Review Assessment

LATEST SUBMISSION GRADE

100%

1. Down-casting _____. **1 / 1 point**

- (●) Converts an object reference from a super-type to a sub-type.
- ( ) Converts an object reference from a sub-class to a reference to a super-class.
- ( ) Converts the object to a new type.
- ( ) Is always safe.

✓ **Correct**
Correct. Down-casting converts an object reference from a super-type to a sub-type.

2. An Abstract Class can be _____ . (Select all that apply) **1 / 1 point**

- [ ] instantiated.
- [ ] used as leaf classes in the inheritance tree.
- [x] incapable of being instantiated.

✓ **Correct**
True, but there's still more.

- [x] a partially implemented class.

✓ **Correct**
True, but there's still more.

3. True or false: Unlike regular classes, abstract classes support multiple inheritances. **1 / 1 point**

- ○ True
- ◉ False

✓ **Correct**
Correct

4. True or false: You must use @Override to override an inherited method. **1 / 1 point**

- ○ True
- ◉ False

✓ **Correct**
Correct. It is just a hint to the compiler.

5. True or false: Because they cannot be instantiated, abstract classes cannot be used as the type of a parameter. **1 / 1 point**

- ○ True
- ◉ False

✓ **Correct**
Correct

6. Abstract can contain _____ . (Select all that apply) **1 / 1 point**

- ☑ Abstract properties

✓ **Correct**
Right, but that just means abstract methods.

- ☑ Abstract methods

✓ **Correct**
Yes.

- ☑ Implemented methods

✓ **Correct**
Yes.

- ☐ Abstract attributes

7. Using the keyword **new** with an Abstract Class will result in a _____. 1 / 1 point

- (●) Compile time error.
- ( ) Runtime error

✓ **Correct**
Correct

8. True or False: An abstract class must have one or more abstract methods. 1 / 1 point

- (●) False
- ( ) True

✓ **Correct**
Correct

9. In order to use an Abstract Class, you _____ . 1 / 1 point

- ( ) Extend it as an Interface
- ( ) Instantiate it
- (●) Create a subclass of it that implements all abstract methods.

✓ **Correct**
Correct.

10. Abstract Classes are best used to _____ . 1 / 1 point

- (●) Provide reusable code, with attributes to support the code.
- ( ) Provide complete implementations of objects that you don't want other classes to instantiate, but want to instantiate, yourself.
- ( ) Define interfaces to be used in disjoint parts of frameworks

✓ **Correct**
Correct