1. What is the output of the following program segment?

```
int num = 5;
while (num >= 0)
{
  num -= 2;
}
System.out.print(num);
```

(A) -2
(B) -1
(C) 0
(D) 2
(E) 21

2. Assuming that x and y are int variables, the expression

```
!(x > y && y <= 0)
```

is equivalent to which of the following?

(A) `!(x <= y) || (y > 0)`
(B) `x > y && y <= 0`
(C) `x <= y || y > 0`
(D) `x > y || y < 0`
(E) `x <= y && y <= 0`

3. Which of the following describes the return value of the following method?

```
// precondition:  amt represents a positive value in dollars
//                and cents (for example, 1.15 represents
//                one dollar and fifteen cents)
private int process(double amt)
{
  return (int)(amt * 100 + 0.5) % 100;
}
```

(A) the cent portion in `amt`
(B) the number of whole dollars in `amt`
(C) `amt` converted into cents
(D) `amt` rounded to the nearest integer
(E) `amt` truncated to the nearest integer

4. What is the output of the following code segment?

```
int sum = 0, d = -1;
for (int count = 10; count > 0; count--)
{
  sum += d;
  if (d > 0)
    d++;
  else
    d--;
  d = -d;
}
System.out.println(sum);
```

(A) 0
(B) 5
(C) -5
(D) 10
(E) -10

5. The following code segment is supposed to calculate and display 1 + 2 + ... + 20:

```
int count = 0, sum = 0;
while (count < 20)
{
  sum += count;
}
System.out.println(sum);
```

Which statement best describes the result:

(A) The total displayed will be correct.
(B) The total displayed will be 20 too small.
(C) The output will be the number 0.
(D) The output will be the number 20.
(E) There will be no output because the program goes into an infinite loop.

6. Consider the following class:

```
public class Rectangle implements Comparable<Rectangle>
{
  private int width, height;

  public Rectangle(int w, int h) { width = w; height = h; }
  public int getHeight() { return height; }
  public int getWidth() { return width; }
  public int getArea() { return width * height; }

  // ... other methods not shown
}
```

Suppose this class implements the `Comparable<Rectangle>` interface in such a way that `Rectangle` objects are compared based on their area: the rectangle with the smaller area is deemed smaller. Which of the following `compareTo` methods must be provided?

(A)

```
public boolean compareTo(int area)
{
  return getArea() < area;
}
```

(B)

```
public int compareTo(int area)
{
  return getArea() - area;
}
```

(C)

```
public int compareTo(Object other)
{
  return getArea() - other.getArea();
}
```

(D)

```
public boolean compareTo(Rectangle other)
{
  return getArea() < other.getArea();
}
```

(E)

```
public int compareTo(Rectangle other)
{
  return getArea() - other.getArea();
}
```

7. Which of the following Boolean expressions properly implement a comparison for equality of two `String` objects `str1` and `str2` and evaluate to `true` if and only if `str1` and `str2` hold the same values?

I. `str1 == str2`

II. `str1.equals(str2)`

III. `str1.compareTo(str2) == 0`

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

8. What is the output of the following code segment?

```
int a = 3;
int b = 4;
int c = 0;

if (a == b && b/c == 1)
{
  c = a * b;
}
else
{
  c = a + b * c;
  System.out.println(c);
}
```

(A) Run-time division-by-zero error
(B) 0
(C) 3
(D) 6
(E) 12

9. Which of the following statements about overloaded methods is FALSE?

(A) Overloaded methods must be made either all public or all private.
(B) Overloaded methods are defined in the same class and have the same name.
(C) Overloaded methods may have the same number of parameters.
(D) One of the overloaded methods may take no parameters.
(E) Overloaded methods cannot be differentiated based only on the names chosen for their parameters.

10. Consider the following class:

```
public class Sphere
{
  public static final double pi = 3.14159;

  public static double volume(int r)
  {
    return 4 / 3 * pi * Math.pow(r, 3);
  }
}
```

Which of the following statements about this code is true?

(A) The class will not compile because no constructors are defined.
(B) The class will not compile because `pi` cannot be declared `public`.
(C) The class will not compile because the `volume` method is declared `static`.
(D) `Math.pow(r, 3)` cannot be used because `r` is an `int`.
(E) The class compiles with no errors but the `volume` method returns a significantly smaller value than the expected $\frac{4}{3}\pi r^3$.

