1. Which of the following pairs of declarations will cause an error message?

I
```
double x = 14.7;
int y = x;
```

II
```
double x = 14.7;
int y = (int) x;
```

III
```
int x = 14;
double y = x;
```

(A) None
(B) I only
(C) II only
(D) III only
(E) I and III only

2. What output will be produced by

```
System.out.print("\\* This is not\n a comment *\\");
```

(A)
```
* This is not a comment *
```

(B)
```
\* This is not a comment *\
```

(C)
```
* This is not
a comment *
```

(D)
```
\\* This is not
a comment *\\
```

(E)
```
\* This is not
a comment *\
```

3. Consider the following code segment

```
if (n != 0 && x / n > 100)
    statement1;
else
    statement2;
```

If `n` is of type `int` and has a value of `0` when the segment is executed, what will happen?

(A) An `ArithmeticException` will be thrown.
(B) A syntax error will occur.
(C) ***statement1***, but not ***statement2***, will be executed.
(D) ***statement2***, but not ***statement1***, will be executed.
(E) Neither ***statement1*** nor ***statement2*** will be executed; control will pass to the first statement following the `if` statement.

4. Refer to the following code fragment:

```
double answer = 13 / 5;
System.out.println("13 / 5 = " + answer);
```

The output is

```
13 / 5 = 2.0
```

The programmer intends the output to be

```
13 / 5 = 2.6
```

Which of the following replacements for the first line of code will *not* fix the problem?

(A) `double answer = (double) 13 / 5;`
(B) `double answer = 13 / (double) 5;`
(C) `double answer = 13.0 / 5;`
(D) `double answer = 13 / 5.0;`
(E) `double answer = (double) (13 / 5);`

5. What value is stored in `result` if

```
int result = 13 - 3 * 6 / 4 % 3;
```

(A) −5
(B) 0
(C) 13
(D) −1
(E) 12

6. Suppose that addition and subtraction had higher precedence than multiplication and division. Then the expression

```
2 + 3 * 12 / 7 - 4 + 8
```

would evaluate to which of the following?

(A) 11
(B) 12
(C) 5
(D) 9
(E) −4

7. Which is true of the following boolean expression, given that `x` is a variable of type `double`?

```
3.0 == x * (3.0 / x)
```

(A) It will always evaluate to false.
(B) It may evaluate to false for some values of `x`.
(C) It will evaluate to false only when `x` is zero.
(D) It will evaluate to false only when `x` is very large or very close to zero.
(E) It will always evaluate to true.

8. Let `x` be a variable of type `double` that is positive. A program contains the boolean expression `(Math.pow(x,0.5) == Math.sqrt(x))`. Even though $x^{1/2}$ is mathematically equivalent to $\sqrt{x}$, the above expression returns the value `false` in a student's program. Which of the following is the most likely reason?
   (A) `Math.pow` returns an `int`, while `Math.sqrt` returns a `double`.
   (B) `x` was imprecisely calculated in a previous program statement.
   (C) The computer stores floating-point numbers with 32-bit words.
   (D) There is round-off error in calculating the `pow` and `sqrt` functions.
   (E) There is overflow error in calculating the `pow` function.

9. What will the output be for the following poorly formatted program segment, if the input value for `num` is 22?

```
int num =  call to a method that reads an integer;
if (num > 0)
if (num % 5 == 0)
System.out.println(num);
else System.out.println(num + " is negative");
```

   (A) `22`
   (B) `4`
   (C) `2 is negative`
   (D) `22 is negative`
   (E) Nothing will be output.

10. What values are stored in x and y after execution of the following program segment?

```
int x = 30, y = 40;
if (x >= 0)
{
    if (x <= 100)
    {
        y = x * 3;
        if (y < 50)
            x /= 10;
    }
    else
        y = x * 2;
}
else
    y = -x;
```

(A) x = 30 y = 90
(B) x = 30 y = -30
(C) x = 30 y = 60
(D) x = 3 y = -3
(E) x = 30 y = 40

11. Which of the following will evaluate to true only if boolean expressions `A`, `B`, and `C` are all false?

(A) `!A && !(B && !C)`
(B) `!A || !B || !C`
(C) `!(A || B || C)`
(D) `!(A && B && C)`
(E) `!A || !(B || !C)`

12. Assume that `a` and `b` are integers. The boolean expression

```
!(a <= b) && (a * b > 0)
```

will always evaluate to `true` given that

(A) `a = b`
(B) `a > b`
(C) `a < b`
(D) `a > b` and `b > 0`
(E) `a > b` and `b < 0`

13. Given that `a`, `b`, and `c` are integers, consider the boolean expression

```
(a < b) || !((c == a * b) && (c < a))
```

Which of the following will *guarantee* that the expression is `true`?

