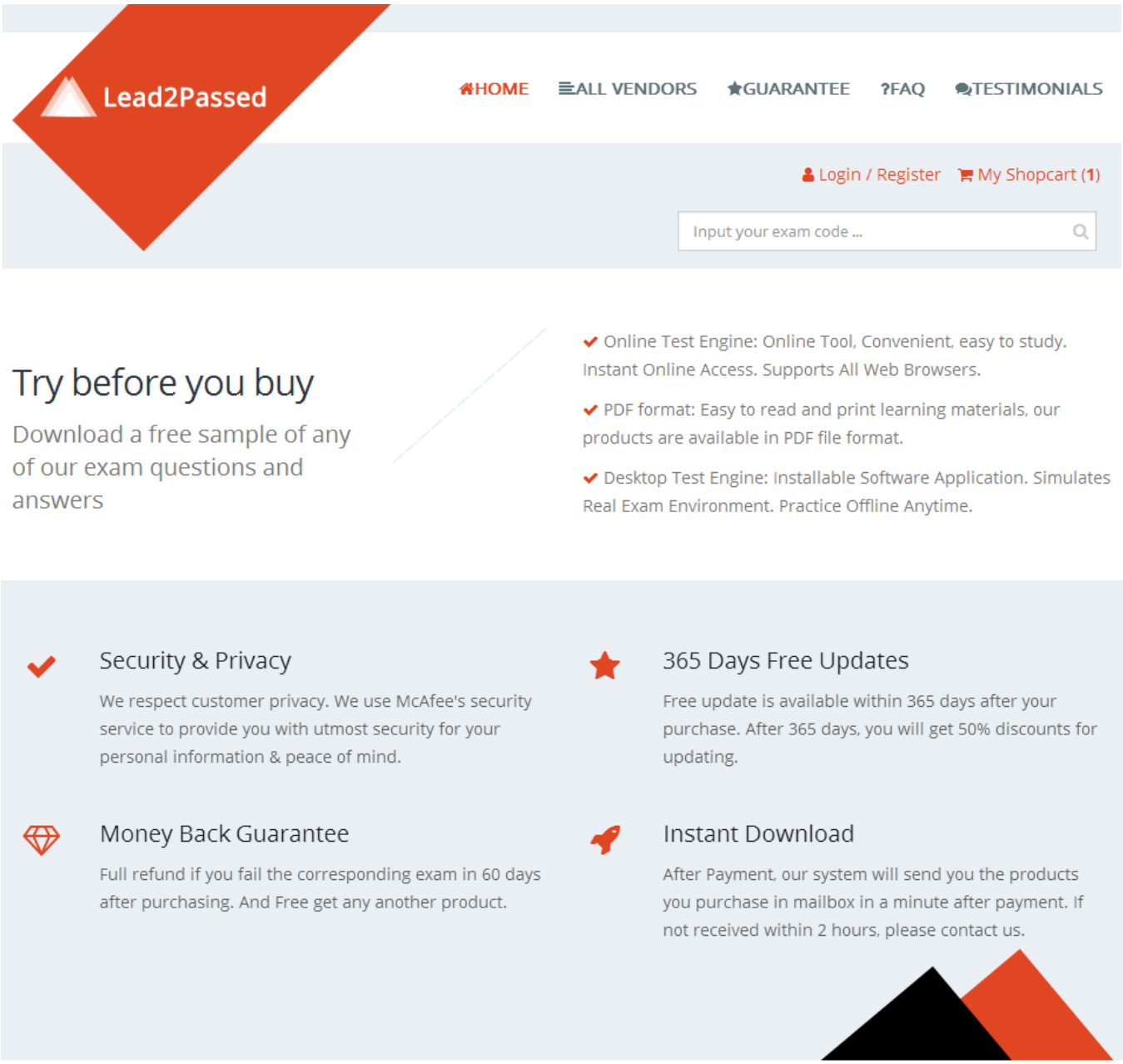
Lead2Passed



http://www.lead2passed.com

Valid Certification Exam Dumps Materials and Study Guide - Lead2Passed

Exam : **1z0-808J**

Title : Java SE 8 Programmer I (1z0-808日本語版)

Vendor : Oracle

Version : DEMO

QUESTION NO: 1
コードの断片を考えると、次のようになります。

```
String[] colors = {"red", "blue", "green", "yellow", "maroon", "cyan"};
```

どのコード部分が青、シアンを出力しますか?

