Chapter 9 Object-Oriented Software Development

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

2.

Company and Employee: Association

Course and Faculty: Association

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

House and Window: Composition

Account and Savings Account: Inheritance

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

4.

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

Integer i = new Integer(23);
Answer: Correct

Integer i = Integer.valueOf("23");
Answer: Correct (Integer.valueOf("23") returns an Integer object)

Integer i = Integer.parseInt("23",8);
Answer: Incorrect

Double d = new Double();
Answer: Correct

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: Incorrect

5. Use new Integer(int).toString() to convert an integer to a string.
6. Use new Double(double).toString() to convert a double to a string.
7. At runtime, JVM attempts to convert numberRef to a Double object, but numberRef is an instance of Integer, not Double.
8. Similar reason as in 7.
9. c.
10. d.