VCE4Plus

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 guarante 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.	✓	✓	✓

Exam : **1z1-819-JPN**

Title : Java SE 11 Developer (1Z0-819日本語版)

Vendor : Oracle

Version : DEMO

QUESTION NO: 1
Stream で forEach 操作の代わりに peek 操作を使用するのはなぜでしょうか?
A. 現在の項目を処理し、void を返します
B. ストリームの末尾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
Answer: C

QUESTION NO: 2
与えられた条件:

```
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.  }
```

どのループでコンパイル時エラーが発生しますか?
A. ループ開始行11
B. ループ開始行7
C. ループ開始行14
D. ループ開始行3
Answer: C

QUESTION NO: 3
java.util.function パッケージ内のどのインターフェースが void 戻り値型を返しますか?
A. 消費者
B. サプライヤー
C. 関数
D. 述語
Answer: A

QUESTION NO: 4
与えられた条件:

```
for(var i = 0; i < 10; i++) {
  switch(i%5) {
    case 2:
      i *= i;
      break;
    case 3:
      i++;
      break;
    case 1:
    case 4:
      i++;
      continue;
    default:
      break;
  }
  System.out.print(i + " ");
  i++;
}
```

結果はどうなりましたか？

A. 何もない

B. 0

C.10

D.0 4 9

Answer: A

QUESTION NO: 5

適切な JDBC URL はどれですか?

A. jdbe.mysql.com://localhost:3306/データベース

B. http://localhost.mysql.com:3306/database

C. http://localhost mysql.jdbc:3306/database

D. jdbc:mysql://localhost:3306/データベース

Answer: D

QUESTION NO: 6

次のコードフラグメントがあるとします:

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

結果はどうなりましたか？

A.01

B. 0

C.1

D. プログラムは何も印刷しません。

E. 無限ループで 1 を出力します。

Answer: D

QUESTION NO: 7

非数値および無限の入力値に対して例外をスローするメソッドはどれですか?

A.

```
static float validate1(String s) throws IllegalArgumentException {
     return Float.parseFloat(s);
}
```

B.

```
static float validate3(String s, float min, float max) throws IllegalArgumentException {
    float f = Float.parseFloat(s);
    if (!Float.isFinite(f) || f < min || f > max) {
         throw new IllegalArgumentException();
    }
    return f;
}
```

C.

```
static float validate2(String s, float min, float max) throws IllegalArgumentException {
    float f = Float.parseFloat(s);
    if (f < min || f > max) {
        throw new IllegalArgumentException();
    }
    return f;
}
```

D.

```
static float validate4(String s, float min, float max) throws IllegalArgumentException {
    float f = Float.parseFloat(s);
    if (Float.isFinite(f) && f < min && f > max) {
         throw new IllegalArgumentException();
    }
    return f;
}
```

Answer: A

QUESTION NO: 8
与えられた条件:

```
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());
    }
}
```

どちらの記述が正しいでしょうか?
A. クラス Tester 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 test
のメンバーにすでに表示されているためです。
B. パッケージ java.time. のすべてのクラスが Diary クラスにロードされます。
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。
Answer: A

QUESTION NO: 9
TripleThis.java の場合:

```
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、次のコンパイラ警告が表示されます。
注意: 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。
どの 2 つの置換を同時に実行すると、このコンパイラの警告が削除されますか?

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. 12 行目を public static int printValue function<Integer, Integer>, f, T num { に置き換えます。

D. 12行目をpublic static <T> void printValue (Function<T, T> f, T num) {, に置き換えます。

E. 9 行目を function<Integer>, Integer> = X -> { return (integer) x * 3; } に置き換えます。

Answer: A,C

QUESTION NO: 10
与えられた条件:

```
public class Main {
  public static void main(String[] args) {
    for(int i = 0; i < args.length; i++) {
      System.out.println(i + "). " + args[i]);
      switch(args[i]) {
        case "one":
          continue;
        case "two":
          i--;
          continue;
        default:
          break;
      }
    }
  }
}
```

このコマンドで実行します:
java メイン 1 2 3

結果はどうなりましたか？
A. 0)。1
B. 0). 11). 22). 3
C. コンパイルに失敗しました。
D. 0). one1). two1). two... と印刷する無限ループを作成します。
E. java.lang.NullPointerException がスローされます。
Answer: D

QUESTION NO: 11
与える：

