# Chapter 11 Object-Oriented Design

1. The relationships among classes are association, aggregation, composition, dependency, and inheritance. For a summary of graphical notations, see Appendix G.

2.

Company and Employee: Aggregation

Course and Faculty: Association

Student and Person: Inheritance (Student is a subclass of Person)

House and Window: Composition

Account and Savings Account: Inheritance

JOptionPane uses String: Dependency

Loan has Date: Composition

3.

The output is

```
1/3 + 1/2 = 5/6
1/2 + 1/3 = 5/6
1/3 - 1/2 = -1/6
1/2 - 1/3 = 1/6
1/3 * 1/2 = 1/6
1/2 * 1/3 = 1/6
1/3 / 1/2 = 2/3
1/2 / 1/3 = 3/2
1/3 = 0.3333333333333333
1/2 = 0.5
```

4.

Line 4

```
    System.out.println(r.add(new Rational()));
```

is wrong, because r’s type is Number, it doesn’t have the add method.

Line 5

```
    System.out.println((Rational)r.add(new Rational()));
```

is wrong, because the . operator is performed before casting.

5.

Line 1
```
    Number r = new Number();
```

is wrong, because Number is an abstract class.

6.

(A)
Yes.

Line 1 is fine, since Rational implements Comparable.
Line 2 is fine, since any object is also an instance of the Object class.

(B)
Yes.

Line 1 is fine, since Rational implements Comparable.
Line 2 is fine, since any object is also an instance of the Object class.

7.
See Section 10.6.1

8.
See Section 10.6.2

9.
See Section 10.6.3

10.
See Section 10.6.3

11.
All of these are poor design

12.
- A static method may reference instance data fields or invoke instance methods. (incorrect)
- An instance method may reference static data fields or invoke static methods. (correct)
- A constructor may be static. (incorrect)

- A constructor may be private. (correct)
- A constructor may invoke a static method. (correct)
- A constructor may invoke an overloaded constructor. (correct)
- A constructor invokes its superclass no-arg constructor by default if a constructor does not invoke an overloaded constructor or its superclass's constructor. (correct)
- An abstract class contains constructors. (correct)
- The constructors in an abstract class are public. (should be protected, but public is fine)
- The constructors in an abstract class are private. (incorrect)
- You may declare a final abstract class. (incorrect)
- An interface may contain constructors. (incorrect)
- An interface may contain instance data fields. (incorrect)
- An interface may contain static methods. (incorrect)