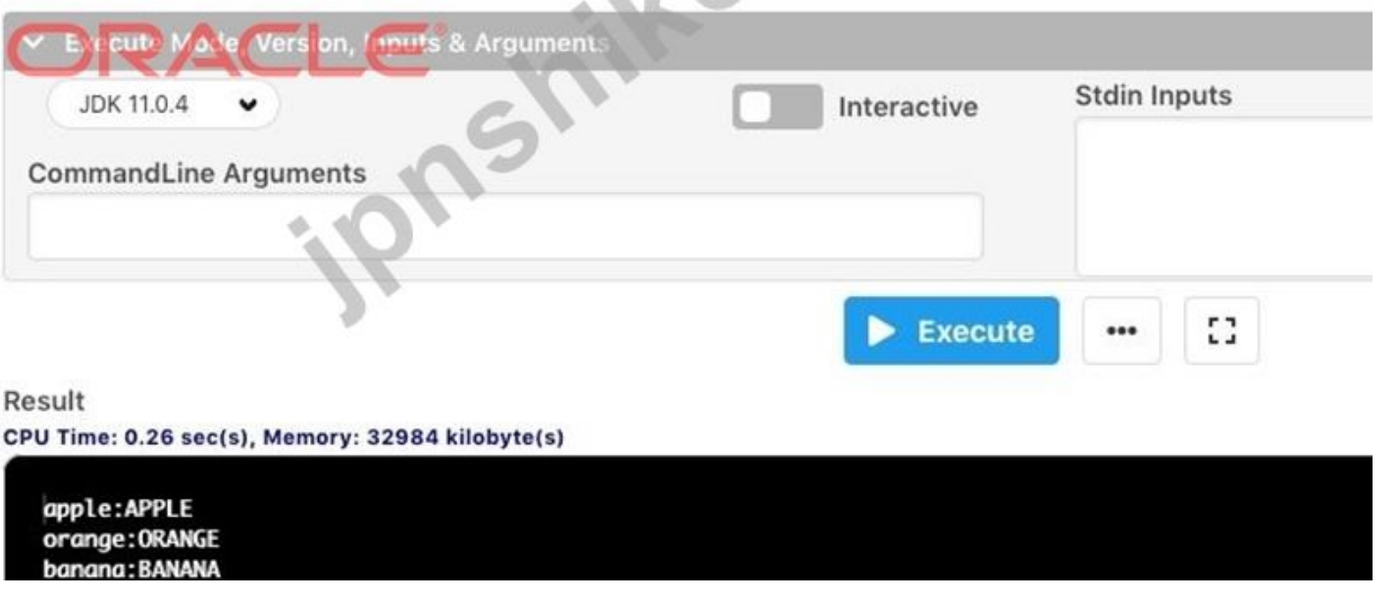
出力は何ですか？
A. : APPLE :ORANGE :BANANA
アップルオレンジバナナ
B. : APPLE :ORANGE :BANANA
C. APPLE :apple ORANGE :orange BANANA :banana
D. appleorangebanana

:APPLE :ORANGE :BANANA

E. アップル :APPLEオレンジ :ORANGEバナナ :BANANA

正解: **(**正解を表示します**)**

```
1  import java.util.*;
2  import java.io.*;
3  import java.lang.Thread;
4  import java.util.ArrayList;
5  import java.util.LinkedList;
6  import java.util.List;
7  import java.util.function.Consumer;
8
9  public class FunctionalInterfaceTest {
10  public static void main (String[] args) {
11          List fruits = Arrays.asList("apple", "orange", "banana");
12          Consumer<String> c = System.out::print;
13          Consumer<String> output = c.andThen(x -> System.out.println(":" + x.toUpperCase()));
14
15     fruits.forEach(output);
16
17   }
18 }
```



質問: **33**

クラスとモジュールの依存関係を識別するために使用される2つのコマンドはどれですか？ 2つ選択してください。）

A. java --show-module-resolution

B. jar --show-module-resolution

C. jdeps --list-deps

D. jmoddescribe

E. java Hello.java

正解: **(**正解を表示します**)**

質問: **34**

コードフラグメントが与えられた場合：

```
int x = 0;
do {
   x++;
   if (x == 1) {
      continue;
   }
   System.out.println(x);
} while(x < 1);
```

結果はどうなりますか？

A. 0

B. 01

C. 無限ループで1を出力します。

D. 1

E. プログラムは何も出力しません。

正解: **(**正解を表示します**)**

質問: **35**

このメソッド宣言を検討してください。

```
void setSessionUser(Connection conn, String user) throws SQLException {
     Statement stmt = conn.createStatement();
     String sql = <EXPRESSION>;
     stmt .execute();
}
```

A）"SETSESSIONAUTHORIZATION"+ユーザー

B）"SETSESSIONAUTHORIZATION"+stmt.enquoteIdentifier（user）

AまたはBは<EXPRESSION>の正しい代替品ですか？その理由は何ですか？

A. B、呼び出し元のコードによって提供されるすべての値を引用符で囲む必要があるため。

B. AとBは機能的に同等です。

C. A、呼び出し元のコードによって提供されたユーザーの値を正確に送信するため。

D. B。呼び出し元のコードによって提供された値を引用すると、SQLインジェクションが防止されるためです。

E. A、識別子を引用符で囲む必要がないため。

正解: **C (**コメントを発表する**)**

質問: **36**

与えられた：

```
public method foo() throws FooException {
     ...
}
```

また、throws FooException句を省略すると、コンパイルエラーが発生します。

FooExceptionについて正しい説明はどれですか。

A. fooの本体は、FooExceptionまたはそのサブクラスの1つをスローできます。
B. FooExceptionはチェックされていません。
C. fooの本体はFooExceptionのみをスローできます。
D. FooExceptionはRuntimeErrorのサブクラスです。
正解: **(**正解を表示します**)**

質問: **37**
モジュラーJDKについて正しい2つのステートメントはどれですか？ 2つ選択してください。)
A. コマンドラインからモジュールのエクスポートを構成することは可能ですが、望ましくありません。
B. Java SEプラットフォームの基本的なAPIは、java.baseモジュールにあります。
C. JDKがモジュール化されているため、APIはより積極的に非推奨になっています。
D. モジュール式JDKで実行するには、アプリケーションをモジュールとして構造化する必要があります。
正解: **(**正解を表示します**)**

質問: **38**
与えられた：

```
1. public class Main {
2.     public static void greet(String... args) {
3.         System.out.print("Hello ");
4.         for (String arg : args) {
5.             System.out.println(arg);
6.         }
7.     }
8.     public static void main(String[] args) {
9.         Main c = null;
10.        c.greet();
11.    }
12. }
```

結果はどうなりますか？
A. NullPointerExceptionが4行目でスローされます。
B. NullPointerExceptionが10行目でスローされます。
C. コンパイルエラーが発生します。
D. こんにちは
正解: **(**正解を表示します**)**

質問: **39**

与えられた：

```
public static void main(String[] args) {
   try (Reader reader1 = new FileReader("File1.txt");
        Reader reader2 = new FileReader("File2.txt");
        Reader reader3 = new FileReader("File3_txt")) {

   } catch (IOException ex) {
      Logger.getLogger(Main.class.getName()).log(Level.SEVERE, null, ex);
   }
   // Line 1
   System.out.println("Done");
}
```

実行して3つのファイルがすべて存在する場合、1行目の各リーダーの状態はどうなりますか？

A. 3つのリーダーすべてが閉じられました。

B. reader1のみが閉じられました。

C. 3つのリーダーはすべて開いたままです。

D. コンパイルは失敗します。

正解: **(**正解を表示します**)**

質問: **40**

与えられた：

```
public class Main {
  private String[] strings = {"ABCDEFGHIJKLMNOPQRSTUVWXYZ",
                              "abcdefghijklmnopqrstuvwxyz", "0123456789"};
  public void write(String filename){
    // line 1
      for (String str: strings) {
        ByteBuffer buffer = ByteBuffer.wrap(str.getBytes());
        fileChannel.write(buffer);
      }
    }catch(IOException e){
      e.printStackTrace();
    }
  }
  public static void main(String[] args) {
    Main test = new Main();
    test.write("file_to_path");
  }
}
```

1行目でFilechannelオブジェクトを取得したいとします。

どのコードフラグメントがこれを達成しますか？

A）

```
try (FileChannel fineChannel = Channels.newChannel(new FileOutputStream(filename));) {
```

B）

```
try(FileChannel fileChannel = new FileOutputStream(filename).getChannel();) {
```

C）

```
try (FileChannel fileChannel = new FileOutputStream(new FIleChannel(filename));) {
```

D）

```
try(FileChannel fileChannel = new FileChannel(new FileOutputStream(filename));) {
```

A. オプションA
B. オプションB
C. オプションD
D. オプションC
正解: **(**正解を表示します**)**

質問: **41**
与えられた：

```
package A;
class Test {
  String name;
  public Test(String name) {
    this.name = name;
  }
  public String toString() {
    return name;
  }
}
```

and

```
package B;
import A.Test;
public class Main {
  public static void main(String[] args) {
    Test test = new Test("Student");
    System.out.println(test);
  }
}
```


結果はどうなりますか？
A. 学生
B. コンパイルに失敗します。
C. java.lang.IllegalAccessExceptionがスローされます。

D. 何もない

E. null

正解: **(**正解を表示します**)**

質問: **42**

与えられた：

```
import java.util.ArrayList;
import java.util.Arrays;
public class NewMain {
  public static void main(String[] args) {
    String[] fruitNames = { "apple", "orange",
         "grape", "lemon", "apricot", "watermelon" };
    var fruits = new ArrayList<>(Arrays.asList(fruitNames));
    fruits.sort((var a, var b) -> -a.compareTo(b));
    fruits.forEach(System.out::println);
  }
}
```

結果はどうなりますか？

A. スイカorangelemongrapeapricotapple

B. 何もない

C. appleapricotgrapelemonorangewatermelon

D. appleorangegrapelemonapricotwatermelon

正解: **(**正解を表示します**)**

```
Console 3

atermelon
range
emon
rape
pricot
pple

ompleted with exit code: 0
```

