

Exam Questions 1z0-808

Java SE 8 Programmer I

<https://www.2passeasy.com/dumps/1z0-808/>



NEW QUESTION 1

Given the code fragment:

```
public static void main(String[] args) {
    int ans;
    try {
        int num = 10;
        int div = 0;
        ans = num / div;
    } catch (ArithmeticException ae) {
        ans = 0; // line n1
    } catch (Exception e) {
        System.out.println("Invalid calculation");
    }
    System.out.println("Answer = " + ans); // line n2
}
```

What is the result?

- A. Answer = 0
- B. Invalid calculation
- C. Compilation fails only at line n1.
- D. Compilation fails only at line n2.
- E. Compilation fails at line n1 and line2.

Answer: C

Explanation:

```
1 public class Test {
2     public static void main(String[] args) {
3         int ans;
4         try {
5             int num = 10;
6             int div = 0;
7             ans = num / div;
8         } catch (ArithmeticException ae) {
9             ans = 0;
10        } catch (Exception e) {
11            System.out.println("Invalid calculation");
12        }
13        System.out.println("Answer = " + ans); //line n2
14    }
15 }
16 }
17 }
```

✖ variable ans might not have been initialized

NEW QUESTION 2

Given the following main method:

```
public static void main(String[] args) {
    int num = 5;
    do {
        System.out.print(num-- + " ");
    } while (num == 0);
}
```

What is the result?

- A. 5 4 3 2 1 0
- B. 5 4 3 2 1
- C. 4 2 1
- D. 5
- E. Nothing is printed

Answer: D

NEW QUESTION 3

You are asked to develop a program for a shopping application, and you are given this information:

- The application must contain the classes Toy, EduToy, and ConsToy. The Toy class is the superclass of the other two classes.
- The int calculatePrice (Toy t) method calculates the price of a toy.
- The void printToy (Toy t) method prints the details of a toy.

Which definition of the Toy class adds a valid layer of abstraction to the class hierarchy?

- A
- ```
public abstract class Toy{
 public abstract int calculatePrice(Toy t);
 public void printToy(Toy t) { /* code goes here */ }
}
```
- B
- ```
public abstract class Toy {
    public int calculatePrice(Toy t) ;
    public void printToy(Toy t) ;
}
```
- C
- ```
public abstract class Toy {
 public int calculatePrice(Toy t);
 public final void printToy(Toy t){ /* code goes here */ }
}
```
- D
- ```
public abstract class Toy {
    public abstract int calculatePrice(Toy t) { /* code goes here */ }
    public abstract void printToy(Toy t) { /* code goes here */ }
}
```

- A. Option A
 B. Option B
 C. Option C
 D. Option D

Answer: A

NEW QUESTION 4

Given:

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

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

Which code fragment can replace the if block?

- 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";
```

- A. Option A
 B. Option B
 C. Option C
 D. Option D

Answer: D

NEW QUESTION 5

Given the code fragment:

```
LocalDate Time dt= LocalDateTime.of (2014, 7, 31, 1, 1);  
dt.plusDays (30);  
dt. plusMonths (1);  
System.out.print (dt format (DateTimeFormatter. ISO_DATE) );
```

What is the result?

- A. An exception is thrown at runtime
- B. 07-31-2014
- C. 2014-07-31
- D. 2014-09-30

Answer: A

NEW QUESTION 6

Given the code fragment:

```
int x = 100;  
int a = x++;  
int b = ++x;  
int c = x++;  
int d = (a < b) ? (a < c) ? a: (b < c )? b: c: x;  
System.out.println(d);
```

What is the result?

- A. 100
- B. 101
- C. 102
- D. 103
- E. Compilation fails

Answer: E

NEW QUESTION 7

Given the code fragment:

```
public static void main(String[] args) {  
    int data[] = {2010, 2013, 2014, 2015, 2014};  
    int key = 2014;  
    int count = 0;  
    for (int e: data) {  
        if (e != key) {  
            continue;  
            count++;  
        }  
    }  
    System.out.print(count + " Found");  
}
```

What is the result?

- A. Compilation fails.
- B. 0 Found
- C. 1 Found
- D. 3 Found

Answer: A

NEW QUESTION 8

Which two statements are true about Java byte code? (Choose two.)

- A. It can be serialized across network.
- B. It can run on any platform that has a Java compiler.
- C. It can run on any platform.
- D. It has ".java" extension.
- E. It can run on any platform that has the Java Runtime Environment.

Answer: AE

NEW QUESTION 9

This grid shows the state of a 2D array:

0	0	
	X	0
X		X

The grid is created with this code:

```
char[][] grid = new char[3][3];
grid[1][1] = 'X';
grid[0][0] = '0';
grid[2][0] = 'X';
grid[0][1] = '0';
grid[2][2] = 'X';
grid[1][2] = '0';
//line n1
```

Which line of code, when inserted in place of //line n1, adds an X into the grid so that the grid contains three consecutive Xs?

- A. grid[2][1] = 'X';
- B. grid[3][2] = 'X';
- C. grid[3][1] = 'X';
- D. grid[2][3] = 'X';

Answer: D

NEW QUESTION 10

Given the code fragment:

```
LocalDateTime dt = LocalDateTime.of(2014, 7, 31, 1, 1);
dt.plusDays(30);
dt.plusMonths(1);
System.out.println(dt.format(DateTimeFormatter.ISO_DATE_TIME));
```

What is the result?

- A. An exception is thrown at runtime
- B. 2014-07-31T01:01:00
- C. 2014-07-31
- D. 2014-09-30T00:00:00

Answer: B

NEW QUESTION 10

Given the code fragment:

```
abstract class Toy {
    int price;
    // line n1
}
```

Which three code fragments are valid at line n1?

A

```
public static void insertToy() {  
    /* code goes here */  
}
```

B

```
final Toy getToy() {  
    return new Toy();  
}
```

C

```
public void printToy();
```

D

```
public int calculatePrice() {  
    return price;  
}
```

E

```
public abstract int computeDiscount();
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D
- E. Option E

Answer: CDE**NEW QUESTION 14**

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);  
    }  
}
```

What is the result?

- 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

NEW QUESTION 18

Given:

```
class Caller {
    private void init () {
        System.out.println("Initialized");
    }

    private void start () {
        init();
        System.out.println("Started");
    }
}

public class TestCall {
    public static void main(String[] args) {
        Caller c = new Caller();
        c.start(); // line n1
        c.init();  // line n2
    }
}
```

What is the result?

- A. Compilation fails at line n1.
- B. InitializedStartedInitialized
- C. InitializedStarted
- D. Compilation fails at line n2.

Answer: D

NEW QUESTION 23

Given this class:

```
public class CheckingAccount {
    public int amount;
    //line n1
}
```

And given this main method, located in another class:

```
public static void main(String[] args) {
    CheckingAccount acct = new CheckingAccount();
    //line n2
}
```

Which three pieces of code, when inserted independently, set the value of amount to 100?

A

```
At line n1 insert:
    public CheckingAccount() {
        amount = 100;
    }
```

B

```
At line n2 insert:
    this.amount = 100;
```

C

```
At line n2 insert:
    amount = 100;
```

D

```
At line n1 insert:
    public CheckingAccount() {
        this.amount = 100;
    }
```

E

```
At line n2 insert:
    acct.amount = 100;
```

F

```
At line n1 insert:
    public CheckingAccount() {
        acct.amount = 100;
    }
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D
- E. Option E
- F. Option F

Answer: DE**NEW QUESTION 28**

Given the code fragment:

```
public static void main(String[] args) {
    LocalDate date = LocalDate.of(2012, 1, 30);
    date.plusDays(10);
    System.out.println(date);
}
```

What is the result?

- A. 2012-02-10
- B. 2012-01-30
- C. 2012-02-10 00:00
- D. A DateTimeException is thrown at runtime.

Answer: C**NEW QUESTION 30**

Given:


```
public class Triangle {
    static double area;
    int b = 2, h = 3;
    public static void main(String[] args) {
        double p, b, h;          //line n1
        if (area == 0) {
            b = 3;
            h = 4;
            p = 0.5;
            area = p * b * h;      //line n2
        }
        System.out.println("Area is " + area);
    }
}
```

What is the result?

- A. Area is 6.0
- B. Area is 3.0
- C. Compilation fails at line n1
- D. Compilation fails at line n2.