```
public enum Season {
   WINTER('w'), SPRING('s'), SUMMER('h'), FALL('f');
   char c;
   private Season(char c) {
     this.c= c;
   }
}
```

コードフラグメント:

```
public static void main(String[] args) {
   Season[] sA = Season.values();
   // line n1
}
```

行 n1 で SPRING を出力する 3 つのコード フラグメントはどれですか。
A. System.out.printin (sA(1)};
B. System.out.printin(Season,values(1)};
C. System.out.printin(Season.SPRING);
D. System.out.printin (to(0)}:
E. System.out.printin(Season.valueof("SPRING").ordinal());
F. System.out.printin(Season.valueOf("SPRING").Ordinal());
G. System.out.printin(Season.valueof('s'));
Answer: A,C,E

QUESTION NO: 12
次のコードフラグメントがあるとします:

```
Locale locale = Locale.US;
// line 1
double currency = 1_00.00;
System.out.println(formatter.format(currency));
```

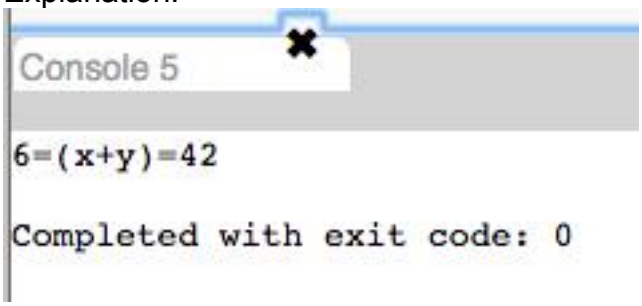
通貨の値を 100.00 ドルとして表示します。
1 行目に挿入されたどのコードがこれを実現しますか?
A. NumberFormat フォーマッタ = NumberFormat.getInstance(locale).getCurrency();
B. NumberFormat フォーマッタ = NumberFormat.getCurrency(locale);

C. NumberFormat フォーマッタ = NumberFormat.getInstance(locale);
D. NumberFormat フォーマッタ = NumberFormat.getCurrencyInstance(locale);
Answer: A

QUESTION NO: 13
与えられた条件:

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

結果はどうなりましたか？
A. 実行時に例外がスローされます。
B. 42=(x+y)=42
C. 42=(x+y)=6
D. 6=(x+y)=42
E. 6=(x+y)=6
Answer: D
Explanation:



QUESTION NO: 14
正しい var 宣言はどれですか? (2 つ選択してください。)
A. var names = new ArrayList<>();
B. var _ = 100;
C. var var = "hello";
D. ここで y = null;
E. 変数 a;
Answer: A,C

QUESTION NO: 15
与えられた条件:

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

and

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

}
```

and

```
Foo f1 = new Foo();
Foo f2 = new Bar();
Bar b1 = new Bar();
Collection<String> c = new ArrayList<>();
```

どれが正しいでしょうか?
A. b1.foo(c) は o1a World! を出力します。
B. b1.foo(c) は Hello world! を出力します。
C. F2.foo(c) は hello world ! を出力します。
D. f1.foo(c) は Hello world! を出力します。
E. f1.foo(c) は ola Mundo1 を出力します
F. f1.foo(c) は Hello world! を出力します。
G. b1. foo(c) は Hello world! を出力します。
H. f2.foo(c) は ola Mund! と出力します。
I. f1.foo(c) は hello world! を出力します。
Answer: D,F,I

QUESTION NO: 16
与えられた条件:

```
import java.sql.Timestamp;
public class Test {
    public static void main(String[] args) {
        Timestamp ts = new Timestamp(1);
    }
}
```

そしてコマンド:
これらのコマンドを実行するとどのような結果になりますか?

A.

```
Test.class -> java.base Test.class -> java.sql java.sql -> java.base
```

B.