質問: **43**

与えられた：

```
public class Tester {
   private int x;
   private static int y;
   public static void main(String[] args) {
      Tester t1 = new Tester();
      t1.x = 2;
      Tester.y = 3;
      Tester t2 = new Tester();
      t2.x = 4;
      t2.y = 5;
      System.out.println(t1.x+","+t1.y);
      System.out.println(t2.x+","+Tester.y);
      System.out.println(t2.x+","+t1.y);
   }
}
```

結果はどうなりますか？

A. 2,34,34,5

B. 2,34,54,5

C. 2,54,54,5

D. 2,34,54,3

正解: **(**正解を表示します**)**

質問: **44**

コードフラグメントが与えられた場合：

```
String s = "";
if (Double.parseDouble("11.00f") > 11) {
    s += 1;
}
if (1_7 == Integer.valueOf("17")) {
    s += 2;
}
if (1024 > 1023L) {
    s += 3;
}
System.out.print(s);
```

結果はどうなりますか？

A. 23

B. 12

C. 123

D. 13

正解: (正解を表示します)

質問: **45**

コードフラグメントが与えられた場合：

```
int x = 0;
while(x < 10){
      System.out.print(x++);
}
```

どの for」ループが同じ出力を生成しますか？

A.
```
int b = 0;
for( ; b < 10; ){
      System.out.print(++b);
}
```

B.
```
for(a; a < 10; a++){
      System.out.print(a);
}
```

C.
```
for(int d = 0; d < 10; ){
      System.out.print(d);
      ++d;
}
```

D.
```
for(int c = 0; ; c++){
      System.out.print(c);
      if(c == 10){
          break;
      }
}
```

A. オプションC

B. オプションA

C. オプションD
D. オプションB
正解: **A** (コメントを発表する)

質問: **46**
与えられた：

```
   private String name = "Joe Bloggs";
   public Person(String name) {
      this.name = name;
   }
   public String toString() {
      return name;
   )
}

and

public class Tester {
   public static void main(String[] args) {
      Person p1 = new Person(); // line 1
      System.out.println(p1);
   }
```


結果はどうなりますか？
A. null
B. ジョーブログ
C. 1行目のエラーが原因で、コンパイルが失敗します。
D. p1
正解: **C** (コメントを発表する)

質問: **47**

与えられた：

```
public interface EulerInterface {
    double getEulerValue();
}

public class EulerLambda {
    public static void main(String[] args) {
        EulerInterface  myEulerInterface;
        myEulerInterface = () -> "2.71828";
        System.out.println("Value of Euler = " + myEulerInterface.getEulerValue());
    }
}
```

結果はどうなりますか？

A. 実行時例外をスローします。

B. オイラーの値= "2.71828"

C. オイラーの値= 2.71828

D. コードはコンパイルされません。

正解: **(**正解を表示します**)**

質問: **48**

与えられた：

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

list1.add（"A"）;

list1.add（"B"）;

リストlist2=List.copyOf（list1）;

list2.add（"C"）;

List <List <String >> list3 = List.of（list1、list2）;

System.out.println（list3）;

結果はどうなりますか？

A. [[A、B]、[A、B]]

B. 実行時に例外がスローされます。

C. [[A、B]、[A、B、C]]

D. [[A、B、C]、[A、B、C]]

正解: **B (**コメントを発表する**)**

```
12 public class Main {
13   public static void main(String[] args) {
14
15     List<String> list1 = new ArrayList<>();
16     list1.add("A");
17     list1.add("B");
18     List list2 = List.copyOf(list1);
19     list2.add("C");
20     List<List<String>> list3 = List.of(list1, list2);
21     System.out.println(list3);
22   }
23
24   }
25
```

Execute Mode, Version, Inputs & Arguments

JDK 11.0.4 | Interactive | Stdin Inputs

CommandLine Arguments

Execute

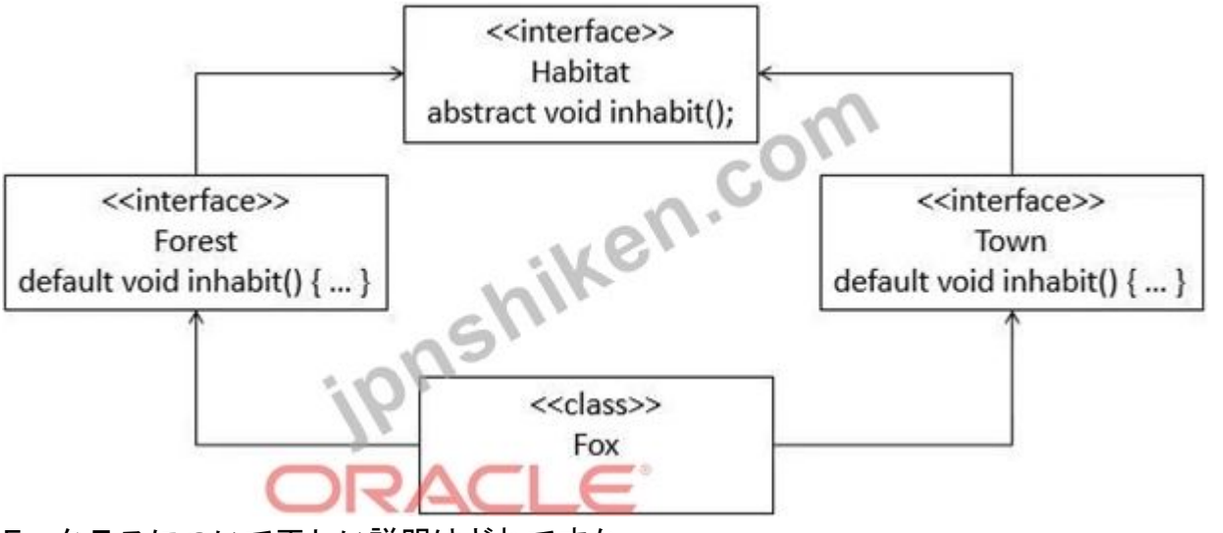
Result

CPU Time: 0.16 sec(s), Memory: 32128 kilobyte(s)

```
Exception in thread "main" java.lang.UnsupportedOperationException
    at java.base/java.util.ImmutableCollections.uoe(ImmutableCollections.java:71)
    at java.base/java.util.ImmutableCollections$AbstractImmutableCollection.add(ImmutableCollections.java:75)
    at Main.main(Main.java:19)
```


質問: **49**

与えられた：



Foxクラスについて正しい説明はどれですか。

A. Foxクラスは、ForestまたはTownインターフェースのいずれかを実装する必要がありますが、両方を実装することはできません。

B. Foxクラスは、呼び出しを試みない限り、inhabitメソッドをオーバーライドする必要はありません。

C. ForestとTownが互換性のある実装を提供している場合、Foxクラスはinhabitメソッドをオーバーライドする必要はありません。
D. Foxが実装する最初のインターフェースからのinhabitメソッドの実装が優先されます。
E. Foxクラスは、inhabitメソッドの実装を提供する必要があります。
正解: **C** (コメントを発表する)

質問: **50**
コードフラグメントが与えられた場合：

```
StringBuilder txt1 = new StringBuilder("PPQRRRSTT");
int i = 0;
a:
while (i < txt1.length()) {
   char x = txt1.charAt(i);
   int j = 0;
   i++;
   b:
   while (j < txt1.length()) {
      char y = txt1.charAt(j);
      if (i != j && y == x) {
         txt1.deleteCharAt(j);
         // line 1
      }
      j++;
   }
}
System.out.println(txt1);
```

1行目に個別に挿入された2つのステートメントのうち、このコードでPRRTを出力できるのはどれですか？
A. 続行b;
B. ブレーク;
C. 続行します;
D. j--;
E. ブレークb;
F. i-;
正解: **B** (コメントを発表する)

質問: **51**
dsがデータソースであり、EMPテーブルが適切に定義されていると仮定します。

```
try (Connection conn = ds.getConnection();
     PreparedStatement ps = conn.prepareStatement("INSERT INTO EMP VALUES(?, ?, ?)")) {
     ps.setObject(1, 101, JDBCType.INTEGER);
     ps.setObject(2, "SMITH", JDBCType.VARCHAR);
     ps.setObject(3, "HR", JDBCType.VARCHAR);
     ps.executeUpdate();
     ps.setInt(1, 102);
     ps.setString(2, "JONES");
     ps.executeUpdate();
}
```

このコードフラグメントの実行は何をしますか？
A. 1行挿入（101、'SMITH'、'HR'）
B. SQLExceptionをスローします
C. 2つの行（101、'SMITH'、'HR'）と（102、'JONES'、NULL）を挿入します

D. 2つの行（101、'SMITH'、'HR'）と（102、'JONES'、'HR'）を挿入します
正解: **A** (コメントを発表する)

質問: **52**
数値、通貨、およびパーセンテージのフォーマットに使用されるデフォルトのロケールを設定する2つのステートメントはどれですか？（2つ選択してください。）
A. Locale.setDefault（Locale.Category.FORMAT、"zh-CN"）;
B. Locale.setDefault（Locale.Category.FORMAT、Locale.CANADA_FRENCH）;
C. Locale.setDefault（"en_CA"）;
D. Locale.setDefault（"es"、Locale.US）;
E. Locale.setDefault（Locale.SIMPLIFIED_CHINESE）;
正解: **B,C** (コメントを発表する)

質問: **53**
これらの2つのクラスを考えると：

```
public class Resource {
   public Worker owner;
   public synchronized boolean claim(Worker worker) {
      if (owner == null) {
           owner = worker;
           return true;
      }
      else return false;
   }
   public synchronized void release() {
      owner = null;
   }
}
```

```
public class Worker {
   public synchronized void work(Resource... resources) {
      for (int i = 0; i < 10; i++) {
           while (!resources[0].claim(this)) { }
           while (!resources[1].claim(this)) { }
           // do work with resource
           resources[1].release();
           resources[0].release();
      }
   }
}
```

そして、このフラグメントを考えると：

```
Worker w1 = new Worker();
Worker w2 = new Worker();
Resource r1 = new Resource();
Resource r2 = new Resource();
new Thread( () -> {
     w1.work(r1, r2);
} ).start();
new Thread( () -> {
     w2.work(r2, r1);
} ).start();
```