A)
```
for (String c:colors) {
    if (c.length() != 4) {
        continue;
    }
System.out.print(c+", ");
}
```

B)
```
for (String c:colors[]) {
    if (c.length() <= 4) {
        continue;
    }
System.out.print(c+", ");
}
```

C)
```
for (String c:String[] colors) {
    if (c.length() >= 3) {
        continue;
    }
System.out.print(c+", ");
}
```

D)
```
for (String c:colors) {
    if (c.length() != 4) {
        System.out.print(c+", ");
        continue;
    }
}
```

A. オプション A
B. オプション B
C. オプション C
D. オプション D

Answer: A

QUESTION NO: 2
このコードの空白を埋めてコンパイルできるものは次のうちどれですか? (選択肢を 2 つ選択してください。)

```
1.  public void method() ____ Exception {
2.  ____________ Exception();
3.  }
```

A. 1 行目にスローを入力します。
B. 1 行目に、throws new を入力します。
C. 2 行目に、throw new と入力します。
D. 2 行目にスローを入力します。
E. 2 行目に、throws new を入力します。
Answer: AC
Explanation:
Option A and C are the correct answer.
In a method declaration, the keyword throws is used. So here at line 1 we have to use option A.
To actually throw an exception, the keyword throw is used and a new exception is created, so at line 2 we have to use throw and new keywords, which is option C: Finally it will look like;
public void method() throws Exception { throw new Exception0;
}
httpsy/docs.oracle.com/javase/tutorial/essential/io/fileOps.html#exception The correct answer is: On line 1, fill in throws. On line 2, fill in throw new

QUESTION NO: 3
与えられた：

```
class S1 {
     protected void display(int x) {
          System.out.print("Parent" + x);
     }
}
class S2 extends S1 {
     public void display(int x, int y) {
          this.display(x);
          display(y);
          super.display(y);
}
     public void display(int x) {
          System.out.println("Child " + x);
     }
}
```

そしてコードの断片：
S2 sobj = new S2 ( ) ; sobj.display (10、100) ;
結果はどうですか？
A. Child 10
Child 100
Parent 100
B. Parent 10
Child 10
Parent 1000
C. Child 10
Parent 100
Parent 100
D.コンパイル時エラーが発生します。
Answer: D
Explanation:

```
Error: Main method not found in class S1, please define the main method as:
   public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application
```

QUESTION NO: 4
与えられる:

```
public abstract class Shape {
    private int x;
    private int y;
    public abstract void draw();
    public void setAnchor(int x, int y) {
        this.x = x;
        this.y = y;
    }
}
```

シェイプ クラスを正しく使用している 2 つのクラスはどれですか?

```
☐ A) public class Circle implements Shape {
       private int radius;
     }

☐ B) public abstract class Circle extends Shape {
       private int radius;
     }

☐ C) public class Circle extends Shape {
       private int radius;
       public void draw();
     }

☐ D) public abstract class Circle implements Shape {
       private int radius;
       public void draw();
     }

☐ E) public class Circle extends Shape {
       private int radius;
       public void draw() {/* code here */}
     }

☐ F) public abstract class Circle implements Shape {
       private int radius;
       public void draw() { /* code here */ }
     }
```

A. オプション A
B. オプション B
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 F

Answer: BE

Explanation:

When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class (E).

However, if it does not, then the subclass must also be declared abstract (B).
Note: An abstract class is a class that is declared abstract--it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

QUESTION NO: 5
与えられる:

```
1. public class SampleClass  {
2.      public static void main(String[] args){
3.           AnotherSampleClass asc = new AnotherSampleClass();
4.           SampleClass sc = new SampleClass();
5.           //insert code here
6.      }
7. }
8. class AnotherSampleClass extends SampleClass {
9. }
```

「// TODO コード アプリケーション
ロジックはここに」という行に挿入された場合、有効な変更はどのステートメントですか?
A. asc = sc;
B. sc = asc;
C. asc = (object) sc;
D. asc= sc.clone ()
Answer: B
Explanation:
Works fine.
Incorrect answers:
asc = sc.clone();
Incompatible types.
asc =sc;
Incompatible types.
asc = (object) sc;
Syntax error

QUESTION NO: 6
与えられた :

```
public class TestScope {
    public static void main(String[] args) {
        int var1 = 200;
        System.out.print(doCalc(var1));
        System.out.print(" "+var1);
    }
    static int doCalc(int var1){
        var1 = var1 * 2;
        return var1;
    }
}
```

