TWO-MINUTE DRILL

Here are some of the key points from the certification objectives in this chapter.

Using String and StringBuilder (OCA Objectives 2.6 and 2.7)

- ❑ `String` objects are immutable, and `String` reference variables are not.
- ❑ If you create a new `String` without assigning it, it will be lost to your program.
- ❑ If you redirect a `String` reference to a new `String`, the old `String` can be lost.
- ❑ `String` methods use zero-based indexes, except for the second argument of `substring()`.
- ❑ The `String` class is `final`—it cannot be extended.
- ❑ When the JVM finds a `String` literal, it is added to the `String` literal pool.
- ❑ Strings have a *method* called `length()`—arrays have an *attribute* named `length`.
- ❑ `StringBuilder` objects are mutable—they can change without creating a new object.
- ❑ `StringBuilder` methods act on the invoking object, and objects can change without an explicit assignment in the statement.
- ❑ Remember that chained methods are evaluated from left to right.
- ❑ `String` methods to remember: `charAt()`, `concat()`, `equalsIgnoreCase()`, `length()`, `replace()`, `substring()`, `toLowerCase()`, `toString()`, `toUpperCase()`, and `trim()`.
- ❑ `StringBuilder` methods to remember: `append()`, `delete()`, `insert()`, `reverse()`, and `toString()`.

Using Arrays (OCA Objectives 4.1 and 4.2)

- ❑ Arrays can hold primitives or objects, but the array itself is always an object.
- ❑ When you declare an array, the brackets can be to the left or right of the name.
- ❑ It is never legal to include the size of an array in the declaration.
- ❑ You must include the size of an array when you construct it (using `new`) unless you are creating an anonymous array.
- ❑ Elements in an array of objects are not automatically created, although primitive array elements are given default values.
- ❑ You'll get a `NullPointerException` if you try to use an array element in an object array, if that element does not refer to a real object.
- ❑ Arrays are indexed beginning with zero.
- ❑ An `ArrayIndexOutOfBoundsException` occurs if you use a bad index value.
- ❑ Arrays have a `length` attribute whose value is the number of array elements.
- ❑ The last index you can access is always one less than the length of the array.
- ❑ Multidimensional arrays are just arrays of arrays.
- ❑ The dimensions in a multidimensional array can have different lengths.
- ❑ An array of primitives can accept any value that can be promoted implicitly to the array's declared type—for example, a `byte` variable can go in an `int` array.
- ❑ An array of objects can hold any object that passes the IS-A (or `instanceof`) test for the declared type of the array. For example, if `Horse` extends `Animal`, then a `Horse` object can go into an `Animal` array.
- ❑ If you assign an array to a previously declared array reference, the array you're assigning must be the same dimension as the reference you're assigning it to.
- ❑ You can assign an array of one type to a previously declared array reference of one of its supertypes. For example, a `Honda` array can be assigned to an array declared as type `Car` (assuming `Honda` extends `Car`).

Using ArrayList (OCA Objective 4.3)

- ❑ `ArrayLists` allow you to resize your list and make insertions and deletions to your list far more easily than arrays.
- ❑ For the OCA 7 exam, the only `ArrayList` declarations you need to know are of this form:

```
ArrayList<type> myList = new ArrayList<type>();
List<type> myList2 = new ArrayList<type>();  // polymorphic
```

- ❑ `ArrayLists` can hold only objects, not primitives, but remember that autoboxing can make it look like you're adding primitives to an `ArrayList` when in fact you're adding a wrapper version of a primitive.
- ❑ An `ArrayList`'s index starts at 0.
- ❑ `ArrayLists` can have duplicate entries. Note: Determining whether two objects are duplicates is trickier than it seems and doesn't come up until the OCP 7 exam.
- ❑ `ArrayList` methods to remember: `add(element)`, `add(index, element)`, `clear()`, `contains()`, `get(index)`, `indexOf()`, `remove(index)`, `remove(object)`, and `size()`.