Homework

The following homework is designed to cover the course objectives for this unit.

Homework Exercise 7.1

Submit your written answers to the following 20 questions to your instructor at the beginning of Unit 8.

1. Which of the following statements converts a double value d into a string s?

   a. s = new Double(d).stringOf();
   b. s = String.stringOf(d);
   c. s = (new Double(d)).toString();
   d. s = (Double.valueOf(s)).toString();

2. Assume Calendar calendar = new GregorianCalendar(). Which of the following statements will return the number of days in a month?

   a. calendar.getActualMaximum(Calendar.DAY_OF_MONTH)
   b. calendar.get(Calendar.MONTH_OF_YEAR)
   c. calendar.get(Calendar.WEEK_OF_MONTH)
   d. calendar.get(Calendar.WEEK_OF_YEAR)
   e. calendar.get(Calendar.MONTH)

3. Assume Calendar calendar = new GregorianCalendar(). Which of the following statements will return the week of the year?

   a. calendar.get(Calendar.MONTH_OF_YEAR)
   b. calendar.get(Calendar.WEEK_OF_YEAR)
   c. calendar.get(Calendar.WEEK_OF_MONTH)
   d. calendar.get(Calendar.MONTH)

4. What will be the output of running the class Test with the following code lines?

```
interface A {
}

class C {
}

class B extends D implements A {
}

public class Test extends Thread {
  public static void main(String[] args) {
    B b = new B();
    if (b instanceof A)
      System.out.println("b is an instance of A");
    if (b instanceof C)
      System.out.println("b is an instance of C");
  }
}

class D extends C {
}
```

   a. b is an instance of A followed by b is an instance of C.
   b. b is an instance of C.
   c. There will be no output.
   d. b is an instance of A.

5. What is the output of the following code?

```
public class Test {
  public static void main(String[] args) {
    java.math.BigInteger x = new java.math.BigInteger("3");
    java.math.BigInteger y = new java.math.BigInteger("7");
    x.add(y);
    System.out.println(x);
  }
}
```

   a. 3
   b. 4
   c. 11
   d. 10

6. Assume Calendar calendar = new GregorianCalendar(). Which of the following statements will return the month of the year?

   a. calendar.get(Calendar.MONTH_OF_YEAR)
   b. calendar.get(Calendar.MONTH)
   c. calendar.get(Calendar.WEEK_OF_YEAR)
   d. calendar.get(Calendar.WEEK_OF_MONTH)

7. Analyze the following code:

   ```
   Number[] numberArray = new Integer[2];
   numberArray[0] = new Double(1.5);
   ```

   What will happen when the code is executed?

   a. At runtime, new Integer[2] is assigned to numberArray. This makes each element of numberArray an Integer object. Therefore, you cannot assign a Double object to it.
   b. You cannot use Number as a data type because it is an abstract class.
   c. Each element of numberArray is of the Number type; therefore, you cannot assign a Double object to it.
   d. Each element of numberArray is of the Number type; therefore, you cannot assign an Integer object to it.

8. Which of the following statements will convert a string s into a double value d?

   a. d = Double.parseDouble(s);
   b. d = Double.valueOf(s).doubleValue();
   c. d = (new Double(s)).doubleValue();
   d. All of the above

9. Which of the following declares an abstract method in an abstract Java class?

   a. public abstract method();
   b. public abstract void method() {}
   c. public void abstract Method();
   d. public abstract void method();
   e. public void method() {}

10. Analyze the following code:

```
public class Test {
  public static void main(String[] args) {
    Number x = new Integer(3);
    System.out.println(x.intValue());
    System.out.println(x.compareTo(new Integer(4)));
  }
}
```

What will happen when the code is executed?

a. The program has a syntax error because intValue is an abstract method in Number.
b. The program has a syntax error because x does not have the compareTo method.
c. The program has a syntax error because an Integer instance cannot be assigned to a Number variable.
d. The program compiles and runs fine.