結果は何ですか？

A.400 200

B.200 200

C.400 400

D.コンパイルが失敗します。

Answer: A

QUESTION NO: 7

Given:

```
class X {
    int i;
    static int j;
    public static void main(String[] args) {
        X x1 = new X();
        X x2 = new X();
        x1.i = 3;
        x1.j = 4;
        x2.i = 5;
        x2.j = 6;
        System.out.println(
            x1.i + " " +
            x1.j + " " +
            x2.i + " " +
            x2.j);
    }
}
```

結果は何ですか？

A. 3 4 5 6

B. 3 4 3 6
C. 5 4 5 6
D. 3 6 5 6
Answer: D
Explanation:

```
3 6 5 6

Completed with exit code: 0
```

QUESTION NO: 8
与えられる:

```
String stuff = "TV";
String res = null;

if (stuff.equals ("TV")) {
res = "Walter";
} else if (stuff.equals ("Movie") ) {
res= "White";
} else {
res= "No Result";
}
```

if ブロックを置き換えることができるコードの断片はどれですか?
A. stuff.equals ("TV") ? res= "Walter" : stuff.equals ("Movie") ? res = "White" : res = "No Result";
B. res = stuff.equals ("TV") ? "Walter" else stuff.equals ("Movie")? "White" : "No Result";
C. res = stuff.equals ("TV") ? stuff.equals ("Movie")? "Walter" : "White" : "No Result";
D. res = stuff.equals ("TV")? "Walter" : stuff.equals ("Movie")? "White" : "No Result";
Answer: B

QUESTION NO: 9
どちらのステートメントが正しいのですか？（二つを選ぶ）
A. エラークラスは拡張不能です。
B. エラークラスは拡張可能です。
C. エラーはRuntimeExceptionです。
D. エラーは例外です。

E. エラーは廃棄可能。
Answer: BE

QUESTION NO: 10
コードの断片を考えると：

```
List<String> arrayList = new ArrayList<>();
arrayList.add("Tech");
arrayList.add("Expert");
arrayList.set(0, "Java");
arrayList.forEach (a -> a.concat("Forum"));
arrayList.replaceAll (s -> s.concat("Group"));
System.out.println(arrayList);
```

結果はどうですか？
A. [JavaForum, ExpertForum]
B. [JavaGroup, ExpertGroup]
C. [JavaForumGroup, ExpertForumGroup]
D. [JavaGroup, TechGroup ExpertGroup]
Answer: B
Explanation:

```
21 public class Main {
22   public static void main(String[] args) {
23     List<String> arrayList = new ArrayList<> ();
24     arrayList.add("Tech");
25     arrayList.add("Expert");
26     arrayList.set(0, "Java");
27     arrayList.forEach (a -> a.concat ("Forum"));
28     arrayList.replaceAll (s -> s.concat("Group"));
29     System.out.println(arrayList);
30   }
31
32
33
34
35 }
```

CPU Time: 0.18 sec(s), Memory: 32824 kilobyte(s)

```
[JavaGroup, ExpertGroup]
```

QUESTION NO: 11
コードの断片を考えると、次のようになります。

```
12. int row = 10;
13. for ( ; row > 0 ; ) {
14.     int col = row;
15.     while (col >= 0) {
16.         System.out.print(col + " ");
17.         col -= 2;
18.     }
19.     row = row / col;
20. }
```

結果は何ですか?
A. 10 8 6 4 2 0
B. 10 8 6 4 2
C. 実行時に AnArithmeticException がスローされる
D. プログラムは無限ループに入り、10 8 6 4 2 0 を出力します。 。
E. コンパイルに失敗しました
Answer: D

