Assessment

LATEST SUBMISSION GRADE

100%

1. The import statement is used to ... **1 / 1 point**

- ○ Compile another class as part of your compilation unit.
- ○ Import a non-public member of another class.

  Feedback. No. Import has no effect on access rules.
- ◉ Import a public member of a package other than the one you are defined in.
- ○ Reference a class from within your package.

✓ **Correct**
Correct.

2. By convention, package names start with ... **1 / 1 point**

- ◉ Your Domain Name in reverse order, starting with the Top Level Domain (com, edu, *etc.*)
- ○ Your Domain Name.
- ○ Your company name and project codename.
- ○ Your initials and the last four digits of your Social Security Number.

✓ **Correct**
Correct.

3. An import statement of the form **importpackagename.*** ... **1 / 1 point**

- ◉ Not recommended, especially when using an IDE, because it imports everything in a package, potentially leading to name collisions with other packages.
- ○ Recommended, because code will execute faster with fewer import statements than many import statements.
- ○ Is invalid. The import statement must specify a fully qualified class name, *e.g.*, **importpackagename.Classname**.
- ○ Is recommended, because it imports all classes in the package.
- ○ Is helpful when not using an IDE, because **importcom.myco.*** imports every class in all of myco's packages everywhere.

✓ **Correct**
Correct. Consider that **importjava.sql.*** and **import java.util.*** would create a name collision with **Date**.

4. How can you resolve a name collision if you need to use two classes of the same name that are in multiple packages, *e.g.*, java.util.Date and java.sql.Date? **1 / 1 point**

- ○ You can't.
- ◉ Use the fully qualified class name of one or both throughout the consuming class.
- ○ Import both, and the compiler will figure it out from context.

✓ **Correct**
Correct.

5. Strings are ... **1 / 1 point**

- ◉ Immutable
- ○ Primitives, which is why we can write String s = "Hello World"
- ○ Designed to efficiently edit text.

✓ **Correct**
Feedback: Correct.

6. StringBuffer and StringBuilder differ in that ... **1 / 1 point**

- ○ StringBuffer is designed to **buffer** strings for I/O, and StringBuilder is designed to help build new String objects
- ○ StringBuilder is synchronized, and StringBuffer is faster.
- ◉ They are identical, even down to sharing the same code, but StringBuffer's methods are synchronized, and StringBuilder's methods are not.
- ○ They are similar, but StringBuffer has some methods that were found to be unsafe, so they are removed in StringBuilder.
- ○ They are the same, but Sun wanted to change the name.

✓ **Correct**
Correct

7. In order to use java.lang.Math, you must ... **1 / 1 point**

- ○ Create an instance. It is implicitly imported because it is in java.lang, but you need to create an instance so that the math functions can remember where they left off.
- ○ import it and create an instance.
- ◉ Simply refer to Math.*f*, where *f* is any of its members.

✓ **Correct**
Correct. All of its members are static.

8. True or false: Anything closed in parenthesis ( ) is converted to a String reference and object by Java. **1 / 1 point**

- ○ True
- ◉ False

✓ **Correct**
Right. Anything enclosed in quotes " " is converted to a String reference and object.

9. Strings can be concatenated (chained together) using _____. **1 / 1 point**

- ◉ the plus sign (+)
- ○ the ampersand (&)
- ○ ampersand and plus (& +)

✓ **Correct**
Correct.

10. _____ provides useful functions for operations and you never instantiate it. **1 / 1 point**

- ○ StringBuilder
- ○ Classpath
- ◉ The Math class

✓ **Correct**
Right.