11. Analyze the following code:

```
public class Test {
  public static void main(String[] args) {
    Number x = new Integer(3);
    System.out.println(x.intValue());
    System.out.println((Integer)x.compareTo(new Integer(4)));
  }
}
```

What will happen when the code is executed?

a. The program has a syntax error because x cannot be cast into Integer.
b. The program has a syntax error because an Integer instance cannot be assigned to a Number variable.
c. The program compiles and runs fine.
d. The program has a syntax error because the member access operator (.) is executed before the casting operator.
e. The program has a syntax error because intValue is an abstract method in Number.

12. Analyze the following code:

```
Number numberRef = new Integer(0);
Double doubleRef = (Double)numberRef;
```

    What will happen when the code is executed?

    a. A runtime class casting exception occurs because numberRef is not an instance of Double.
    b. You can convert an int to double; therefore, you can cast an Integer instance to a Double instance.
    c. There is no such class named Integer. You should use the class Int.
    d. The compiler detects that numberRef is not an instance of Double.
    e. The program runs fine because Integer is a subclass of Double.

13. Which of the following statements correctly declares an interface?

    a. abstract interface A { abstract void print() { };}
    b. interface A { void print() { }; }
    c. abstract interface A { print(); }
    d. interface A { void print();}

14. What is the output of Integer.parseInt("10", 2)?

    a. 2;
    b. Invalid statement;
    c. 10;
    d. 1;

15. Which of the following class definitions defines a legal abstract class?

    a. public class abstract A { abstract void unfinished(); }
    b. abstract class A { abstract void unfinished(); }
    c. class A { abstract void unfinished(); }
    d. class A { abstract void unfinished() { } }

16. Analyze the following code:

```
1. import java.util.*;
2. public class Test {
3. public static void main(String[] args) {
4. Calendar[] calendars = new Calendar[10];
5. calendars[0] = new Calendar();
6. calendars[1] = new GregorianCalendar();
7. }
8. }
```

    What will happen when the code is executed? (Select all that apply.)

    a. The program has a syntax error on Line 6 because Calendar[1] is not of a GregorianCalendar type.
    b. The program has a syntax error on Line 5 because java.util.Calendar is an abstract class.
    c. The program has a syntax error on Line 4 because java.util.Calendar is an abstract class.

17. __________ is a special form of association that represents an ownership relationship between two objects.

    a. Inheritance
    b. Aggregation
    c. Association
    d. Composition

18. The Rational class in this chapter extends java.lang.Number and implements java.lang.Comparable. Analyze the following code:

    ```
    public class Test {
    public static void main(String[] args) {
    Number[] numbers = {new Rational(1, 2), new Integer(4), new Double(5.6)};
    java.util.Arrays.sort(numbers);
    }
    }
    ```

    What will happen when the code is executed?

    a. The program has a syntax error because numbers is declared as Number[]; therefore, you cannot pass it to Arrays.sort(Object[]).
    b. The program has a syntax error because numbers is declared as Number[]; therefore, you cannot assign {new Rational(1, 2), new Integer(4), new Double(5.6)} to it.
    c. The program has a runtime error because the compareTo methods in Rational, Integer, and Double classes do not compare the value of one type with a value of another type.
    d. The program has a runtime error because number is declared as Number[]; therefore, you cannot assign {new Rational(1, 2), new Integer(4), new Double(5.6)} to it.

19. __________ represents the roles the object plays. The objects at the top of the diagram represent class roles.

    a. Activation
    b. Method invocation
    c. Class role
    d. Lifeline

20. Which of the following statements is ***incorrect*** about constructors?

    a. A constructor may invoke a static method.
    b. A constructor may be private.
    c. A constructor may invoke an overloaded constructor.
    d. A constructor invokes its superclass no-arg constructor by default if a constructor does not invoke an overloaded constructor or its superclass's constructor.
    e. A constructor may be static.