Java Certified #16

A question lead guide to prepare Java certification

Working with Arrays and Collections

Given:

```
var array1 = new String[]{ "foo", "bar", "buz" };
var array2[] = { "foo", "bar", "buz" };
var array3 = new String[3] { "foo", "bar", "buz" } ;
var array4 = { "foo", "bar", "buz" };
String array5[] = new String[]{ "foo", "bar", "buz" };
```

Which arrays compile? (Select 2)

- ➔ **array1**
- ➔ **array2**
- ➔ **array3**
- ➔ **array4**
- ➔ **array5**

array1 & array5

Array1

var array1 = new String[]{ "foo", "bar", "buz" };

This syntax is correct. Using `new String[]{ ... }` explicitly initializes a string array, and `var` allows the type to be inferred. **Compiles successfully.**

Array2

var array2[] = { "foo", "bar", "buz" };

This syntax is incorrect. The `var` keyword cannot be used with array brackets (`[]`) as part of the variable name. **Does not compile.**

Array3

var array3 = new String[3] { "foo", "bar", "buz" };

This syntax is incorrect. When specifying the size of an array (`new String[3]`), you cannot initialize its elements in the same statement. **Does not compile.**

Array4

var array4 = { "foo", "bar", "buz" };

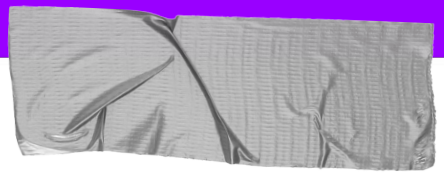
This syntax is incorrect. For type inference with `var`, the array must be explicitly initialized with `new String[]{ ... }`. Without the `new String[]`, the compiler cannot infer the type. **Does not compile.**

Array5

String array5[] = new String[]{ "foo", "bar", "buz" };

This syntax is correct. The `new String[]{ ... }` explicitly initializes the array, and the variable is declared as a `String` array. **Compiles successfully.**

Using Java I/O API



Given:

```
try ( FileOutputStream fos = new FileOutputStream( "t.tmp");
      ObjectOutputStream oos = new ObjectOutputStream( fos)) {
    fos.write("Today");
    fos.writeObject("Today");
    oos.write("Today");
    oos.writeObject("Today");
} catch (Exception ex) {}
```

Which statement compiles?

- ➔ fos.write("Today");
- ➔ fos.writeObject("Today");
- ➔ oos.write("Today");
- ➔ oos.writeObject("Today");

oos.writeObject("Today");

None of the other options compile due to type mismatches or incorrect method calls. Thus, only one statement compiles successfully, and **none of the others are correct.**

`fos.write("Today");`
`fos` is a `FileOutputStream` instance. The `FileOutputStream` class has a `write(int b)` method to write a single byte or a `write(byte[] b)` method to write a byte array. Passing a `String` directly to `write` is not allowed, as there is no overload for `String`. **Does not compile.**

`fos.writeObject("Today");`
`fos` is a `FileOutputStream`, and the `FileOutputStream` class does not have a `writeObject` method. This method is specific to `ObjectOutputStream`, not `FileOutputStream`. **Does not compile.**

`oos.write("Today");`
`oos` is an `ObjectOutputStream`. The `ObjectOutputStream` class does not have a `write(String s)` method. It has methods such as `write(int b)` or `write(byte[] b)` but no overload for directly writing a `String`. **Does not compile.**

`oos.writeObject("Today");`
`oos` is an `ObjectOutputStream`. The `writeObject(Object obj)` method of `ObjectOutputStream` is specifically designed to serialize objects, and a `String` is serializable because it implements the `Serializable` interface. **Compiles successfully.**

Using Object-Oriented Concepts in Java

Given:

```
var _ = 3;
var $ = 7;
System.out.println( _ + $ );
```

What is printed?

- ➔ **10**
- ➔ **_$**
- ➔ **It throws an exception.**
- ➔ **Compilation fails.**

Compilation fails.

The correct answer is: **Compilation fails.**

Explanation:

1. **Variable Names in Java:**
   - `_` and `$` are valid variable names in Java. However, starting from **Java 9**, using `_` as an identifier is **no longer allowed** because it is reserved as a keyword for future use. Attempting to declare a variable named `_` results in a compilation error.
2. **Specific Error:**
   - The compiler will produce an error like:
     error: as of release 9, '_' is a keyword, and may not be used as an identifier

Using underscore ("_") as an identifier generates an error in Java SE 9. :
https://openjdk.org/jeps/213

udemy Pro

https://bit.ly/javaOCP

Sale:$9.99

bitly

jroadmap

https://bit.ly/3NaZFf6