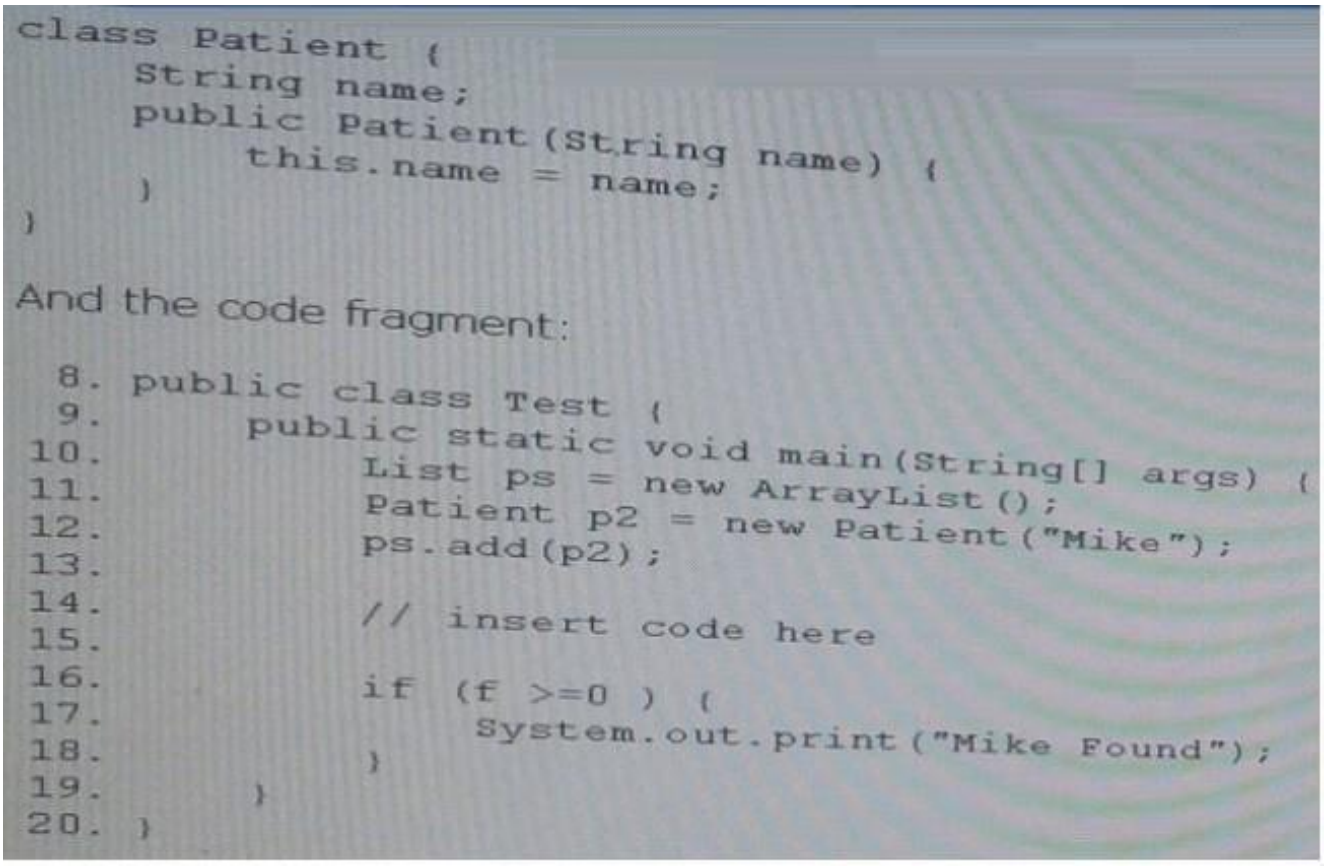
QUESTION NO: 12
コードの断片を考えます：

```
int wd = 0;
String days[] = {"sun", "mon", "wed", "sat"};
for (String s:days) {
    switch (s) {
        case "sat":
        case "sun":
            wd -= 1;
            break;
        case "mon":
            wd -= 1;
            break;
        case "wed":
            wd += 2;
    }
}
System.out.println(wd);
```

結果はどうですか？
A. 3
B. 0
C. コンパイルは失敗します。
D. -1
Answer: D

QUESTION NO: 13
Java クラスの構造に関して正しい 3 つの記述はどれですか?
A. クラスはプライベート コンストラクターを 1 つだけ持つことができます。
B. メソッドはフィールドと同じ名前を持つことができます。
C. クラスにはオーバーロードされた静的メソッドを含めることができます。
D. パブリック クラスには main メソッドが必要です。
E. メソッドはクラスの必須コンポーネントです。
F. フィールドは使用前に初期化する必要はありません。
Answer: BCF

QUESTION NO: 14
与えられる:



```
class Patient {
    String name;
    public Patient(String name) {
        this.name = name;
    }
}
```

And the code fragment:

```
 8. public class Test {
 9.     public static void main(String[] args) {
10.         List ps = new ArrayList();
11.         Patient p2 = new Patient("Mike");
12.         ps.add(p2);
13.
14.         // insert code here
15.
16.         if (f >=0 ) {
17.             System.out.print("Mike Found");
18.         }
19.     }
20. }
```

14 行目に挿入すると、Mike Found を出力できるコード フラグメントはどれですか?