フラグメントを説明するのはどれですか？

A. コードはコンパイルされません。

B. ライブロックの対象となります。

C. デッドロックが発生する可能性があります。

D. IllegalMonitorStateExceptionをスローします。

正解: **(**正解を表示します**)**

質問: **54**

与えられた：

```
public class X {
  private Collection collection;
  public void set(Collection collection) {
    this.collection = collection;
  }
}

and

public class Y extends X {
  public void set(Map<String,String> map) {
    super.set(map); // line 1
  }
}
```


Yクラスがコンパイルされるように1行目を置き換えることができる2行はどれですか？ 2つ選択してください。）

A. set（map.values（）;）

B. super.set（List <String> map）

C. set（map）

D. map.forEach（k、v）-> set（v）;）

E. super.set（map.values（）;）
正解: **A,E (コメントを発表する)**

質問: **55**
Customerテーブル構造が与えられた場合：
*ID番号の主キー
* NAME Text Nullable
与えられたコードフラグメント：

```
12. PreparedStatement stmt = con.prepareStatement("INSERT INTO CUSTOMER VALUES(?,?)");
13. stmt.setInt(1, 42);
14. /* Insert code here */
15. int n = stmt.executeUpdate();
```

14行目に挿入されたどのステートメントがNAME列をNULL値に設定しますか？
A. Stmt.setNull（2、null）;
B. Stmt.setNull（2 string、class）;
C. Stmt.setNull（2、java、sql。Types、VARCHAR）;
D. Stmt.setNull（2、java.lang、string）;
正解: **(正解を表示します)**

質問: **56**
与えられた：

```
    public void foo(Collection arg) {
        System.out.println("Bonjour le monde!")
    }

and

ublic class Bar extends Foo {
    public void foo(Collection arg) {
        System.out.println("Hello world!");
    }
    public void foo(List arg) {
        System.out.println("Hola Mundo!");
    }
}

and

oo f1 = new Foo();
oo f2 = new Bar();
ar b1 = new Bar();
ist<String> li = new ArrayList<>();
```


正しいのはどれですか？（3つ選択してください。）
A. f2.foo（）はBonjour le mondeを出力します！
B. b1.foo（）はHello world！を出力します。
C. f2.foo（）はHola Mundoを出力します！
D. b1.foo（）はHola Mundoを出力します！
E. f1.foo（）はHello world！を出力します。

F. f1.foo ()はHola Mundoを出力します！
G. b1.foo ()はBonjour le mondeを出力します！
H. f2.foo ()はHello world！を出力します。
I. f1.foo ()はBonjour le mondeを出力します！
正解: **(**正解を表示します**)**

質問: **57**
JDKをモジュール化する理由を説明しているのはどれですか？ 2つ選択してください。）
A. アプリケーションモジュールとJDKモジュールをリンクするカスタムランタイムを簡単に構築できます
B. 実装の詳細を公開しやすく
C. セキュリティと保守性を向上させます
D. アプリケーションの堅牢性を向上させる
E. Java言語を理解しやすい
正解: **C,D (**コメントを発表する**)**

質問: **58**
ストリームでforEachoperationの代わりにpeekoperationを使用することを選択するのはなぜですか？
A. 現在のアイテムを処理してvoidを返す
B. ストリームの最後からアイテムを削除します
C. 現在のアイテムを処理してストリームを返す
D. ストリームの先頭からアイテムを削除します
正解: **C (**コメントを発表する**)**
説明/参照 :https ://www.baeldung.com/java-streams-peek-api

質問: **59**
名前とyearsMembershipのフィールドを持つMemberclassが与えられます。これには、getterとsetter、printメソッド、およびclubMembersmembersのリストが含まれます。

```
String testName = "smith";
int testMembershipLength = 5;
long matches = clubMembers
.peek(new Consumer<Member>() {
    @Override
    public void accept(Member m) {
       m.print();
       }
    })
.filter(m -> m.getYearsMembership() >= testMembershipLength)
.map(m -> testName.compareToIgnoreCase(m))
.filter(a -> a == 0)
.count();
System.out.println(matches);
```

メソッド参照を使用するために変更できる2つのStreamメソッドはどれですか？ 2つ選択してください。）

A. filter（Member :: getYearsMembership（）> = testMembershipLength）

B. ピーク（メンバー::印刷）

C. map（testName :: compareToIgnoreCase）

D. filter（Integer :: equals（0））

正解: **A,C (**コメントを発表する**)**

質問: **60**

与えられた：

```
import java.util.List;
import java.util.Optional;
public class Test {
   public static void main(String[] args) {
      var items = List.of(new Item("A", 10),new Item("B", 2),
                          new Item("C", 12),new Item("D", 5),
                          new Item("E", 6));
      double avg = items.stream().mapToInt(i -> i.amount).average().orElse(0.0);
      Optional<Item> item = items.parallelStream()
                                 .filter(i -> i.amount < avg).findAny();
      System.out.println(item.orElseThrow());
   }
}
class Item {
   public String name; public int amount;
   public Item(String name, int amount) {
      this.name = name; this.amount = amount;
   }
   @Override
   public String toString() { return "Name: " + name + ", Amount: " + amount; }
}
```

何が本当？

A. 実行時にNoSuchElementExceptionがスローされます。

B. コンパイルは失敗します。
C. これにより、プログラムを実行するたびに同じ結果が出力されます。
D. プログラムを実行するたびに同じ結果が出力されない場合があります。
正解: (正解を表示します)

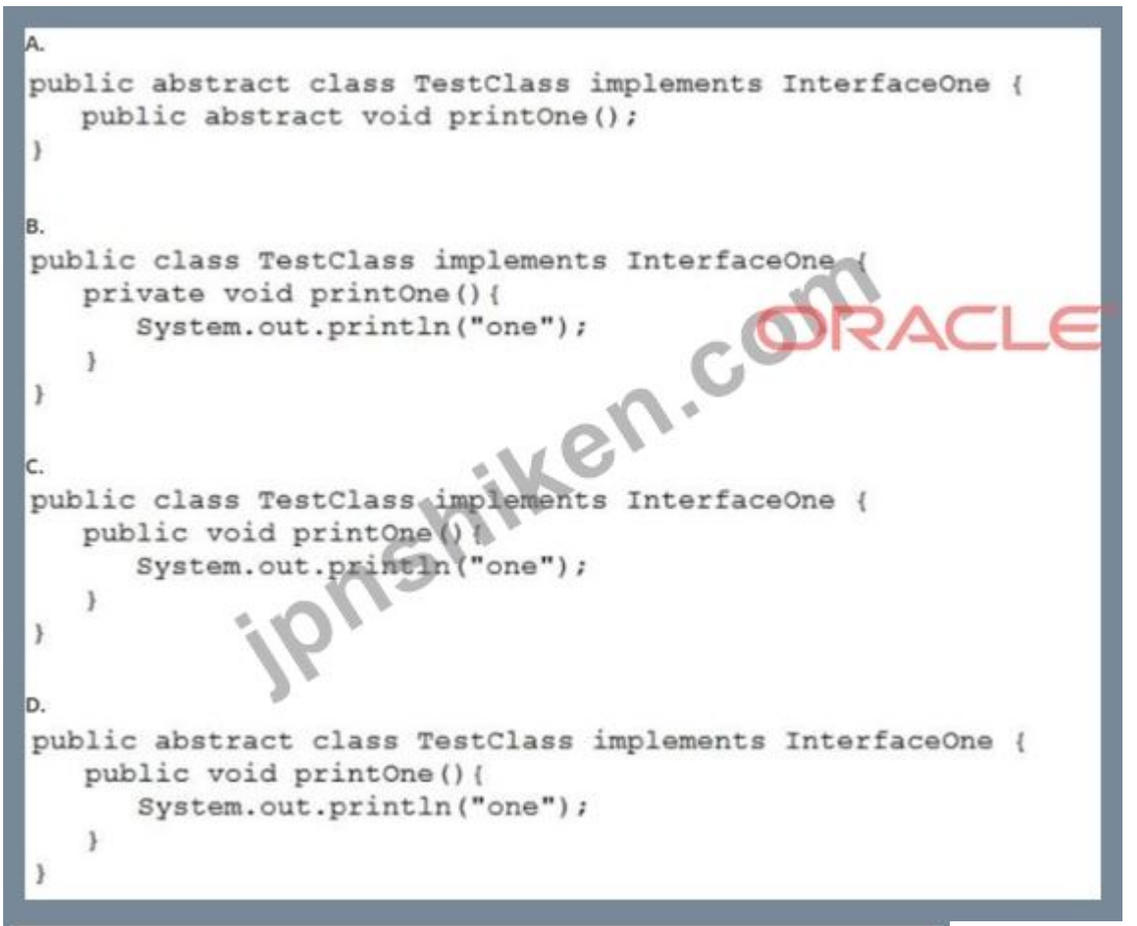
```java
public class Test {
    public static void main (String[] args) {
        var items = List.of(new Item("A", 10),new Item("B", 2),
                            new Item("C", 12), new Item("D", 5),
                            new Item("E", 6));
    double avg = items.stream().mapToInt(i -> i.amount). average().orElse(0.0);
    Optional<Item> item = items.parallelStream()
                               .filter(i -> i.amount < avg).findAny();
                               System.out.println(item.orElseThrow());
    }
}

class Item {
    public String name; public int amount;
    public Item(String name, int amount) {
        this.name = name; this.amount = amount;
    }
    @Override
    public String toString() { return "Name: "+ name + ", Amount: " + amount; }
}
```

質問: **61**

与えられた：

```
public interface InterfaceOne {
   void printOne();
}
```

printOne（を正常にオーバーライドする3つのクラスはどれですか？ 3つ選択してください。）



A. オプションE
B. オプションB
C. オプションC
D. オプションF
E. オプションA
F. オプションD
正解: **C,E,F** (コメントを発表する)

有効的な**1Z0-819**問題集はJPNTest.com提供され、**1Z0-819**試験に合格することに役に立ちます！JPNTest.comは今最新**1Z0-819**試験問題集を提供します。JPNTest.com 1Z0-819試験問題

質問: **62**
コードフラグメントが与えられた場合：

```
int x = 0;
do {
   x++;
   if (x == 1) {
      continue;
   }
   System.out.println(x);
} while(x < 1);
```