11. Consider the following method:

```
// precondition: a != null; a.length > 0
private static void doIt(double[] a)
{
  double temp;

  for (int k = 0; k < a.length / 2; k++)
  {
    temp = a[k];
    a[k] = a[a.length - 1 - k];
    a[a.length - 1 - k] = temp;
  }
}
```

Which of the following best describes the task performed by this method?

(A) Sorts an array in ascending order
(B) Sorts an array in descending order
(C) Swaps the first and last elements of an array
(D) Reverses the order of elements in an array
(E) None of the above tasks is implemented correctly.

12. Consider the following method:

```
public String filter(String str, String pattern)
{
  int pos = str.indexOf(pattern);
  if (pos == -1)
    return str;
  else
    return filter(str.substring(0, pos) +
             str.substring(pos + pattern.length()), pattern);
}
```

What is the output of

```
System.out.println(filter("papaya", "pa"));
```

(A) p
(B) pa
(C) ya
(D) aya
(E) paya

13. Consider the following code segment:

```
int n = IO.readInt();    // read an int value
n = Math.abs(n);

while (n >= 2)
{
  n = n/2 - 1;
}
System.out.println(n);
```

Which of the following is the list of all possible outputs?

(A) 0
(B) -1, 0
(C) 0, 1
(D) -1, 1
(E) -1, 0, 1

14. Which of the following recommendations for testing software is NOT good advice?

(A) Test a program with all possible values of input data.
(B) When testing a large program, test the smaller pieces individually before testing the entire program.
(C) If possible, use automated testing procedures or read test data from files so that you can re-run the tests after corrections have been made.
(D) Design test data that exercises as many different paths through the code as is practical.
(E) Test on data that is at the boundary of program conditionals to check for "off by one" errors.

15. Whitney is a cheerleader and a programmer. She has written the following recursive method that is supposed to generate the cheer "2 4 6 8 Who do we appreciate!":

```
public void cheer(int i)
{
  if (i != 8)                                    // Line 1
  {                                              // Line 2
    i = i + 2;                                   // Line 3
    cheer(i);                                    // Line 4
    System.out.print(i + " ");                   // Line 5
  }                                              // Line 6
  else                                           // Line 7
  {                                              // Line 8
    System.out.print("Who do we appreciate!");  // Line 9
  }                                              // Line 10
}
```

However, Whitney's method doesn't work as expected when she calls `cheer(0)`. To get the right cheer, Whitney should

(A) replace `if (i != 8)` with `if (i <= 8)` on Line 1
(B) replace `if (i != 8)` with `if (i == 8)` on Line 1
(C) replace `if (i != 8)` with `while (i != 8)` on Line 1
(D) swap Line 4 and Line 5
(E) move Line 3 after Line 5

16. What is displayed when the following method is called with `splat("**")`?

```
public void splat(String s)
{
  if (s.length() < 8)
    splat(s + s);
  System.out.println(s);
}
```

(A)
```
**
```

(B)
```
****
```

(C)
```
********
```

(D)
```
********
**
```

(E)
```
********
****
**
```

Questions 17-18 refer to the following `sortX` method:

```
public void sortX(int[] a)
{
  for (int i = 1; i < a.length; i++) // Line 1
  {
    int current = a[i];              // Line 2
    int j = 0;                       // Line 3

    while (a[j] < current)           // Line 4
    {
      j++;                           // Line 5
    }
    for (int k = i; k > j; k--)      // Line 6
    {
      a[k] = a[k-1];                 // Line 7
    }
    a[j] = current;                  // Line 8
  }
}
```

17. The sorting algorithm implemented in the `sortX` method can be best described as:

    (A) Selection Sort
    (B) Insertion Sort
    (C) Quicksort
    (D) Mergesort
    (E) Incorrect implementation of a sorting algorithm

18. Given

    ```
    int[] a = {24, 16, 68, 56, 32};
    ```

    what will be the result after the statement on Line 8 in `sortX` completes for the second time?

    (A) The values in `a` are 16, 24, 68, 56, 32
    (B) The values in `a` are 16, 24, 32, 56, 68
    (C) The values in `a` are 24, 16, 32, 56, 68
    (D) The code has failed with an `ArrayIndexOutOfBoundsException` on Line 4
    (E) The code has failed with an `ArrayIndexOutOfBoundsException` on Line 8

19. Consider the following class:

```
public class BuddyList
{
  private ArrayList<String> buddies;  // contains the names
                                      //   of buddies

  // ... constructors and other methods and variables not shown

  public ArrayList<String> getBuddies() { return buddies; }
}
```

If `BuddyList myFriends` is declared and initialized in some other class, a client of `BuddyList`, which of the following correctly assigns to `name` the name of the first buddy in the `myFriends` list?

