

Chapter 9 Abstract Classes and Interfaces

1. C and F
2. The benefits are for generic programming. A variable of `GeometricObject` type can use the `findArea` and `findPerimeter` methods at compilation time.
3. d.
4. No you must cast the return value to the target class.

```
String s = (String)Max.max("abc", "efg");  
Date date = (Date)Max.max(new Date(), new Date());
```

5. The benefits are for generic programming. A variable of the class that implements `Comparable` can be passed to a method that requires a `Comparable` type.
6. If the `CloneableCircle` class does not override the `clone()` method, the program would receive a syntax error because `clone()` is protected in `java.lang.Object`. If `CloneableCircle` does not implement `java.lang.Cloneable` in Line 31, the program would receive a runtime error because `c2` would be `null` in Line 7.
7.
It displays the first line a time string such as Sun Nov 02 13:07:26 EST 2003
true
true
8. A syntax error is reported because `clone()` is protected in `Object`. To enable cloning, do two things: (1) override `clone()` in the class for the object to be cloned; (2) implement `java.lang.Cloneable` for the class.

9.
`new int[10]` cannot be assigned to into a variable of `Object[]` type, but `new String[100]`, `new Object[50]`, or `new Calendar[20]` are fine.

10. See the section "Processing Primitive Type Values as Objects." These classes are useful when passing numerical values as objects.

11.
Integer i = new Integer("23");
Answer: Correct

```
Integer i = new Integer(23);
```

Answer: Correct

```
Integer i = Integer.valueOf("23");
```

Answer: Correct

```
Integer i = Integer.parseInt("23",8);
```

Answer: Incorrect

```
Double d = new Double();
```

Answer: Incorrect, no default constructor in Double

```
Double d = Double.valueOf("23.45");
```

Answer: Correct

```
int i = (Integer.valueOf("23")).intValue();
```

Answer: Correct

```
double d = (Double.valueOf("23.4")).doubleValue();
```

Answer: Correct

```
int i = (Double.valueOf("23.4")).intValue();
```

Answer: Correct

```
String s = (Double.valueOf("23.4")).toString();
```

Answer: Correct

12. Use new Integer(int).toString() to convert an integer to a string. Use new Double(double).toString() to convert a double to a string.
13. At runtime, JVM attempts to convert numberRef to a Double object, but numberRef is an instance of Integer, not Double.
14. numberArray[0] is of the Integer type.
15. The program has a syntax error because x does not have the compareTo method.
16. The program has a syntax error because the member access operator (.) is executed before the casting operator.
- 17.

```
Number x = 3; // Correct, this is same as x = new Integer(3)  
Integer x = 3; // Correct  
Double x = 3; // Wrong, this is same as Double x = new Integer(3)  
Double x = 3.0; // Correct
```

```
int x = new Integer(3); // Correct  
int x = new Integer(3) + new Integer(4); // Correct  
double y = 3.4;
```

```
y.intValue(); // Wrong
```

18. Find these terms in this chapter.

19. Indicate true or false for the following statements:

1. An abstract class can have instances created using the constructor of the abstract class.

Answer: No, but an instance of its concrete subclass is also an instance of the parent abstract class. You cannot create an object using the constructor of the abstract class. However, you can create an object using a concrete subclass of the abstract class. This object is an instance of the subclass and it is also an instance of the abstract class.

2. An abstract class can be extended.

Answer: True

3. You can always successfully cast a subclass to a superclass.

Answer: True

4. You can always successfully cast a superclass to a subclass.

Answer: False

5. An interface can be a separate unit and can be compiled into a bytecode file.

Answer: True

6. A subclass of a non-abstract superclass cannot be abstract.

Answer: False

7. A subclass cannot override a concrete method in a superclass to declare it abstract.

Answer: False, This is rare, but useful when the implementation of the method in the superclass becomes invalid in the subclass. In this case, the subclass must be declared abstract.