結果はどうなりますか？
A. 0
B. 01
C. 無限ループで1を出力します。
D. プログラムは何も出力しません。
E. 1
正解: **(**正解を表示します**)**

質問: **63**
与えられた：

```
public interface A {
   abstract void x();
}

and

public abstract class B /* position 1 */ {
   /* position 2 */
   public void x() { }
   public abstract void z();
}

and

public class C extends B implements A {
/* position 3 */
}
```

1つ以上のマークされた位置に挿入されたときに、クラスBとCをコンパイルできるコードはどれですか？
A. Aを実装します//位置1 @Override//位置2
B. public void z（){}//位置3
C. @Override//位置3voidx（){}//位置3@Override//位置3publicvoidz（){}//位置3
D. @Override//位置2publicvoid z（){}//位置3

正解: (正解を表示します)

質問: **64**

与えられた：

```
public interface InterfaceOne {
    void printOne();
}
```

printOne（を正常にオーバーライドする3つのクラスはどれですか？ 3つ選択してください。）

A.

```
public abstract class TestClass implements InterfaceOne {
    public abstract void printOne();
}
```

B.

```
public class TestClass implements InterfaceOne {
    private void printOne(){
        System.out.println("one");
    }
}
```

C.

```
public class TestClass implements InterfaceOne {
    public void printOne(){
        System.out.println("one");
    }
}
```

D.

```
public abstract class TestClass implements InterfaceOne {
    public void printOne(){
        System.out.println("one");
    }
}
```

E.

```
public abstract class TestClass implements InterfaceOne {
   public String printOne(){
      return "one";
   }
}
```

F.

```
public class TestClass{
   public void printOne(){
      System.out.println("one");
   }
}
```

A. オプションB
B. オプションF
C. オプションC
D. オプションD
E. オプションE
F. オプションA
正解: **C,D,F (**コメントを発表する**)**

質問: **65**
与えられた：

```
public class ExSuper extends Exception {
  private final int eCode;
  public ExSuper(int eCode, Throwable cause) {
    super(cause);
    this.eCode = eCode;
  }

  public ExSuper(int eCode, String msg, Throwable cause) {
    super(msg, cause);
    this.eCode = eCode;
  }
  public String getMessage() {
    return this.eCode+": "+super.getMessage()+"-"+this.getCause().getMessage();
   }
}

public class ExSub extends ExSuper {
  public ExSub(int eCode, String msg, Throwable cause)
    { super(eCode, msg, cause); }
}
```

およびコードフラグメント：

```
try {
  String param1 = "Oracle";
  if (param1.equalsIgnoreCase("oracle")) {
    throw new ExSub(9001, "APPLICATION ERROR-9001", new FileNotFoundException ("MyFile.txt"));
  }
    throw new ExSuper(9001, new FileNotFoundException ("MyFile.txt"));  // Line 1
} catch (ExSuper ex) {
  System.out.println(ex.getMessage());
}
```

結果はどうなりますか？

9001 :java.io.FileNotFoundException :MyFile.txt-MyFile.txt

A. 9001 :アプリケーションエラー--9001-MyFile.txt

B. コンパイルは1行目で失敗します。

C. 9001 :java.io.FileNotFoundException :MyFile.txt-MyFile.txt
9001 :アプリケーションエラー--9001-MyFile.txt

D.

正解: **B** (コメントを発表する)

質問: **66**

与えられた：

```
public class Main {

  public static void checkConfiguration(String filename)
    File file = new File(filename);
    if(!file.exists()) {
      throw new Error("Fatal Error: Configuration File, "
            + filename + ", is missing.");
    }
  }

  public static void main(String[] args) {
    checkConfiguration("App.config");
    System.out.println("Configuration is OK");
  }
```

ファイル「App.config」が見つからない場合、結果はどうなりますか？

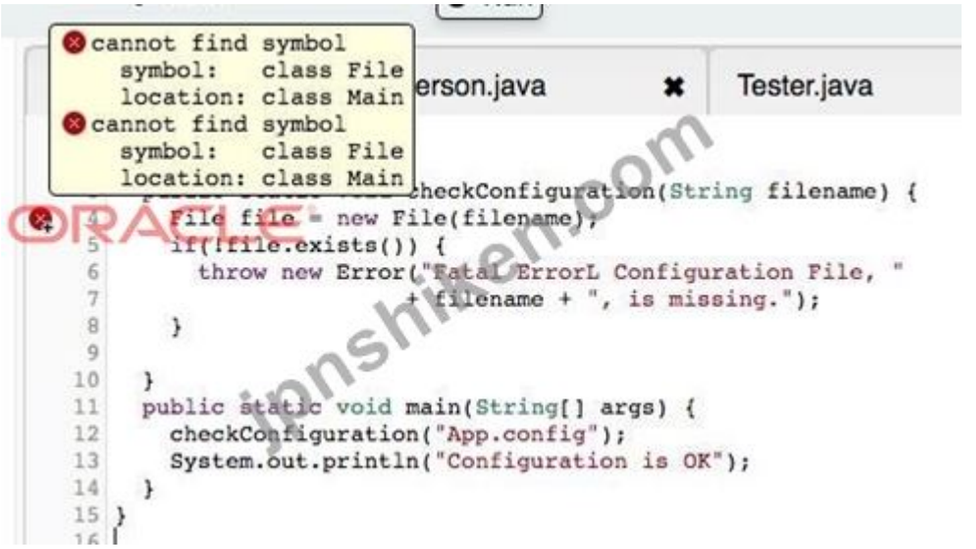
A. 設定はOKです

B. コンパイルは失敗します。

C. スレッド "main"の例外java.lang.Error :Fatalエラー：構成ファイルApp.configがありません。

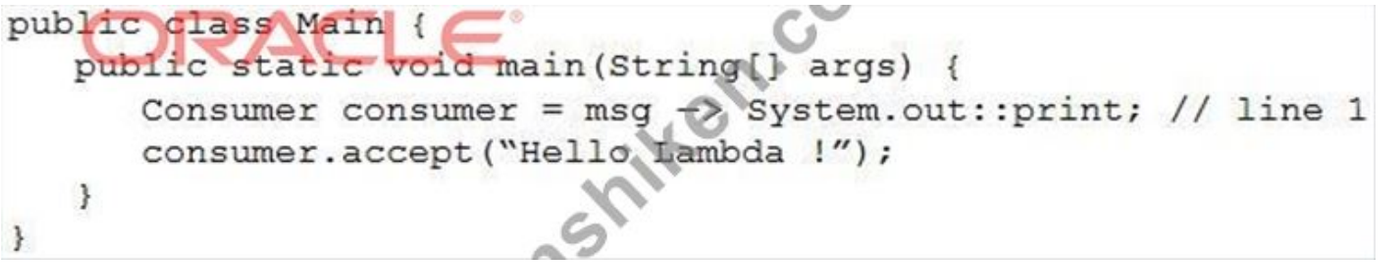
D. 何もない

正解: **B** (コメントを発表する)



質問: **67**

与えられた：



このコードにより、コンパイルエラーが発生します。

コンパイルを成功させるには、1行目にどのコードを挿入する必要がありますか？

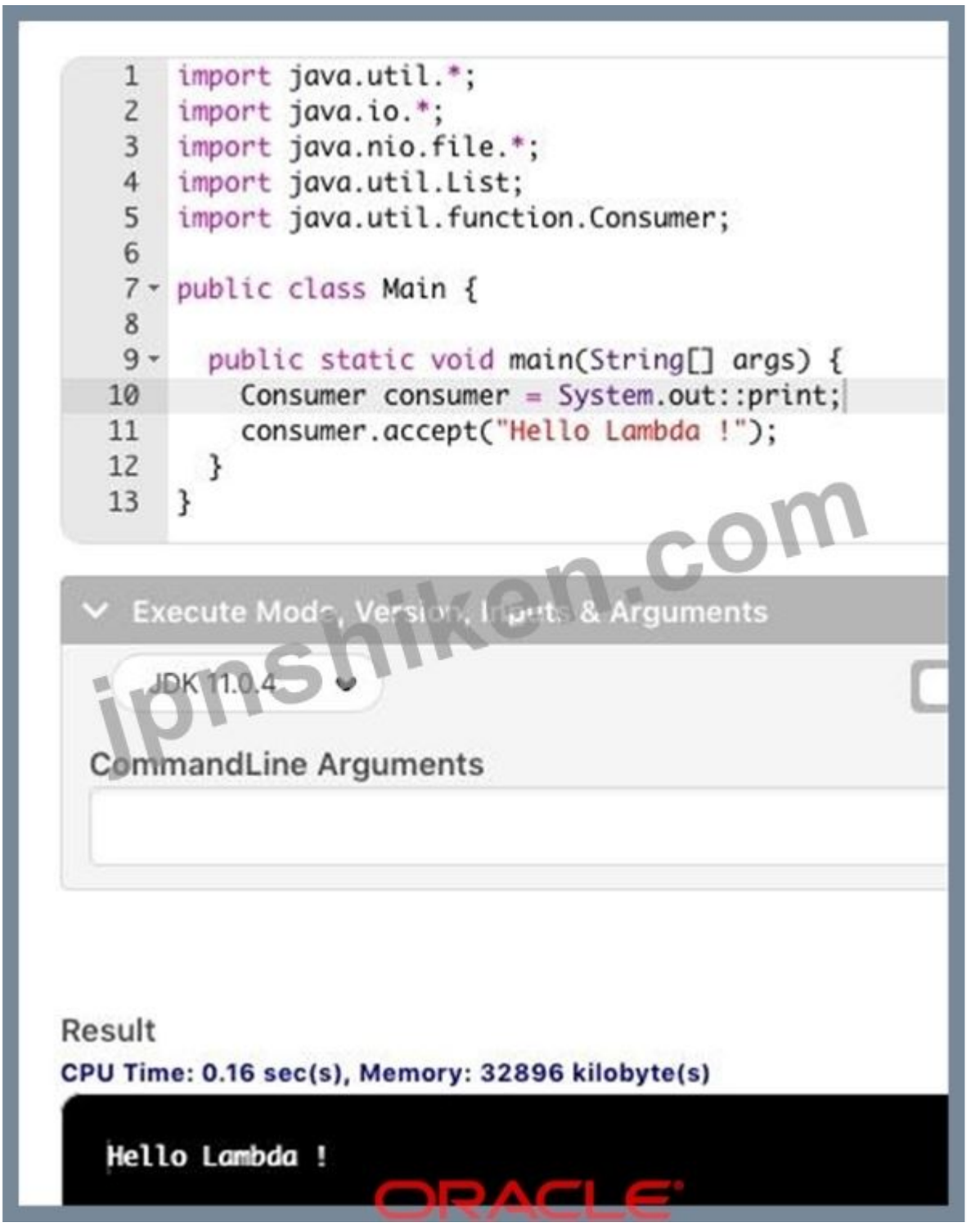
A. コンシューマーコンシューマー= msg-> {return System.out.print (msg); };