On execution, the jdeps command displays an error.

C.

```
Test.class -> java.base Test.class -> java.sql
```

D.

```
Test.class -> java.sql -> java.base
```

Answer: C

QUESTION NO: 17
与えられた条件:

```
public class MyResource {
    public MyResource () {
    }
    // Resource methods
}
```

try-with-resources ステートメントで myResource
クラスを使用したいと考えています。どのような変更を加えることでこれを実現できますか?

A. AutoCloseable を拡張し、close メソッドをオーバーライドします。
B. AutoCloseable を実装し、autoClose メソッドをオーバーライドします。
C. AutoCloseable を拡張し、autoClose メソッドをオーバーライドします。
D. AutoCloseable を実装し、close メソッドをオーバーライドします。

Answer: D

QUESTION NO: 18
与えられた条件:

```
int i = 3;
int j = 25;
System.out.println( i > 2 ? i > 10 ? i * (j + 10) : i * j + 5 : i);
```

結果はどうなりましたか?

A.385
B. 3
C. コンパイルに失敗しました。
D. 80
E. 25
Answer: A

QUESTION NO: 19
与えられた条件:

```
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します:
java メイン 1 2 3
このクラスの出力は何ですか?
A. コンパイルに失敗しました。
B. 1) 12) 23) 3
C. java.lang.ArrayIndexOutOfBoundsException がスローされます。
D. 1) 1つ
E. 何もない
Answer: B

QUESTION NO: 20
与えられた条件:

```
class MyPersistenceData {
   String str;
   private void methodA() {
     System.out.println("methodA");
   }
}
```

あなたはjavを実装したい
a. Io、MypersisteneData クラスへのシリアル化可能なインターフェイス。
どのメソッドをオーバーライドする必要がありますか?
A. readExternalメソッドとwriteExternalメソッド
B. readExternalメソッド
C. writeExternalメソッド

D. 何もない
Answer: A

QUESTION NO: 21
与えられた条件:

```
public class DNASynth {
     int aCount;
     int tCount;
     int cCount;
     int gCount;

     DNASynth(int a, int tCount, int c, int g){
          // line 1
     }
     int setCCount(int c){
          return c;
     }
     void setGCount(int gCount){
          this.gCount = gCount;
     }
}
```

1 行目に挿入されたときにインスタンス変数を正しく変更する 2 行のコードはどれですか? (2 つ選択してください。)
A. setCCount(c) = cCount;
B. tCount = tCount;
C. setGCount(g);
D. cCount = setCCount(c);
E. aCount = a;
Answer: B,E

QUESTION NO: 22
与えられた条件:

```
import java.util.function.BiFunction;
public class Pair<T> {
    final BiFunction<T, T, Boolean> validator;
    T left = null;
    T right = null;
    private Pair() {
      validator=null;
    }
    Pair(BiFunction<T, T, Boolean> v, T x, T y) {
        validator = v;
        set(x, y);
    }
    void set(T x, T y) {
        if (!validator.apply(x, y)) throw new IllegalArgumentException();
        setLeft(x);
        setRight(y);
    }
    void setLeft(T x) {
        left = x;
    }
    void setRight(T y) {
        right = y;
    }
    final boolean isValid() {
        return validator.apply(left, right);
    }
}
```

p instanceof Pair の場合、p.isValid() が true を返す必要があります。
この要件を満たすために必要な可視性の変更の最小セットはどれですか?

A. setLeft と setRight は保護する必要があります。

B. 左と右はプライベートである必要があります。

C. isValid は public である必要があります。

D. left、right、setLeft、setRight はプライベートにする必要があります。

Answer: B

QUESTION NO: 23

次のコードフラグメントがあるとします:

```
Consumer<String> c1 = arg -> System.out.println(arg);
c1.accept("c1 accepted");
Consumer<String> c2 = arg -> System.out.println(arg);
c2.accept("c2 accepted");
c2.andThen(c1).accept("after then");
c2.accept("c2 accepted again");
```

結果はどうなりましたか ?

A.

```
c1 accepted
c2 accepted
and followed by an exception
```

B.

