

Passcert

Higher Quality, better service!



Q&A

[Http://www.passcert.com](http://www.passcert.com)

We offer free update service for one year.

Exam : 1Z0-809

Title : Java SE 8 Programmer II

Version : DEMO

1. Given the definition of the Vehicle class:

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

and this code fragment:

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

What is the result?

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

Answer: A

2. Given:

```
IntStream stream = IntStream.of (1,2,3);
IntFunction<Integer> inFu= x -> y -> x*y;          //line n1
IntStream newStream = stream.map(inFu.apply(10));   //line n2
newStream.forEach(System.out::print);
```

Which modification enables the code fragment to compile?

- A. Replace line n1with:
IntFunction<UnaryOperator> inFu = x -> y -> x*y;
- B. Replace line n1with:
IntFunction<IntUnaryOperator> inFu = x -> y -> x*y;
- C. Replace line n1with:
BiFunction<IntUnaryOperator> inFu = x -> y -> x*y;
- D. Replace line n2with:
IntStream newStream = stream.map(inFu.applyAsInt (10));

Answer: B

3. Given the code fragment:

```
List<Integer> values = Arrays.asList (1, 2, 3);
values.stream ()
    .map(n -> n*2)          //line n1
    .peek(System.out::print) //line n2
    .count ();
```

What is the result?

- A. 246
- B. The code produces no output.
- C. A compilation error occurs at line n1.
- D. A compilation error occurs at line n2.

Answer: A

4. Given the code fragment:

```
public class Foo {
    public static void main (String [ ] args) {
        Map<Integer, String> unsortMap = new HashMap< > ( );
        unsortMap.put (10, "z");
        unsortMap.put (5, "b");
        unsortMap.put (1, "d");
        unsortMap.put (7, "e");
        unsortMap.put (50, "j");

        Map<Integer, String> treeMap = new TreeMap <Integer, String> (new
        Comparator<Integer> ( ) {
            @Override public int compare (Integer o1, Integer o2) {return o2.
            compareTo(o1); } } );

        treeMap.putAll (unsortMap);

        for (Map.Entry<Integer, String> entry : treeMap.entrySet ( ) ) {
            System.out.print (entry.getValue ( ) + " ");
        }
    }
}
```

What is the result?

- A. A compilation error occurs.
- B. d b e z j
- C. j z e b d
- D. z b d e j

Answer: C

5. Which two reasons should you use interfaces instead of abstract classes? (Choose two.)

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

Answer: BE

Explanation:

Reference: <https://books.google.com.br/books?id=nS2tBQAAQBAJ&pg=PT235&lpg=PT235&dq=You+want+to+share+code+among+several+closely+related+classes.&source=bl&ots=3oY0u2XXN-&sig=uVFS0KB15BqyEgghXnnjJSUdcrE&hl=pt-BR&sa=X&ved=0ahUKEwjIsKe-n6baAhVEhZAKHeiEDTgQ6AEIMDAB#v=onepage&q=You%20want%20to%20share%20code%20among%20several%20closely%20related%20classes.&f=false>