B. コンシューマーコンシューマー= var arg> {System.out.print (arg);};

C. コンシューマーコンシューマー= (String args)> System.out.print (args);

D. コンシューマーコンシューマー= System.out :: print;

正解: **(正解を表示します)**



質問: 68
与えられた：

```
public class Tester {
   public static void main(String[] args) {
      String s = "hat at store";
      int x = s.indexOf("at");
      s.substring(x + 3);
      x = s.indexOf("at");
      System.out.println(s + " " + x);
   }
}
```

結果はどうなりますか？

A. 店舗1の帽子

B. 実行時にindexOutofBoundsExceptionがスローされます。

C. 一度に0

D. 一度に1

E. 4号店の帽子

正解: **(**正解を表示します**)**

質問: **69**

与えられた：

```
public class Foo {
   private final ReentrantLock lock = new ReentrantLock();
   private State state;
   public void foo() throws Exception {
      try {
         lock.lock();
         state.mutate();
      }
      finally {
         lock.unlock();
      }
   }
}
```

Fooクラスのスレッドを安全にするために何が必要ですか？

A. ロックの宣言を静的にします。

B. ロックの宣言をfooメソッド内に移動します。

C. ロックコンストラクター呼び出しを新しいReentrantLock（true）に置き換えます。

D. 変更は必要ありません。

正解: **(**正解を表示します**)**

質問: **70**

java.util.functionパッケージのどのインターフェースがvoidリターン型を返しますか？

A. 消費者

B. 述語

C. 機能

D. サプライヤー

正解: **(**正解を表示します**)**

質問: **71**

このmodule-info.javaファイルを持つメインモジュールを持つアプリケーションがあるとします。

```
module main {
    exports country;
    uses country.CountryDetails;
}
```

どちらが正しいですか？ 2つ選択してください。）

A. エラーなしで実行するには、アプリケーションは、country.CountryDetailsの実装を提供するモジュールパスに少なくとも1つのモジュールを持っている必要があります。

B. エラーなしでコンパイルするには、アプリケーションは、country.CountryDetailsの実装を提供するモジュールソースパスに少なくとも1つのモジュールを持っている必要があります。

C. country.CountryDetailsの実装を提供するモジュールは、メインモジュールを再コンパイルせずにコンパイルおよび追加できます。

D. country.CountryDetailsの実装を提供するモジュールにはrequiresmainが必要です。module-info.javaファイルのディレクティブ。

E. country.countryDetailsの実装をメインモジュールに追加できます。

正解: **(**正解を表示します**)**

質問: **72**

与えられた：

```
public class Test {
    public static void main(String[] args) {
        int x;
        int y = 5;
        if (y > 2) {
            x = ++y;
            y = x + 7;
        } else {
            y++;
        }
        System.out.print(x + " " + y);
    }
}
```


結果はどうなりますか？

A. コンパイルエラー

B. 0 5

C. 6 13

D. 5 12

正解: **A (**コメントを発表する**)**

```
1  public class Test {
2  public static void main (String[] args) {
3      int x;
4      int y = 5;
5      if (y > 2) {
6          x = ++y;
7           y = x + 7;
8      } else {
9          y++;
variable x might not have been initialized
11     System.out.print(x + " "+y);
12     }
13 }
```


質問: **73**

与えられた：

```
public interface Copier {
  public default void print(String msg){
      System.out.println("Message from Copier: "+msg);
  }
}

and

public abstract class AbstractCopier {
  protected void print(String load){
     System.out.println("Message from Abstract Copier: "+load);
  }
}

and

public class TestImpl extends AbstractCopier implements Copier {
 public static void main(String[] args){
    TestImpl test = new TestImpl();
    test.print("Attempt00");
  }
}
```


出力は何ですか？

A. ランタイムエラーがスローされます。

B. コンパイルエラーがスローされます。

C. 抽象コピー機からのメッセージ :Attempt00

D. コピー機からのメッセージ :Attempt00

正解: **(**正解を表示します**)**

質問: **74**

モジュールcom.exampleからメインクラスcom.acme.Mainを実行するコマンドラインはどれですか？

A. java --module-path mods -m com.example / com.acme.Main
B. java --module-path mods com.example / com.acme.Main
C. java -classpath com.example.jar com.acme.Main
D. java -classpath com.example.jar -m com.example / com.acme.Main
正解: (正解を表示します)

質問: **75**
与えられた：

```
public class Tester {
   static class Person implements /* line 1 */ {
      private String name;
      Person(String name) { this.name = name; }
      /* line 2 */
   }
   public static void main(String[] args) {
      Person[] people = {new Person("Joe"),
                         new Person("Jane"),
                         new Person("John")};
      Arrays-sort(people);
      for(Person person: people) {
         System.out.println(person.name);
      }
   }
}
```


コードで次の出力を生成する必要があります。
ジョン
ジョー
ジェーン
出力を生成するには、1行目と2行目にどのコードフラグメントを挿入する必要がありますか？
A. 1行目にComparator <Person>を挿入します。
入れる
public int compare（Person person）{
person.name.compare（this.name）;を返します。
}
2行目。
B. 1行目にComparator <Person>を挿入します。
入れる
public int compare（Person p1、Person p2）{
p1.name.compare（p2.name）;を返します。
}
2行目。
C. 1行目にComparable <Person>を挿入します。
入れる

public int compare（Person p1、Person p2）{
p1.name.compare（p2.name）;を返します。
}
2行目。
D. 1行目にComparator <Person>を挿入します。
入れる
public int compareTo（Person person）{
person.name.compareTo（this.name）;を返します。
}
2行目。
正解: **(**正解を表示します**)**

質問: **76**
与えられた：

```
public class Test {
  private String[] strings;
}
```

クラスフィールド文字列をコンパイルして設定する2つのコンストラクターはどれですか？（2つ選択してください。）

```
public Test(List<String> strings) {
  this.strings = strings;
}
```

```
public Test(String... strings) {
   strings = strings;
}
```

```
public Test(String... strings) {
  this.strings = strings;
}
```

```
public Test(String strings) {
  strings = strings;
}
```

```
public Test(String[] strings) {
  this.strings = strings;
}
```

A. オプションD
B. オプションA
C. オプションB
D. オプションC
E. オプションE
正解: **(**正解を表示します**)**

質問: **77**

与えられた：

```
public class Test {
   public static void doThings() throws GeneralException {
      try {
         throw new RuntimeException("Someting happened");
      } catch (Exception e) {
         throw new SpecificException(e.getMessage());
      }
   }
   public static void main(String args[]) {
      try{
         Test.doThings();
      } catch (Exception e) {
         System.out.println(e.getMessage());
      }
   }
}
class GeneralException /* line 1 */ {
   public GeneralException(String s) { super(s); }
}
class SpecificException /* line 2 */ {
   public SpecificException(String s) { super(s); }
}
```


コードが何かを印刷できるようにするには、どのオプションを選択する必要がありますか？

A. 1行目にextendsGeneralExceptionを追加します。
2行目にextendsExceptionを追加します。

B. 1行目にextendsSpecificExceptionを追加します。
2行目にGeneralExceptionを追加します。

C. 1行目にextendsExceptionを追加します。
2行目にextendsExceptionを追加します。

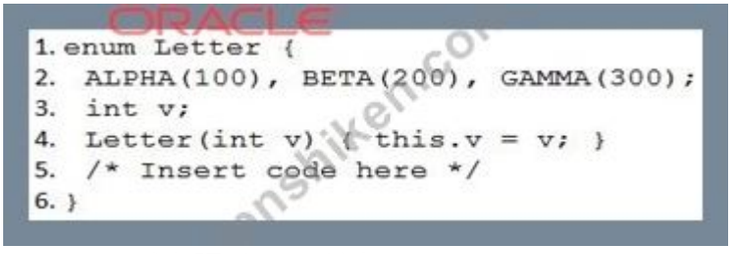
D. 1行目にextendsExceptionを追加します。
2行目にGeneralExceptionを追加します。

正解: **D** (コメントを発表する)

```java
import java.util.*;
import java.io.*;
import java.lang.Thread;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;

public class Test {

  public static void doThings() throws GeneralException {
    try{
      throw new RuntimeException("Something happened");
    } catch (Exception e) {
      throw new SpecificException (e.getMessage());

    }
  }

  public static void main(String args[]) {
    try{
      Test.doThings();
    }catch (Exception e) {
        System.out.println(e.getMessage());
      }
    }
    class GeneralException extends Exception {
      public GeneralException(String s) { super(s); }
    }
  class SpecificException extends GeneralException {
    public SpecificException(String s) { super(s);}
  }
}
```


質問: **78**

この列挙型宣言を考えると：



```
1. enum Letter {
2.  ALPHA(100), BETA(200), GAMMA(300);
3.  int v;
4.  Letter(int v) { this.v = v; }
5.  /* Insert code here */
6. }
```

このコードを調べてください：

System.out.println（Letter.values（）[1]）;