(A) `c < a` is `false`.
(B) `c < a` is `true`.
(C) `a < b` is `false`.
(D) `c == a * b` is `true`.
(E) `c == a * b` is `true`, and `c < a` is `true`.

14. In the following code segment, you may assume that `a`, `b`, and `n` are all type `int`.

```
if (a != b && n / (a - b) > 90)
{
    /* statement 1 */
}
else
{
    /* statement 2 */
}
/* statement 3 */
```

What will happen if `a == b` is false?

(A) `/* statement 1 */` will be executed.
(B) `/* statement 2 */` will be executed.
(C) Either `/* statement 1 */` or `/* statement 2 */` will be executed.
(D) A compile-time error will occur.
(E) An exception will be thrown.

15. Given that `n` and `count` are both of type `int`, which statement is true about the following code segments?

I
```
for (count = 1; count <= n; count++)
    System.out.println(count);
```

II
```
count = 1;
while (count <= n)
{
    System.out.println(count);
    count++;
}
```

(A) I and II are exactly equivalent for all input values `n`.
(B) I and II are exactly equivalent for all input values $n \geq 1$, but differ when $n \leq 0$.
(C) I and II are exactly equivalent only when $n = 0$.
(D) I and II are exactly equivalent only when `n` is even.
(E) I and II are not equivalent for any input values of `n`.

16. The following fragment intends that a user will enter a list of positive integers at the keyboard and terminate the list with a sentinel:

```
int value = 0;
final int SENTINEL = -999;
while (value != SENTINEL)
{
    //code to process value
        ...
    value = IO.readInt();       //read user input
}
```

The fragment is not correct. Which is a true statement?

(A) The sentinel gets processed.
(B) The last nonsentinel value entered in the list fails to get processed.
(C) A poor choice of SENTINEL value causes the loop to terminate before all values have been processed.
(D) The code will always process a value that is not on the list.
(E) Entering the SENTINEL value as the first value causes a run-time error.

17. Suppose that base-2 (binary) numbers and base-16 (hexadecimal) numbers can be denoted with subscripts, as shown below:

$$2A_{hex} = 101010_{bin}$$

Which is equal to $3D_{hex}$?

(A) 111101_{bin}
(B) 101111_{bin}
(C) 10011_{bin}
(D) 110100_{bin}
(E) 101101_{bin}

18. A common use of hexadecimal numerals is to specify colors on web pages. Every color has a red, green, and blue component. In decimal notation, these are denoted with an ordered triple (x, y, z), where x, y, and z are the three components, each an `int` from 0 to 255. For example, a certain shade of red, whose red, green, and blue components are 238, 9, and 63, is represented as $(238, 9, 63)$.

    In hexadecimal, a color is represented in the format #RRGGBB, where RR, GG, and BB are hex values for the red, green, and blue. Using this notation, the color $(238, 9, 63)$ would be coded as #EE093F.

    Which of the following hex codes represents the color $(14, 20, 255)$?

(A) #1418FE
(B) #0E20FE
(C) #0E14FF
(D) #0FE5FE
(E) #0D14FF

19. In Java, a variable of type `int` is represented internally as a 32-bit signed integer. Suppose that one bit stores the sign, and the other 31 bits store the magnitude of the number in base 2. In this scheme, what is the largest value that can be stored as type `int`?

(A) 2^{32}
(B) $2^{32} - 1$
(C) 2^{31}
(D) $2^{31} - 1$
(E) 2^{30}

$11 = 3 = 2^2 - 1$

20. Consider this code segment:

```
int x = 10, y = 0;
while (x > 5)
{
    y = 3;
    while (y < x)
    {
        y *= 2;
        if (y % x == 1)
            y += x;
    }
    x -= 3;
}
System.out.println(x + " " + y);
```

What will be output after execution of this code segment?