A. int f = ps.indexOf {new patient ("Mike")};

B. int f = ps.indexOf (patient("Mike"));

C. patient p = new Patient ("Mike");
int f = pas.indexOf(P)

D. int f = ps.indexOf(p2);

Answer: D

QUESTION NO: 15

G次のコード・フラグメントがあります:

```
public class Person {
    String name;
    int age = 25;

    public Person (String name) {
        this (); //                              //line n1
        setName(name);
    }
    public Person (String name, int age) {
        Person (name);                          //line n2
        setAge (age);
    }
    //setter and getter methods go here

    public String show () {
        return name + " " + age;
    }
    public static void main (String [] args) {
        Person p1 = new Person ("Jesse");
        Person p2 = new Person ("Walter", 52);
        System.out.println (p1.show () );
        System.out.println (p2.show () );
    }
}
```

結果は何ですか？
A. ラインN1とラインN2のみでコンパイルが失敗します。
B. ラインN2のみでコンパイルが失敗します。
C. ラインN1のみでコンパイルが失敗します。
D. Jesse 25Walter 52
Answer: A
Explanation:
At line n1, Person class hasn't any constructor without arguments.
At line n2, there isn't any method Person. If we want to call the constructor that should be "this(name)".

QUESTION NO: 16
与えられる:
class Test {
public static void main (String[] args) {
int day = 1;

```
switch (day) {
case "7":
System.out.print("Uranus");
case "6":
System.out.print("Saturn");
case "1":
System.out.print("Mercury");
case "2":
System.out.print("Venus");
case "3":
System.out.print("Earth");
case "4":
System.out.print("Mars");
case "5":
System.out.print("Jupiter");
}
}
}
```

独立して行われた 2
つの変更により、コードのコンパイルと実行が可能になるのはどれですか?
A. 各 print ステートメントの後に Break ステートメントを追加します。
B. スイッチのコードブロック内にデフォルトのセクションを追加します。
C. 各ケースのラベルの文字列リテラルを整数に変更します。
D. 変数 day の型を String に変更します。
E. ケースラベルを昇順に並べます。
Answer: CD

QUESTION NO: 17
与えられる:

```
package clothing;
public class Shirt {
    public statuc String getColor() {
        return "Green";
    }
}
```

コードの断片を考えると、次のようになります。

```
package clothing.pants;
// line n1
public class Jeans {
    public void matchShirt(){
        //line n2
        if(color.equals("Green")) {
            System.out.print("Fit")
        }
    }
    public static void main (String[] args) {
        Jeans trouser = new Jeans();
        trouser.matchShirt();
    }
}
```

コード フラグメントが Fit を出力できるようにする 2 つのアクション
セットは、独立してどれですか?
A. n1 行目で insert:import clothing. Shirt;n2 行目で insert:String color = getColor();
B. n1 行目で insert:import clothing.*;n2 行目で insert:String color = Shirt.getColor();
C. n1 行目で挿入:import static clothing. Shirt.getcolor;n2 行目で挿入:String color = getColor()
;
D. n1 行目では変更は必要ありません。n2 行目で次のように挿入します。String color =
Shirt.getColor();
E. n1 行目に挿入:衣類のインポート;n2 行目に挿入:String color = Shirt.getColor();
Answer: A

QUESTION NO: 18
考慮する：
整数 = Integer.valueOff 808.1");
上記の記述について正しいのはどれですか?
A. 変数番号の値は 808.1 になります。
B. 変数番号の値は 808 になります。
C. 変数numberの値は0になります。
D. NumberFormatException がスローされます。
E. コンパイルされません。
Answer: D
Explanation:
The Integer class value of 0 returns an Integer from given string. But we need to pass string which has correct format for integer otherwise it will throw a NumberFormatException.
In this case we have passed string which is not an integer value (since what we passed is fractional number), so option D is correct.