このコードが200を出力するには、5行目にどのコードを記述する必要がありますか？

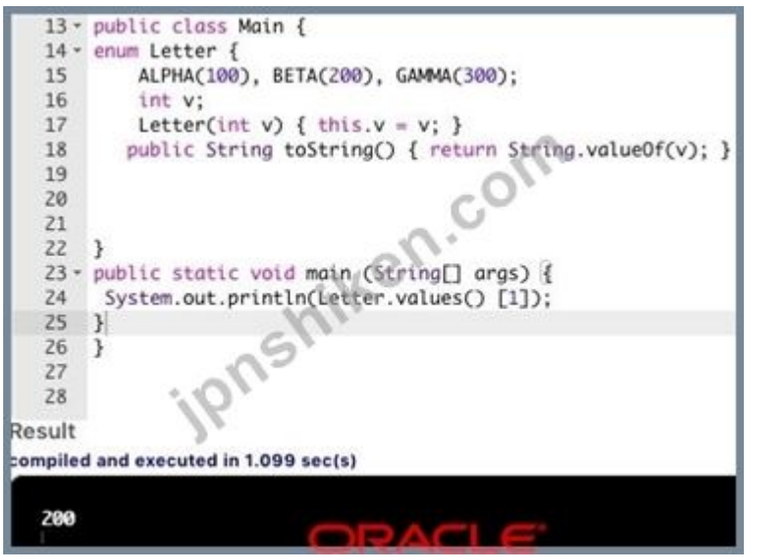
A. public String toString（）{return String.valueOf（ALPHA.v）; }

B. public String toString（）{return String.valueOf（Letter.values（）[1]）; }

C. public String toString（）{return String.valueOf（v）; }

D. String toString（）{return "200"; }

正解: **C (**コメントを発表する**)**



質問: **79**

与えられた：

```
1.   void insertionSort(int values[]) {
2.        int n = values.length;
3.        for (int j = 1; j < n; j++) {
4.             int tmp = values[j];
5.             int i = j - 1;
6.             while ( (i > -1) && (values[i] > tmp) ) {
7.                values[i + 1] = values[i];
8.                  i--;
9.             }
10.            values[i + 1] = tmp;
11.       }
12.  }
```

その後、assert i<0||を挿入できます。values [i] <= values [i + 1]; 値の配列が部分的にソートされていることを確認するには？

A. 8行目以降

B. 6行目以降

C. 5行目以降

D. 10行目以降

正解: **(**正解を表示します**)**

```
1  import java.util.*;
2  import java.io.*;
3  import java.lang.Thread;
4  import java.util.ArrayList;
5  import java.util.LinkedList;
6  import java.util.List;
7  import java.util.function.Consumer;
8  import java.util.stream.Stream;
9  import java.util.stream.IntStream;
10
11
12 public class Main {
13
14
15     void insertionSort (int values[]) {
16       int n = values.length;
17         for (int j = 1; j < n; j++) {
18           int tmp = values[j];
19
20           int i = j - 1;
21           assert i < 0 || values[i] <= values[i + 1]
22           while ((i > 1) && (values[i] > tmp) ) {
23             values[i + 1] = values[i];
24             i--;
25
26           }
27           values[i + 1] = tmp;
28
29         }
30     }
31 }
```

質問: **80**

与えられた：

```
package a;
public abstract class Animal {
  protected abstract void walk();
}
package b;
public abstract class Human extends Animal {
  // line 1
}
```

1行目に挿入された2行のうち、このコードをコンパイルできるのはどれですか？ 2つ選択してください。）

A. public abstract void walk（）;

B. 保護されたvoid walk（）{}

C. abstract void walk（）;

D. private void walk（）{}

E. void walk（）{}

正解: (正解を表示します)

質問: **81**
与えられた：

```
public method foo() throws FooException {
    ...
}
```

また、throws FooException句を省略すると、コンパイルエラーが発生します。
FooExceptionについて正しい説明はどれですか。
A. FooExceptionはチェックされていません。
B. FooExceptionはRuntimeErrorのサブクラスです。
C. fooの本体はFooExceptionのみをスローできます。
D. fooの本体は、FooExceptionまたはそのサブクラスの1つをスローできます。
正解: (正解を表示します)

質問: **82**
適切なJDBCURLはどれですか？
A. jdbc :mysql :// localhost :3306 / database
B. http :// localhost mysql.jdbc :3306 / database
C. http://localhost.mysql.com:3306/database
D. jdbe.mysql.com :// localhost :3306 / database
正解: **A** (コメントを発表する)

質問: **83**
与えられた：
/code/a/Test.java
含む：

```
package a;
import b.Best;
public class Test {
   public static void main(String[] args) {
      Best b = new Best();
   }
}
```

と
/code/b/Best.java
含む：
パッケージb;
パブリッククラスベスト{}
すべてのクラスのバイトコードを生成する有効な方法はどれですか？
A. javac -d / code / code / a / Test
B. javac -d / code /code/a/Test.java /code/b/Best.java
C. java -cp / code a.Test

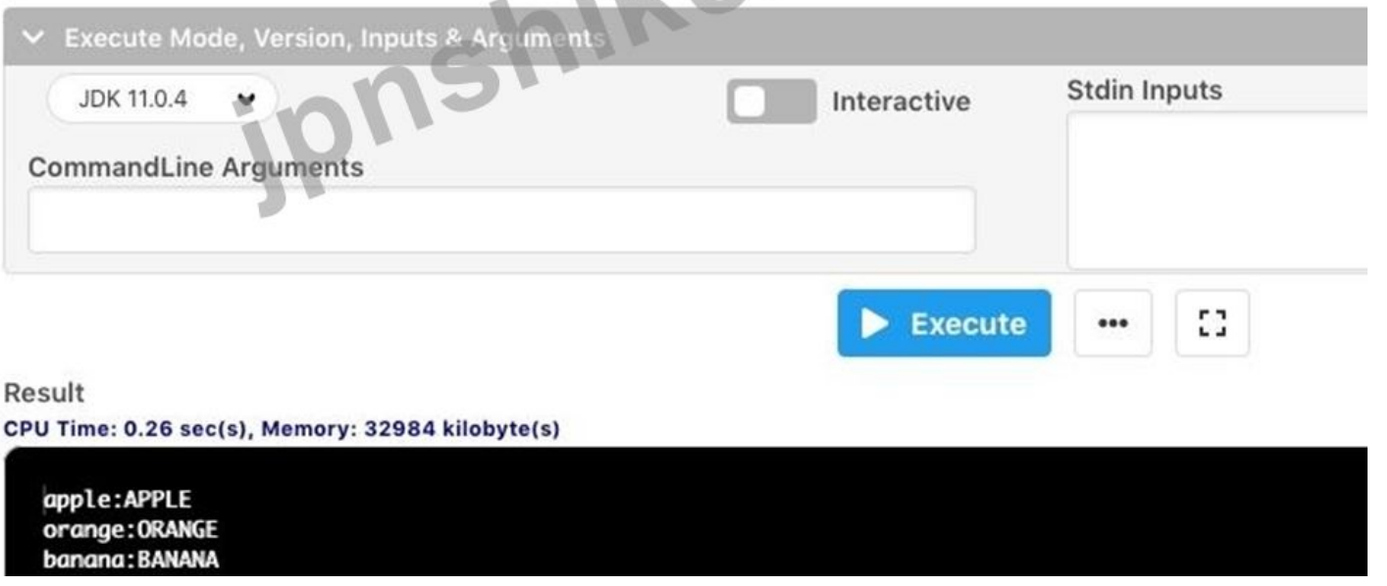
D. java /code/a/Test.java /code/b/Best.java
E. javac -d / code /code/a/Test.java
F. java /code/a/Test.java
正解: **B (**コメントを発表する**)**

質問: **84**
与えられた：

```
public class FunctionalInterfaceTest {
   public static void main(String[] args) {
      List fruits = Arrays.asList("apple", "orange", "banana");
      Consumer<String> c = System.out::print;
      Consumer<String> output = c.andThen(x -> System.out.println(":" + x.toUpperCase
()));
      fruits.forEach(output);
   }
}
```


出力は何ですか？
A. : APPLE :ORANGE :BANANA
アップルオレンジバナナ
B. : APPLE :ORANGE :BANANA
C. APPLE :apple ORANGE :orange BANANA :banana
D. appleorangebanana
:APPLE :ORANGE :BANANA
E. アップル :APPLEオレンジ :ORANGEバナナ :BANANA
正解: **(**正解を表示します**)**

```java
import java.util.*;
import java.io.*;
import java.lang.Thread;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import java.util.function.Consumer;

public class FunctionalInterfaceTest {
 public static void main (String[] args) {
        List fruits = Arrays.asList("apple", "orange", "banana");
        Consumer<String> c = System.out::print;
        Consumer<String> output = c.andThen(x -> System.out.println(":" + x.toUpperCase()));

   fruits.forEach(output);

  }
}
```



質問: **85**
入力を安全に検証する2つはどれですか？ 2つ選択してください。）
A. 値の数値範囲チェックをデータベースに委任します。
B. 信頼できるドメイン固有のライブラリを使用して入力を検証します。
C. 必要に応じて入力値を変更し、検証に合格します。
D. 有効な文字と入力値のみを受け入れます。
E. 入力がすでに検証されていると仮定します。
正解: **(正解を表示します)**

質問: **86**
tryブロックについて正しい2つのステートメントはどれですか？ 2つ選択してください。）
A. finallyブロックは、tryブロックまたはcatchブロックの直後に配置する必要があります。
B. tryブロックは複数のcatchブロックを持つことができます。

C. try-with-resourcesステートメントのfinallyブロックは、宣言されたリソースが閉じられる前に実行されます。
D. tryブロックにはcatchブロックとfinallyブロックが必要です。
E. キャッチブロックは、一般的な例外タイプから特定の例外タイプの順に並べる必要があります。
正解: **(**正解を表示します**)**

質問: **87**
与えられた：

```
public class MethodTest {
  // line 1
}
```

1行目に個別に挿入した場合、正しい2つのメソッド実装はどれですか？ 2つ選択してください。）

```
.
public boolean methodD(int x) {
    return x > 0;
}

.
public String methodB() {
    System.out.println("methodB");
}

.
public char methodE (String msg) {
    return msg;
}

.
public void methodC(int x) {
    return ++x;
}

.
public void methodA() {
    System.out.println("methodA");
}
```


A. オプションC
B. オプションD
C. オプションA
D. オプションE
E. オプションB
正解: **(**正解を表示します**)**

質問: **88**
与えられた：

```
public interface EulerInterface {
    double getEulerValue();
}

public class EulerLambda {
    public static void main(String[] args) {
        EulerInterface myEulerInterface;
        myEulerInterface = () -> "2.71828";
        System.out.println("Value of Euler = " + myEulerInterface.getEulerValue());
    }
}
```