Answer: D

NEW QUESTION 34

What is the name of the Java concept that uses access modifiers to protect variables and hide them within a class?

- A. Encapsulation
- B. Inheritance
- C. Abstraction
- D. Instantiation
- E. Polymorphism

Answer: A

Explanation:

Using the private modifier is the main way that an object encapsulates itself and hide data from the outside world.

NEW QUESTION 35

Given:

```
public class Test {
    int x, y;

    public Test(int x, int y) {
        initialize(x, y);
    }

    public void initialize(int x, int y) {
        this.x = x * x;
        this.y = y * y;
    }

    public static void main(String[] args) {
        int x = 3, y = 5;
        Test obj = new Test(x, y);
        System.out.println(x + " " + y);
    }
}
```

What is the result?

- A. Compilation fails.
- B. 3 5
- C. 0 0
- D. 9 25

Answer: B

NEW QUESTION 39

Given the code fragment:

```
public static void main(String[] args) {
    StringBuilder sb = new StringBuilder("Java");
    String s = "Java";

    if (sb.toString().equals(s.toString())) {
        System.out.println("Match 1");
    } else if (sb.equals(s)) {
        System.out.println("Match 2");
    } else {
        System.out.println("No Match");
    }
}
```

What is the result?

- A. Match 1
- B. Match 2
- C. No Match
- D. A NullPointerException is thrown at runtime.

Answer: A

NEW QUESTION 41

Given:

```
class Caller {
    private void init () {
        System.out.println("Initialized");
    }

    private void start () {
        init();
        System.out.println("Started");
    }
}

public class TestCall {
    public static void main(String[] args) {
        Caller c = new Caller();
        c.start();
        c.init();
    }
}
```

What is the result?

- A. An exception is thrown at runtime.
- B. InitializedStartedInitialized
- C. InitializedStarted
- D. Compilation fails.

Answer: D

NEW QUESTION 43

Given the code fragment:

```
String[] strs = {"A", "B"};
int idx = 0;
for (String s : strs) {
    strs[idx].concat(" element " + idx);
    idx++;
}
for (idx = 0; idx < strs.length; idx++) {
    System.out.println(strs[idx]);
}
```

What is the result?

- A. AB
- B. A element 0B element 1
- C. A NullPointerException is thrown at runtime.
- D. A 0B 1

Answer: C

NEW QUESTION 44

Given the code fragment:

```
if (aVar++ < 10) {
    System.out.println(aVar + " Hello Universe!");
} else {
    System.out.println(aVar + " Hello World!");
}
```

What is the result if the integer aVar is 9?

- A. Compilation fails.
- B. 10 Hello Universe!
- C. 10 Hello World!
- D. 9 Hello World!

Answer: B

NEW QUESTION 45

Given the code fragment:

```
public static void main(String[] args) {
    int[][] arr = new int [2] [4];
    arr[0] = new int []{1, 3, 5, 7};
    arr[1] = new int []{1, 3};
    for (int[] a : arr) {
        for (int i : a) {
            System.out.print(i+ " ");
        }
        System.out.println();
    }
}
```

What is the result?