QUESTION NO: 19
与えられる:

```
public class Test {
      public static int stVar = 100;
      public int var = 200;
      public String toString() {
            return var + ":" + stVar;
      }
}
```

コードの断片を考えると、次のようになります。

```
Test t1 = new Test();
t1.var = 300;
System.out.println(t1);
Test t2 = new Test();
t2.stVar = 300;
System.out.println(t2);
```

結果は何ですか?
A. 300:300200:300
B. 300:100200:300
C. 300:00:300
D. 200:300200:300
Answer: D

QUESTION NO: 20
次のコード・フラグメントがあります:

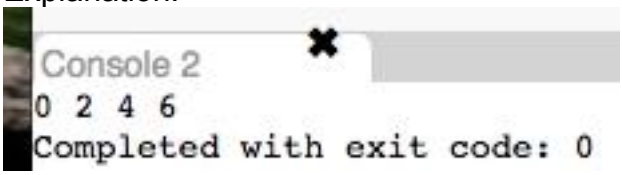
```
public static void main(String[] args) {
      int ii = 0;
      int jj = 7;
      for (ii = 0; ii < jj; ii = ii + 2) {
            System.out.print(ii + " ");
       }
}
```

結果は何ですか。
A. 2 4
B. 0 2 4 6
C. 0 2 4

D. コンパイルが失敗します。

Answer: B

Explanation:



QUESTION NO: 21

ここで指定されたステートメントにより、次の例外がスローされるのはどれですか? int array[] = new int[-2];

A. NullPointerException

B. NegativeArraySizeException

C. ArrayIndexOutOfBoundsException

D. IndexOutOfBoundsException

E. このステートメントは例外を引き起こしません。

Answer: B

Explanation:

In given statement we can see that, we have passed negative value for creating int array, which results a NegativeArraySize Except ion.

Hence option B is correct. Option A is incorrect as it is thrown when an application attempts to use null in a case where an object is required.

Option D is incorrect as IndexOutOfBoundsException thrown to indicate that an index of some sort (such as to an array, to a string, or to a vector) is out of range.

rhttpy/docs.oracle.com/iavase/S/docs/api/java/lang/NegativeArraySizeException.html

QUESTION NO: 22

与えられたコード断片：

```
int num[][] = new int[1][3];
for (int i = 0; i < num.length; i++) {
    for (int j = 0; j < num[i].length; j++) {
        num[i][j] = 10;
    }
}
```

外部ループの正常終了後のnum配列の状態を表すオプションはどれですか？

A)
```
num[0][0]=10
num[0][1]=10
num[0][2]=10
```

B)
```
num[0][0]=10
num[1][0]=10
num[2][0]=10
```

C)
```
num[0][0]=10
num[0][1]=0
num[0][2]=0
```

D)
```
num[0][0]=10
num[0][1]=10
num[0][2]=10
num[0][3]=10
num[1][0]=0
num[1][1]=0
num[1][2]=0
num[1][3]=0
```

A.オプションA
B.オプションB
C.オプションC
D.オプションD
Answer: A
Explanation:
At first look we can exclude option D because the number of elements in the array is 3, the result of multiplying the two array dimensions 1 x 3.
We can run the code
public class Main {
public static void main(String[] args) {
int num[][] = new int[1][3];
for (int i=0; i<num.length; i++) {
for (int j=0; j<num[i].length; j++) {
num[i][j] = 10;
System.out.println("num[" + i + "][" + j + "]= " + num[i][j]);
}
}
}
}
the output is

num[0][0]= 10
num[0][1]= 10
num[0][2]= 10

QUESTION NO: 23
コードの断片を考えると：

```
6. char colorCode = 'y';
7. switch (colorCode) {
8.   case 'r':
9.        int color = 100;
10.       break;
11.  case 'b':
12        color = 10;
13.       break;
14.  case 'y':
15.       color = 1;
16.       break;
17. }
18. System.out.println(color);
```

結果は何ですか？
A.18行目にコンパイル時エラーが発生します。
B.9行目でコンパイル時エラーが発生します。
C.印刷します：1
D.行12および15の行でコンパイル時エラーが発生します。
Answer: A
Explanation:

```
2  class colorCode {
3    public static void main(String[] args) {
4
5      char colorCode = 'y';
6      switch (colorCode) {
7        case 'r':
8          int color = 100;
9          break;
10       case 'b':
11         color = 10;
12         break;
13       case 'y':
14         color = 1;
15         break;
16     }
17     System.out.println(color);
18   }
19 }
```

QUESTION NO: 24
与えられる:

```
public class Vowel {
    private char var;
    public static void main(String[] args) {
        char var1 = 'a';
        char var2 = var1;
        var2 = 'e';

        Vowel obj1 = new Vowel();
        Vowel obj2 = obj1;
        obj1.var = 'i';
        obj2.var = 'o';

        System.out.println(var1 + ", " +var2);
        System.out.print(obj1.var + ", " + obj2.var);
    }
}
```