結果はどうなりますか？

A. オイラーの値= 2.71828

B. コードはコンパイルされません。

C. 実行時例外をスローします。

D. オイラーの値= "2.71828"

正解: **(**正解を表示します**)**

質問: **89**

与えられた：

```
public class Sportscar extends Automobile{
  private float turbo;
   ....
  public void setTurbo (float turbo){
     this.turbo = turbo;
  }
}
```

スポーツカークラスについて何が知られていますか？

A. Sportscarクラスは、Automobileクラスよりも多くの機能を備えたスーパークラスです。

B. Sportscarクラスは、スーパークラスAutomobileからsetTurboメソッドを継承します。

C. Sportscarサブクラスは、スーパークラスAutomobileのsetTurboメソッドをオーバーライドできません。

D. SportscarクラスはAutomobileのサブクラスであり、そのメソッドを継承します。

正解: **D (**コメントを発表する**)**

質問: **90**

Javaクラスファイルの構造で可能なステートメントの正しい順序はどれですか？

A. インポート、クラス、パッケージ

B. インポート、パッケージ、クラス

C. パッケージ、インポート、クラス

D. クラス、パッケージ、インポート

E. パッケージ、クラス、インポート

正解: **(**正解を表示します**)**

質問: **91**

コードフラグメントが与えられた場合：
パスcurrentFile=Paths.get（"/ strike / Exam / temp.txt"）;
パスoutputFile=パスget（"/ strike / Exam / new.txt"）;
パスディレクトリ=Paths.get（"/ strike /"）;
Files.copy（currentFile、outputFile）;
Files.copy（outputFile、directory）;
Files.delete（outputFile）;
/scratch/exam/temp.txtファイルが存在します。/scratch/exam/new.txtファイルと/scratch/new.txtファイルは存在しません。
結果はどうなりますか？
A. /scratch/exam/new.txtと/scratch/new.txtが削除されます。
B. プログラムはFileaAlreadyExistsExceptionをスローします。
C. プログラムはNoSuchFileExceptionをスローします。
D. /scratch/exam/new.txtのコピーが/scratchディレクトリに存在し、/scratch/exam/new.txtが削除されます。
正解: **C** (コメントを発表する)

質問: **92**
与えられた：

```
package test;
import java.time.*;
public class Diary {
    private LocalDate now = LocalDate.now();
    public LocalDate getDate() {
        return now;
    }
}
```

and

```
package test;
public class Tester {
    public static void main(String[] args) {
        Diary d = new Diary();
        System.out.println(d.getDate());
    }
}
```

正しい説明はどれですか？

A. パッケージjava.timeのすべてのクラス。クラスDiaryにロードされます。

B. クラステスターはjava.time.LocalDateをインポートする必要はありません。これは、パッケージテストのメンバーにすでに表示されているためです。

C. java.timeパッケージのLocalDateクラスのみがロードされます。

D. テスターはコンパイルするためにjava.time.LocalDateをインポートする必要があります。

正解: **(**正解を表示します**)**

質問: **93**

与えられた：

```
1.  void insertionSort(int values[]) {
2.      int n = values.length;
3.      for (int j = 1; j < n; j++) {
4.          int tmp = values[j];
5.          int i = j - 1;
6.          while ( (i > -1) && (values[i] > tmp) ) {
7.              values[i + 1] = values[i];
8.              i--;
9.          }
10.         values[i + 1] = tmp;
11.     }
12. }
```

その後、assert i<0||を挿入できます。values [i] <= values [i + 1]; 値の配列が部分的にソートされていることを確認するには？

A. 8行目以降

B. 6行目以降

C. 5行目以降

D. 10行目以降

正解: **(**正解を表示します**)**

```
1  import java.util.*;
2  import java.io.*;
3  import java.lang.Thread;
4  import java.util.ArrayList;
5  import java.util.LinkedList;
6  import java.util.List;
7  import java.util.function.Consumer;
8  import java.util.stream.Stream;
9  import java.util.stream.IntStream;
10
11
12 public class Main {
13
14
15     void insertionSort (int values[]) {
16       int n = values.length;
17       for (int j = 1; j < n; j++) {
18         int tmp = values[j];
19
20         int i = j - 1;
21         assert i < 0 || values[i] <= values[i + 1];
22         while ((i > 1) && (values[i] > tmp) ) {
23           values[i + 1] = values[i];
24           i--;
25
26         }
27         values[i + 1] = tmp;
28
29       }
30     }
31   }
```

質問: **94**

与えられた：

```
1.    public class Secret {
2.          String[] names;
3.          public Secret(String[] names) {
4.                this.names = names;
5.          }
6.          public String[] getNames() {
7.                return names;
8.          }
9.    }
```

Java SEセキュリティガイドラインを実装する3つのアクションはどれですか？ 3つ選択してください。）

A. 7行目を変更してnames.clone（）を返します。

B. 4行目をthis.names = names.clone（）に変更します。

C. 2行目を保護された揮発性のString []名に変更します;。

D. 3行目をprivate Secret $tring [] names){に変更します。
E. getNames（）メソッド名をget $ Names（）に変更します。
F. 2行目をprivate final String [] names;に変更します。
G. 6行目をpublicsynchronized String [] getNames（）{に変更します。
正解: **(**正解を表示します**)**

質問: **95**
与えられた：

```
public static void main(String[] args)
   char letter = 'b';
   int i = 0;
   switch(letter) {
      case 'a':
         i++;
         break;
      case 'b':
         i++;
      case 'c' | 'd': // line 1
         i++;
      case 'e':
         i++;
         break;
      case 'f':
         i++;
         break;
      default:
         System.out.print(letter);
   }
   System.out.println(i);
}
```

結果はどうなりますか？
A. b1
B. 2
C. b2
D. 1
E. b3
F. 3
G. 1行目のエラーが原因で、コンパイルが失敗します。
正解: **(**正解を表示します**)**

Result
CPU Time: 0.23 sec(s), Memory: 32708 kilobyte(s)

```
3
```

質問: **96**

書店の売り上げは、顧客の名前と購入した本が入力された販売オブジェクトのリストで表されます。
パブリッククラスセール{
プライベート文字列の顧客。
プライベートリスト<Book>アイテム;
//コンストラクター、セッター、ゲッターは表示されません
}
パブリッククラスブック{
プライベート文字列名。
プライベートダブル価格;
//コンストラクター、セッター、ゲッターは表示されません
}
Saleオブジェクトのリストが与えられた場合、どのコードフラグメントが各顧客の総売上のリストを昇順で作成しますか？

A.
```
List<String> totalByUser = tList.stream()
        .collect(flatMapping(t -> t.getItems().stream(),
                 groupingBy(Sale::getCustomer,
                 summingDouble(Book::getPrice))))
        .entrySet().stream()
        .sorted(Comparator.comparing(Entry::getValue))
        .collect(mapping(e -> e.getKey() + ":" + e.getValue(),toList()));
```

B.
```
List<String> totalByUser = tList.stream()
      .collect(groupingBy(Sale::getCustomer,
               flatMapping(t -> t.getItems().stream(),
               summingDouble(Book::getPrice))))
      .sorted(Comparator. comparing (Entry::getValue))
      .collect(mapping(e -> e.getKey() + ":" + e.getValue(),toList()));
```

C.
```
List<String> totalByUser = tList.stream()
      .collect(groupingBy(Sale::getCustomer,
               flatMapping(t -> t.getItems().stream(),
               summingDouble(Book::getPrice))))
      .entrySet().stream()
      .sorted(Comparator.comparing(Entry::getValue))
      .collect(mapping(e -> e.getKey() + ":" + e.getValue(),toList()));
```

D.
```
List<String> totalByUser = tList.stream()
      .collect(flatMapping(t -> t.getItems().stream(),
               groupingBy(Sale::getCustomer,
               summingDouble(Book::getPrice))))
      .sorted(Comparator.comparing (Entry::getValue))
      .collect(mapping(e -> e.getKey() + ":" + e.getValue(),toList()));
```

A. オプションC
B. オプションB
C. オプションD
D. オプションA
正解: **(**正解を表示します**)**

質問: **97**
与えられた：

```
public interface A {
   abstract void x();
}
```

and

```
public abstract class B /* position 1 */ {
   /* position 2 */
   public void x() { }
   public abstract void z();
}
```

and

```
public class C extends B implements A {
/* position 3 */
}
```

1つ以上のマークされた位置に挿入されたときに、クラスBとCをコンパイルできるコードはどれですか？

A. Aを実装します//位置1 @Override//位置2

B. public void z（){}//位置3

C. @Override // position 3void x（){} // position 3 @ Override // position 3public void z（){} // position 3

D. @Override//位置2publicvoid z（){}//位置3

正解: **(**正解を表示します**)**

質問: **98**

与えられた：

```
public class Main {
   public static void main(String[] args) {
      int i = 1;
      for(String s : args) {
         System.out.println((i++) + ") " + s);
      }
   }
}
```

このコマンドで実行：

javaメイン1、2、3

このクラスの出力は何ですか？

A. コンパイルは失敗します。

B. 1）1つ

C. 1）1）2）23）3

D. java.lang.ArrayIndexOutOfBoundsExceptionがスローされます。

E. 何もない

正解: **(**正解を表示します**)**

質問: **99**

コードフラグメントが与えられた場合：

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

  var symbols = List.of("USD", "GBP", "EUR", "CNY");
  var exchangeRate = List.of(1.0, 1.3255, 1.1969, 0.1558094);

  var map1 =
    IntStream.range(0, Math.min(symbols.size(), exchangeRate.size()))
    .boxed ()
    .collect(Collectors.toMap(i -> symbols.get(i), i ->
                        1.0 / exchangeRate.get(i))) ;

  var map2 = map1.entrySet().stream()
    .sorted(Map.Entry.comparingByKey())
    .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue,
        (oldValue, newValue) -> oldValue, LinkedHashMap::new));
  map2.forEach((var k, var v)->System.out.printf("%s -> %.2f\n",k, v));