A. Compilation fails.

B.

```
1 3
1 3
```

C.

```
1 3
```

followed by an `ArrayIndexOutOfBoundsException`

D.

```
1 3
1 3 0 0
```

E.

```
1 3 5 7
1 3
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D
- E. Option E

Answer: E

Explanation:

	0.75	0.76	0.77	0.78	0.79	0.80	0.81	0.82	0.83	0.84	0.85	0.86	0.87	0.88	0.89	0.90	0.91	0.92	0.93	0.94	0.95	0.96	0.97	0.98	0.99	1.00	1.01	1.02	1.03	1.04	1.05	1.06	1.07	1.08	1.09	1.10	1.11	1.12	1.13	1.14	1.15	1.16	1.17	1.18	1.19	1.20	1.21	1.22	1.23	1.24	1.25	1.26	1.27	1.28	1.29	1.30	1.31	1.32	1.33	1.34	1.35	1.36	1.37	1.38	1.39	1.40	1.41	1.42	1.43	1.44	1.45	1.46	1.47	1.48	1.49	1.50	1.51	1.52	1.53	1.54	1.55	1.56	1.57	1.58	1.59	1.60	1.61	1.62	1.63	1.64	1.65	1.66	1.67	1.68	1.69	1.70	1.71	1.72	1.73	1.74	1.75	1.76	1.77	1.78	1.79	1.80	1.81	1.82	1.83	1.84	1.85	1.86	1.87	1.88	1.89	1.90	1.91	1.92	1.93	1.94	1.95	1.96	1.97	1.98	1.99	2.00	2.01	2.02	2.03	2.04	2.05	2.06	2.07	2.08	2.09	2.10	2.11	2.12	2.13	2.14	2.15	2.16	2.17	2.18	2.19	2.20	2.21	2.22	2.23	2.24	2.25	2.26	2.27	2.28	2.29	2.30	2.31	2.32	2.33	2.34	2.35	2.36	2.37	2.38	2.39	2.40	2.41	2.42	2.43	2.44	2.45	2.46	2.47	2.48	2.49	2.50	2.51	2.52	2.53	2.54	2.55	2.56	2.57	2.58	2.59	2.60	2.61	2.62	2.63	2.64	2.65	2.66	2.67	2.68	2.69	2.70	2.71	2.72	2.73	2.74	2.75	2.76	2.77	2.78	2.79	2.80	2.81	2.82	2.83	2.84	2.85	2.86	2.87	2.88	2.89	2.90	2.91	2.92	2.93	2.94	2.95	2.96	2.97	2.98	2.99	3.00	3.01	3.02	3.03	3.04	3.05	3.06	3.07	3.08	3.09	3.10	3.11	3.12	3.13	3.14	3.15	3.16	3.17	3.18	3.19	3.20	3.21	3.22	3.23	3.24	3.25	3.26	3.27	3.28	3.29	3.30	3.31	3.32	3.33	3.34	3.35	3.36	3.37	3.38	3.39	3.40	3.41	3.42	3.43	3.44	3.45	3.46	3.47	3.48	3.49	3.50	3.51	3.52	3.53	3.54	3.55	3.56	3.57	3.58	3.59	3.60	3.61	3.62	3.63	3.64	3.65	3.66	3.67	3.68	3.69	3.70	3.71	3.72	3.73	3.74	3.75	3.76	3.77	3.78	3.79	3.80	3.81	3.82	3.83	3.84	3.85	3.86	3.87	3.88	3.89	3.90	3.91	3.92	3.93	3.94	3.95	3.96	3.97	3.98	3.99	4.00	4.01	4.02	4.03	4.04	4.05	4.06	4.07	4.08	4.09	4.10	4.11	4.12	4.13	4.14	4.15	4.16	4.17	4.18	4.19	4.20	4.21	4.22	4.23	4.24	4.25	4.26	4.27	4.28	4.29	4.30	4.31	4.32	4.33	4.34	4.35	4.36	4.37	4.38	4.39	4.40	4.41	4.42	4.43	4.44	4.45	4.46	4.47	4.48	4.49	4.50	4.51	4.52	4.53	4.54	4.55	4.56	4.57	4.58	4.59	4.60	4.61	4.62	4.63	4.64	4.65	4.66	4.67	4.68	4.69	4.70	4.71	4.72	4.73	4.74	4.75	4.76	4.77	4.78	4.79	4.80	4.81	4.82	4.83	4.84	4.85	4.86	4.87	4.88	4.89	4.90	4.91	4.92	4.93	4.94	4.95	4.96	4.97	4.98	4.99	5.00	5.01	5.02	5.03	5.04	5.05	5.06	5.07	5.08	5.09	5.10	5.11	5.12	5.13	5.14	5.15	5.16	5.17	5.18	5.19	5.20	5.21	5.22	5.23	5.24	5.25	5.26	5.27	5
--	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	---

• • • • •

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual 1z0-808 Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the 1z0-808 Product From:

<https://www.2passeasy.com/dumps/1z0-808/>

Money Back Guarantee

1z0-808 Practice Exam Features:

- * 1z0-808 Questions and Answers Updated Frequently
- * 1z0-808 Practice Questions Verified by Expert Senior Certified Staff
- * 1z0-808 Most Realistic Questions that Guarantee you a Pass on Your FirstTry
- * 1z0-808 Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year