A. a, e
i, o
B. a, e
o, o
C. e, e
l, o
D. e, e
o, o
Answer: B

QUESTION NO: 25
次のクラスがあるとします。

```
public class CheckingAccount {
    public int amount;
    public CheckingAccount(int amount){
        this.amount = amount;
    }
    public int getAmount(){
        return amount;
    }
    public void changeAmount(int x){
        amount += x;
    }
}
```

そして、別のクラスにある次の main メソッドがあるとします。

```
public static void main(String[] args) {
    CheckingAccount acct = new CheckingAccount((int)(Math.random()*1000));
    //line n1
    System.out.println(acct.getAmount());
}
```

n1 行目に独立して挿入すると、プログラムに ao バランスを出力させる 3 行はどれですか?

A. this.amount = 0;

B. amount = 0;

C. acct (0) ;

D. acct.amount = 0;

E. acct. getAmount () = 0;

F. acct.changeAmount(0);

G. acct.changeAmount(-acct.amount);

H. acct.changeAmount(-acct.getAmount());

Answer: DGH

Explanation:

A and B don't compile because there isn't a variable amount in method main.

C is wrong because we can't call the constructor acct directly.

E is wrong because we can't make a method on acct equal to 0.

F is wrong because does not change variable amount of class CheckingAccount.

QUESTION NO: 26

与えられる:

```
1. import java.util.ArrayList;
2. import java.util.List;
3.
4. public class Whizlabs{
5.
6.          public static void main(String[] args){
7.                    List<Integer> list = new ArrayList<>();
8.                    list.add(21); list.add(13);
9.                    list.add(30); list.add(11);
10.                   list.add(2);
11.                   //insert here
12.                   System.out.println(list);
13.         }
14. }
```

11 行目に挿入されたものは次の出力を提供しますか?
[21、15、11]
A. list.removeIf(e > e%2 != 0);
B. list.removeIf(e -> e%2 != 0);
C. Ust.removeIf(e -> e%2 = 0);
D. list.remove(e -> e%2 = 0);
E. 上記のどれでもない。
Answer: C
Explanation:
In output we can see that only odd numbers present, so we need to remove only even numbers to get expected output.
From Java SE 8, there is new method call removeIf which takes predicate object and remove elements which satisfies predicate condition. Predicate has functional method call take object and check if the given condition met or not, if met it returns true, otherwise false. Option C we have passed correct lambda expression to check whether the number is odd or even that matches to the functional method of predicate interface.
Option A is incorrect as it is invalid lambda expression. Option B is incorrect as it removes all odd numbers.
Option D is incorrect as there is no remove method that takes predicate as argument.
https://docs.oracle.eom/javase/8/docs/api/java/util/ArrayList.html

QUESTION NO: 27
与えられる:

```
public class Circle {
    double radius;
    public double area;
    public Circle(double r) { radius = r; }
    public double getRadius() { return radius; }
    public void setRadius(double r) { radius = r; }
    public double getArea() { return /* ??? */; }
}

class App {
    public static void main(String[] args) {
        Circle c1 = new Circle(17.4);
        c1.area = Math.PI * c1.getRadius() * c1.getRadius();
    }
}
```

クラスのカプセル化が不十分です。
代わりに面積を計算して返すように円クラスを変更する必要があります。
クラスが適切にカプセル化されていることを確認するために必要な 3 つの変更はどれですか?
(3つ選択してください)

A. エリアフィールドを削除します。

B. getArea () メソッドを変更します public double getArea () { return area; }

C. Circle コンストラクターと setRadius()
メソッドで半径が設定されている場合、面積が再計算され、それが area フィールドに格納されます。

D. getRadius () メソッドを変更します。 public double getRadius () { area = Math.PI * radius * radius;戻り半径。 }

Answer: ABC

QUESTION NO: 28
コードの断片を考えると、次のようになります。

```
public static void main (String [] args) {
    String myStr = "Hello World";
    myStr.trim ()
    int i1 = myStr.indexOf (" ");
    System.out.printLn (i1);
}
```

結果は何ですか?