(A) `1  6`
(B) `7  12`
(C) `-3  12`
(D) `4  12`
(E) `-3  6`

Questions 21 and 22 refer to the following method, `checkNumber`, which checks the validity of its four-digit integer parameter.

```
/** @param n a 4-digit integer
 *  @return true if n is valid, false otherwise
 */
boolean checkNumber(int n)
{
    int d1,d2,d3,checkDigit,nRemaining,rem;
    //strip off digits
    checkDigit = n % 10;
    nRemaining = n / 10;
    d3 = nRemaining % 10;
    nRemaining /= 10;
    d2 = nRemaining % 10;
    nRemaining /= 10;
    d1 = nRemaining % 10;
    //check validity
    rem = (d1 + d2 + d3) % 7;
    return rem == checkDigit;
}
```

A program invokes method `checkNumber` with the statement

```
boolean valid = checkNumber(num);
```

21. Which of the following values of `num` will result in `valid` having a value of `true`?
    (A) `6143`
    (B) `6144`
    (C) `6145`
    (D) `6146`
    (E) `6147`

22. What is the purpose of the local variable `nRemaining`?
    (A) It is not possible to separate `n` into digits without the help of a temporary variable.
    (B) `nRemaining` prevents the parameter `num` from being altered.
    (C) `nRemaining` enhances the readability of the algorithm.
    (D) On exiting the method, the value of `nRemaining` may be reused.
    (E) `nRemaining` is needed as the left-hand side operand for integer division.

23. What output will be produced by this code segment? (Ignore spacing.)

```
for (int i = 5; i >= 1; i--)
{
    for (int j = i; j >= 1; j--)
        System.out.print(2 * j - 1);
    System.out.println();
}
```

(A)
```
9  7  5  3  1
9  7  5  3
9  7  5
9  7
9
```

(B)
```
9  7  5  3  1
7  5  3  1
5  3  1
3  1
1
```

(C)
```
9  7  5  3  1
7  5  3  1 -1
5  3  1 -1 -3
3  1 -1 -3 -5
1 -1 -3 -5 -7
```

(D)
```
1
1  3
1  3  5
1  3  5  7
1  3  5  7  9
```

(E)
```
1  3  5  7  9
1  3  5  7
1  3  5
1  3
1
```

24. Which of the following program fragments will produce this output? (Ignore spacing.)

```
2 - - - - -
- 4 - - - -
- - 6 - - -
- - - 8 - -
- - - - 10 -
- - - - - 12
```

I
```
for (int i = 1; i <= 6; i++)
{
    for (int k = 1; k <= 6; k++)
        if (k == i)
            System.out.print(2 * k);
        else
            System.out.print("-");
    System.out.println();
}
```

II
```
for (int i = 1; i <= 6; i++)
{
    for (int k = 1; k <= i - 1; k++)
        System.out.print("-");
    System.out.print(2 * i);
    for (int k = 1; k <= 6 - i; k++)
        System.out.print("-");
    System.out.println();
}
```

6 - i - 1 +1
6 -(i+1) +1

III
```
for (int i = 1; i <= 6; i++)
{
    for (int k = 1; k <= i - 1; k++)
        System.out.print("-");
    System.out.print(2 * i);
    for (int k = i + 1; k <= 6; k++)
        System.out.print("-");
    System.out.println();
}
```

(A) I only
(B) II only
(C) III only
(D) I and II only
(E) I, II, and III

25. Consider this program segment:

```
int newNum = 0, temp;
int num = k;            //k is some predefined integer value ≥ 0
while (num > 10)
{
    temp = num % 10;
    num /= 10;
    newNum = newNum * 10 + temp;
}
System.out.print(newNum);
```

Which is a true statement about the segment?

I If `100 ≤ num ≤ 1000` initially, the final value of `newNum` must be in the range `10 ≤ newNum ≤ 100`.
II There is no initial value of `num` that will cause an infinite `while` loop.
III If `num ≤ 10` initially, `newNum` will have a final value of `0`.

(A) I only
(B) II only
(C) III only
(D) II and III only
(E) I, II, and III

26. Consider the method `reverse`:

```
/** Precondition:  n > 0.
 *  Postcondition:
 *   - Returns n with its digits reversed.
 *   - Example: If n = 234, method reverse returns 432.
 *  @param n a positive integer
 *  @return n with its digits reversed
 */
int reverse(int n)
{
    int rem, revNum = 0;

    /* code segment */

    return revNum;
}
```

Which of the following replacements for `/* code segment */` would cause the method to work as intended?

I
```
for (int i = 0; i <= n; i++)
{
    rem = n % 10;
    revNum = revNum * 10 + rem;
    n /= 10;
}
```

II
```
while (n != 0)
{
    rem = n % 10;
    revNum = revNum * 10 + rem;
    n /= 10;
}
```

III
```
for (int i = n; i != 0; i /= 10)
{
    rem = i % 10;
    revNum = revNum * 10 + rem;
}
```

(A) I only
(B) II only
(C) I and II only
(D) II and III only
(E) I and III only