}
```


結果はどうなりますか？

EUR-> 0.84

A. GBP-> 0.75

USD-> 1.00

CNY-> 6.42

B. コンパイルは失敗します。

CNY-> 6.42

C. EUR-> 0.84

英ポンド->0. 75

USD-> 1.00

USD-> 1.00

D. GBP-> 0.75

EUR-> 0.84

CNY-> 6.42

正解: **(**正解を表示します**)**

Result

CPU Time: sec(s), Memory: kilobyte(s) compiled and executed in 0

```
/ques.java:15: error: cannot find symbol
    IntStream.range(0, Math.min(symbols.size(), exchangeRate.size()))
                               ^
  symbol:   variable symbols
  location: class ques
/ques.java:17: error: cannot find symbol
      .collect(Collectors.toMap(i -> symbols.get(i), i ->
                                ^
  symbol:   variable symbols
  location: class ques
/ques.java:18: error: incompatible types: Object cannot be converted to int
                          1.0 / exchangeRate.get(i)));
                                                 ^
/ques.java:22: error: incompatible types: cannot infer type-variable(s) T,K#1,U,M,K#2,V#1
      .collect(Collectors.toMap(Map.Entry::getKey, Map.Entry::getValue,
                                ^
    (argument mismatch; invalid method reference
      method getKey in interface Entry<K#3,V#2> cannot be applied to given types
        required: no arguments
        found: Object
        reason: actual and formal argument lists differ in length)
  where T,K#1,U,M,K#2,V#1,K#3,V#2 are type-variables:
    T extends Object declared in method <T,K#1,U,M>toMap(Function<? super T,? extends K#1>,Function<? super T,? extends U>,BinaryOperator<U>,Supplier<M>)
    K#1 extends Object declared in method <T,K#1,U,M>toMap(Function<? super T,? extends K#1>,Function<? super T,? extends U>,BinaryOperator<U>,Supplier<M>)
    U extends Object declared in method <T,K#1,U,M>toMap(Function<? super T,? extends K#1>,Function<? super T,? extends U>,BinaryOperator<U>,Supplier<M>)
    M extends Map<K#1,U> declared in method <T,K#1,U,M>toMap(Function<? super T,? extends K#1>,Function<? super T,? extends U>,BinaryOperator<U>,Supplier<M>)
    K#2 extends Object declared in class LinkedHashMap
    V#1 extends Object declared in class LinkedHashMap
    K#3 extends Object declared in interface Entry
    V#2 extends Object declared in interface Entry
Note: Some messages have been simplified; recompile with -Xdiags:verbose to get full output
4 errors
```

質問: **100**

与えられた：

```
public class Main {
    class Student {                                    // line 1
        String classname;
        Student(String classname) {                    // line 2
            this.classname = classname;
        }
    }
    public static void main(String[] args) {
        var student = new Student("Biology");          // line 3
    }
}
```

Mainクラスをコンパイルする2つの独立した変更はどれですか？ 2つ選択してください。）

A. Studentクラス宣言全体を別のJavaファイルStudent.javaに移動します。

B. 2行目をpublic Student $String classname）に変更します。

C. 1行目をパブリッククラスのStudent{に変更します。

D. 3行目をStudent student = new Student（"Biology"）;に変更します。

E. 1行目を静的クラスStudent{に変更します。

正解: **(**正解を表示します**)**

```java
import java.util.*;
import java.io.*;
import java.lang.Thread;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import java.util.function.Consumer;
import java.util.stream.Stream;
import java.util.stream.IntStream;
import java.util.Optional;


public class Main {
    class Student {
        String classname;
        public Student (String classname) {
            this.classname = classname;
        }

            }
            public static void main (String[] args) {
                var student = new Student ("Biology");
            }
}
```

質問: **101**

与えられた：

```java
import java.time.LocalDate;
import static java.time.DayOfWeek.*;
public class Main {
  public static void main(String[] args) {
  var today = LocalDate.now().with(TUESDAY).getDayOfWeek();
    switch(today) {
      case SUNDAY:
      case SATURDAY:
        System.out.println("Weekend");
        break;
      case MONDAY:  FRIDAY:
        System.out.println("Working");
      default:
        System.out.println("Unknown");
    }
  }
}
```

結果はどうなりますか？

A. WorkingUnknown

B. 不明

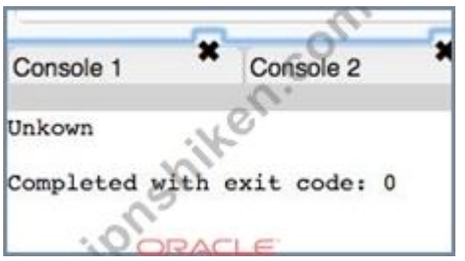
C. 火曜日不明

D. コンパイルは失敗します。

E. 火曜日

F. 作業中

正解: **B** (コメントを発表する)



質問: **102**
与えられた：

```
List<Integer> numbers = List.of(2, 3, 0, 8, 1, 9, 5, 7, 6, 4);
int sum = numbers.stream().reduce(0, (n, m) -> n + m);  // line 1
```

削減操作を並列化する必要があります。
これを達成する2つの変更はどれですか？
A. 行1をint sum=numbersに置き換えます。ParallelStream（）減らす（0、（n、m）-> n + m）;
B. 1行目をint sum=numbersに置き換えます。平行（）。ストリーム（）。減らす（0、（n、m）-> n + m）;
C. 1行目をint sum=numberに置き換えます。ストリーム（）。flatMap（a-> a）.reduce（0、（n、m）-> n + m）;
D. 行1をint sum=number.streamに置き換えます。平行（）。（0、（n、m）-> n + m）;を減らします。
E. 1行目をint sum=numbersに置き換えます。ストリーム（）Interate（0、a-> a +1）。Reduce（0、（n、m）-> n + m）;
正解: **C,E (**コメントを発表する**)**

質問: **103**
与えられた：

```
package test.t1;
public class A {
   public int x = 42;
   protected A() {}                    // line 1
}
```

and

```
package test.t2;
import test.t1.*;
public class B extends A {
   int x = 17;                         // line 2
   public B() { super(); }             // line 3
}
```

and

```
package test;
import test.t1.*;
import test.t2.*;
public class Tester {
   public static void main(String[] args) {
      A obj = new B();                 // line 4
      System.out.println(obj.x); // line 5
   }
}
```

結果はどうなりますか？

A. 42

B. 4行目のエラーが原因でコンパイルが失敗します。

C. 5行目のエラーが原因でコンパイルが失敗します。

D. 17

E. 1行目のエラーが原因で、コンパイルが失敗します。

F. 3行目のエラーが原因でコンパイルが失敗します。

G. 2行目のエラーが原因でコンパイルが失敗します。

正解: **(**正解を表示します**)**

質問: **104**

どのコードフラグメントが100個のランダムな数値を出力しますか？

```
A. var r=  new Random();
   new DoubleStream(r::nextDouble).limit(100).forEach(System.out::print);
B. DoubleStream.generate(Random::nextDouble)
            .limit (100).forFach(System.out::print);
C. Doublestream.generate(Random.nextDouble).limit(100).forEach(System.out.print);
D. var r = new Random(); DoubleStream.generate(r::nextDouble).limit(100).forEach(System.out::print);
```

A. オプションD

B. オプションB

C. オプションA

D. オプションC

正解: **A** (コメントを発表する)

質問: **105**

Javaモジュールについて正しい2つのステートメントはどれですか？ 2つ選択してください。）

A. --module-pathからロードされたモジュラーjarは自動モジュールです。

B. -classpathで見つかったクラスは、名前のないモジュールの一部です。

C. -classpathからロードされたモジュラーjarは自動モジュールです。

D. パッケージが名前付きモジュールと名前なしモジュールの両方で定義されている場合、名前なしモジュール内のパッケージは無視されます。

E. 任意の名前付きモジュールは、自動モジュール内のすべてのクラスに直接アクセスできます。

正解: **A,B** (コメントを発表する)

質問: **106**

コードフラグメントが与えられた場合：

```
public static void main(String[] args) {
   List<Integer> even = List.of();
   even.add(0, -1);
   even.add(0, -2);
   even.add(0, -3);
   System.out.println(even);
}
```

出力は何ですか？

A. 実行時例外がスローされます。

B. [-1、-2、-3]

C. コンパイルは失敗します。

D. [-3、-2、-1]

正解: **A** (コメントを発表する)

質問: **107**

与えられた：

```
public interface A {
    public Iterable a();
}
public interface B extends A {
    public Collection a();
}
public interface C extends A {
    public Path a();
}
public interface D extends B, C {
}
```

Dがコンパイルエラーを引き起こすのはなぜですか？

A. DはBとCからa（を継承しますが、戻り型には互換性がありません。

B. DはCからのみa（を継承します。

C. Dは複数のインターフェースを拡張します。

D. Dはメソッドを定義しません。

正解: (正解を表示します)

質問: **108**

開発組織で使用されているツールの機能的なバグに取り組んでいます。調査の結果、この許可を含むセキュリティポリシーファイルを使用してツールが実行されていることがわかりました。

```
grant codebase "file:${klib.home}/j2se/home/klib.jar" {
   permission java.security.AllPermission;
};
```

あなたはどのような行動を取るべきですか？

A. 内部ツールであり、一般に公開されていないため、何もありません。

B. 必要な権限だけをリストすることは継続的なメンテナンスの課題になるため、何もありません。

C. 調査しているバグとは関係がないため、何もありません。

D. 助成金が多すぎるため、助成金を削除します。

E. 付与された過剰なアクセス許可を参照するツールに対してセキュリティバグを報告します。

正解: (正解を表示します)

質問: **109**

与えられた：

```
public class Foo {
    private void print() {
        System.out.println("Bonjour le monde!");
    }
    public void foo() {
        print();
    }
}

public class Bar extends Foo {
    private void print() {
        System.out.println("Hello world!");
    }
    public void bar() {
        print();
    }
    public static void main(String... args) {
        Bar b = new Bar();
        b.foo();
        b.bar();
    }
}
```

出力は何ですか？

A. Hello world！Bonjour le monde！

B. Hello world！Hello world！

C. Bonjour le monde！Hello world！

D. Bonjour le monde！Bonjour le monde！

正解: (正解を表示します)