A. 実行時に例外がスローされます。

B. -1

C. 5

D. 0

Answer: C

QUESTION NO: 29
与えられたコードフラグメント：

```
24. float var1 = (12_345.01 >= 123_45.00) ? 12_456 : 124_56.02f;
25. float var2 = var1 + 1024;
26. System.out.print(var2);
```

結果は何ですか？
A. 実行時に例外がスローされます。
B. コンパイルが失敗します。
C. 13480.0
D. 13480.02
Answer: C

QUESTION NO: 30
カプセル化の利点となる 3 つのステートメントはどれですか?
A. クライアントを変更せずにクラス実装を変更できるようにします
B. 機密データがオブジェクトから漏洩するのを防ぎます。
C. コードによる例外の発生を防止します。
D. クラス実装がその不変条件を保護できるようにします。
E. クラスを同じパッケージに結合することを許可します
F. 同じクラスの複数のインスタンスを安全に作成できるようにします。
Answer: ABD

QUESTION NO: 31
与えられる:

```
public class App {
    public static void main(String[] args) {
        int i = 10;
        int j = 20;
        int k = j += i / 5;
        System.out.print(i + " : " + j + " : " + k);

    }
}
```

結果は何ですか?
A. 10 : 22 : 20
B. 10 : 22 : 22
C. 10 : 22 : 6
D. 10 : 30 : 6
Answer: B

QUESTION NO: 32
与えられた：

```
4. public class Shop{
5.   public static void main(String[] args) {
6.        int price = 1000;
7.        int qty = 2;
8.        String grade = "2";
9.        double discount = 0.0;
10.       switch(grade) {
11.            case "1":
12.                 discount = price * 0.1;
13.                 break;
14.            case "2":
15.                 discount = price * 0.5;
16.                 continue:
17.            default:
18.                 System.out.println("Thank You!");
19.       }
20.       System.out.println(discount);
21.       }
22. }
```

どちらが正しいですか？

A. The program executes and prints:
500.0

B. Commenting line 16 enables the program to print:
Thank You! 500.0

C. Commenting line 13 enables the program to print:
Thank You! 500.0

D. The program executes and prints:
Thank You! 500.0

Answer: B

Explanation:

```
16 public class Shop {
17     public static void main(String[] args) {
18         int price = 1000;
19         int qty = 2;
20         String grade = "2";
21         double discount = 0.0;
22         switch(grade) {
23             case "1":
24                 discount = price * 0.1;
25                 break;
26             case "2":
27                 discount = price * 0.5;
28                 //continue;
29             default:
30                 System.out.println("Thank You!");
31         }
32         System.out.println(discount);
33     }
34 }
```

Result

CPU Time: 0.16 sec(s), Memory: 32260 kilobyte(s)

```
Thank You!
500.0
```

QUESTION NO: 33

Java ソース ファイルがあるとします。

```
class X {
    X() { }
    private void one() { }
}

public class Y extends X {
    Y() { }
    private void two() { one(); }
    public static void main(String[] args) {
        new Y().two();
    }
}
```

このコードをコンパイルするにはどのような変更を加える必要がありますか?

A. クラス x の宣言に public 修飾子を追加します。

B. protected 修飾子を x() コンストラクターに追加します
C. one() メソッドの宣言の private 修飾子を protected に変更します。
D. Y () コンストラクターを削除します。
E. two () メソッドから private 修飾子を削除します
Answer: C
Explanation:
Using the private protected, instead of the private modifier, for the declaration of the one() method, would enable the two() method to access the one() method.

QUESTION NO: 34
どのステートメントがswitchステートメントについて真ですか？
A. デフォルトセクションが含まれている必要があります。
B. break文は、各caseブロックの最後に必須です。
C. そのケースのラベルリテラルは、実行時に変更することができます。
D. その式は単一の値に評価されなければなりません。
Answer: D

QUESTION NO: 35
与えられる:
class Sports {
int num_players;
String name, ground_condition;
Sports(int np, String sname, String sground){
num_players = np;
name = sname;
ground_condition = sground;
}
}
class Cricket extends Sports {
int num_umpires;
int num_substitutes;
Which code fragment can be inserted at line //insert code here to enable the code to compile?
A. Cricket() {
super(11, "Cricket", "Condidtion OK");
num_umpires =3;
num_substitutes=2;
}
B. Cricket() {
super.ground_condition = "Condition OK";
super.name="Cricket";
super.num_players = 11;
num_umpires =3;
num_substitutes=2;

```
}
```
C.
```
Cricket() {
this(3,2);
super(11, "Cricket", "Condidtion OK");
}
Cricket(int nu, ns) {
this.num_umpires =nu;
this.num_substitutes=ns;
}
```
D.
```
Cricket() {
this.num_umpires =3;
this.num_substitutes=2;
super(11, "Cricket", "Condidtion OK");
}
```
Answer: A
Explanation:
Incorrect:
not C, not D: call to super must be the first statement in constructor.