```
c1 accepted
c2 accepted
after then
c1 accepted
c2 accepted again
```

C.

```
c1 accepted
c2 accepted
after then
c2 accepted again
```

D.

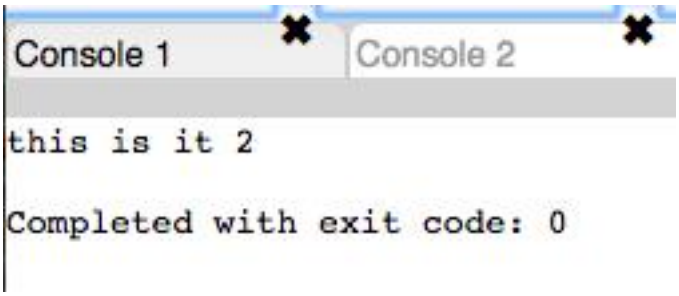
```
c1 accepted
c2 accepted
after then
after then
c2 accepted again
```

Answer: C

QUESTION NO: 24
与えられた条件:

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

結果はどうなりましたか？
A. 1ですか
B. 実行時に IndexOutOfBoundsException がスローされます。
C. 0ですか
D. これがそれだ 2
E. これがそれだ 3
Answer: D
Explanation:



QUESTION NO: 25
オートボクシングの成功例を 2 つ挙げてください。(2 つ選択してください。)
A. 文字列 a = "A";
B. 整数 e = 5;
C. 浮動小数点数 g = Float.valueOf(null);
D. ダブルd = 4;
E. ロング c = 23L;
F. 浮動小数点数 f = 6.0;
Answer: A,B

QUESTION NO: 26
次のコードフラグメントがあるとします:

```
public class Test {
  class L extends Exception { }
  class M extends L { }
  class N extends RuntimeException { }
  public void p() throws L { throw new M(); }
  public void q() throws N { throw new N(); }
  public static void main(String[] args) {
    try {
      Test t = new Test();
      t.p();
      t.q();
    } /* line 1 */ {
      System.out.println("Exception caught");
    }
  }
}
```

1 行目のどのような変更によってこのコードがコンパイルされるようになるでしょうか?
A. catch (L |N e) を追加します。
B. catch (L |MN e) を追加します。
C. catch (L e) を追加します。
D. catch (N | L | M e) を追加します。
E. catch (M |L e) を追加します。
Answer: C

QUESTION NO: 27
次のコードフラグメントがあるとします:

```
char[][] arrays = {{'a', 'd'}, {'b', 'e'}, {'c', 'f'}};
for (char[] xx : arrays) {
    for (char yy : xx) {
        System.out.print(yy);
    }
    System.out.print(" ");
}
```

結果はどうなりましたか？
A. ab cd ef
B. 実行時にArrayIndexOutOfBoundsExceptionがスローされます。
C. コンパイルに失敗しました。
D. abc 定義
E. 広告はcfである
Answer: E

QUESTION NO: 28
サービスは、印刷インターフェースを使用してサービス
プロバイダーをロードするためにどのコード フラグメントを使用しますか?
A. プライベート プリント print = com.service.Provider.getInstance();
B. プライベート java.util.ServiceLoader<Print> ローダー = ServiceLoader.load (Print.class);
C. プライベート java.util.ServiceLoader<Print> ローダー = 新しい java.util.ServiceLoader<>
();
D. プライベート Print print = new com.service.Provider.PrintImpl();
Answer: B

QUESTION NO: 29
与えられた条件:

```
void myLambda() {
    int i = 25;
    Supplier<Integer> foo = () -> i;
    i++;
    System.out.println(foo.get());
}
```

どちらが本当でしょうか？
A. コードはコンパイルされますが、結果は印刷されません。
B. コードは 25 を出力します。
C. コードはコンパイルされません。
D. コードは実行時に例外をスローします。
Answer: C

QUESTION NO: 30
与えられた条件:

```
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. こんにちは、世界!
B. こんにちは世界!こんにちは世界!
C. こんにちは、世界!
D. こんにちは世界! こんにちは世界!
Answer: C
Explanation: