Java Certified #4

A question lead guide to prepare Java certification

Using Object-Oriented Concepts in Java

Given:

```
void verifyNotNull( Object input ) {
    boolean enabled = false;
    assert enabled = true;
    assert enabled;
    System.out.println( input.toString() );
    assert input != null;}
```

When does the given method throw a NullPointerException?

- ➔ **Only if assertions are enabled and the input argument is null**
- ➔ **Only if assertions are disabled and the input argument is null**
- ➔ **A NullPointerException is always thrown if the input argument is null**
- ➔ **A NullpointerException is never thrown**

A NullPointerException is always thrown if the input argument is null

If assertions are disabled, only the first and the fourth statements in the given method get executed. If the `input` is `null`, a `NullPointerException` will be thrown.

If assertions are enabled, the first `assert` statement will set variable `enabled` to `true`, then the second `assert` statement confirms that. If the `input` argument is `null`, a `NullPointerException` will still be thrown from the next statement.

From the above analysis, we can see that all the assert statements in the given method are useless.

https://docs.oracle.com/javase/8/docs/technotes/guides/language/assert.html

Using Java I/O API

Given:

```
public class Test {
    public static void main( String[] args ) throws IOException {
        Path p1 = Path.of( "f1.txt" );
        Path p2 = Path.of( "f2.txt" );
        Files.move( p1, p2 );
        Files.delete( p1 );
    }}//In which case does the given program throw an exception?
```

➔ **File f1.txt exists while file f2.txt doesn't**

➔ **File f2.txt exists while file f1.txt doesn't**

➔ **Both files f1.txt and f2.txt exist**

➔ **Neither files f1.txt nor f2.txt exist**

➔ **An exception is always thrown**

An exception is always thrown

If file `f1.txt` doesn't exist, a `NoSuchFileException` is always thrown as the source file of the move operation is missing.

If both files exist, a `FileAlreadyExistsException` is thrown since the target file of the move operation already existed.

If file `f1.txt` exists while `f2.txt` doesn't, the move operation succeeds. However, this means `f1.txt` will go away after that operation.

Consequently, the delete operation will fail with a `NoSuchFileException` .

https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/nio/file/Files.html

Working with Arrays and Collections

Given:

```
List l1 = new ArrayList<>( List.of( "a", "b" ) );
List l2 = new ArrayList<>( Collections.singletonList( "c" ) );
Collections.copy( l1, l2 );
l2.set( 0, "d" );
System.out.println( l1 );//What is the output of the given code fragment?
```

- [a,b]
- [c,b]
- [d]
- [d,b]
- An IndexOutOfBoundsException is thrown
- An UnsupportedOperationException is thrown

[c,b]

`Collections.copy( List<? super T> dest, List<? extends T> src)` copies all of the elements from one list into another. After the operation, the index of each copied element in the destination list will be identical to its index in the source list. The destination list's size must be greater than or equal to the source list's size. If it is greater, the remaining elements in the destination list are unaffected. With parameters:`dest` - The destination list. and `src` - The source list.

We can see that after the `copy` operation, the first element in the destination list is replaced by the first one in the source list, which is letter `c`. The second element in the destination list is untouched. Meanwhile, all changes to the source list after the copy aren't relevant to the destination list.

https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Collections.html#copy(java.util.List,java.util.List)

https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/util/Collections.html https://bit.ly/javaOCP