I. `String name = myFriends.buddies[0];`

II. `String name = myFriends.buddies.get(0);`

III. `String name = myFriends.getBuddies().get(0);`

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

20. The class `PlayList` provides methods that allow you to represent and manipulate a list of tunes, but you are not concerned with how these operations work or how the list is stored in memory. You only know how to initialize and use `PlayList` objects and have no direct access to the implementation of the `PlayList` class or its private data fields. This is an example of:

(A) encapsulation
(B) overriding
(C) inheritance
(D) polymorphism
(E) method overloading

21. Which of the following statements about constructors is NOT true?

(A) All constructors must have the same name as the class they are in.
(B) Constructors' return type must be declared `void`.
(C) A class may have a constructor that takes no parameters.
(D) A constructor is invoked when a program creates an object with the `new` operator.
(E) A constructor of a subclass can call a constructor of its superclass using the Java reserved word `super`.

22. Consider the following method, intended to use Binary Search to find the location of `target` within an `ArrayList a`:

```
public int findLocation(ArrayList<String> a, String target)
{
  int first = 0, last = a.size() - 1;
  while (first <= last)
  {
    int middle = (first + last) / 2;
    int compResult = target.compareTo(a.get(middle));
    if (compResult == 0)
      return middle;
    if (compResult < 0)
      last = middle - 1;
    else
      first = middle + 1;
  }
  return -1;
}
```

This method may fail if it is applied to a list that is not sorted. For which of the following lists will `findLocation(a, "C")` return `-1`?

(A) `"A", "B", "C", "D", "E", "F", "G"`
(B) `"G", "F", "E", "D", "C", "B", "A"`
(C) `"A", "C", "D", "G", "E", "B", "F"`
(D) `"B", "A", "D", "C", "F", "E", "G"`
(E) `"D", "F", "B", "A", "G", "C", "E"`

Questions 23 and 24 refer to the following class:

```
public class Sample
{
  private double[] amps;

  public Sample(int n) { < missing statements > }
  public double get(int k) { return amps[k]; }
}
```

23. Which of the following code segments can replace < *missing statements* > in `Sample`'s constructor so that it initializes `amps` to hold `n` values and fills them with random values $0.0 \leq$ `amps[i]` < 1.0?

(A)
```
amps = new double[n];
```

(B)
```
amps = new double[n];
for (int k = 0; k < n; k++)
  amps[k] = new Random();
```

(C)
```
amps = new double[n];
for (int k = 0; k < n; k++)
  amps[k] = Random.nextDouble();
```

(D)
```
amps = new double[n];
Random randGen = new Random();
for (int k = 0; k < n; k++)
  amps[k] = randGen.nextDouble();
```

(E)
```
amps = new Double[n];
Random randGen = new Random();
for (int k = 0; k < n; k++)
  amps[k] = new Double(randGen.nextDouble());
```

24. Given

```
int size = 100;
Sample s = new Sample(size);
```

which of the following statements assigns to `x` the last value in `amps` from `s`?

(A) `double x = s[99];`
(B) `double x = s.amps[amps.length - 1];`
(C) `double x = s.get(amps.length - 1);`
(D) `double x = s.get(size - 1);`
(E) `double x = s.get[s.length - 1];`

25. A project needs two related classes, *X* and *Y*. A programmer has decided to provide an abstract class *A* and derive both *X* and *Y* from *A* rather than implementing *X* and *Y* completely independently of each other. Which of the following is NOT a valid rationale for this design decision?

(A) Being able to use some common code accessible in classes *X* and *Y* without duplication
(B) Being able to cast objects of type *X* into *Y* and vice-versa
(C) Being able to pass as a parameter an object of either type, *X* or *Y*, to the same constructor or method in place of a parameter of the type *A*.
(D) Being able to place objects of both types, *X* and *Y*, into the same array of type `A[]`
(E) Making it easier to implement in the future another class that reuses some code from *A*

26. Suppose `ArrayList<Integer> numbers` and `ArrayList<String> names` are created as follows:

```
ArrayList<Integer> numbers = new ArrayList<Integer>();
Integer x = new Integer(1);
numbers.add(x);
numbers.add(x);

ArrayList<String> names = new ArrayList<String>();
names.add(0, "Anya");
names.add(0, "Ben");
names.add(0, "Cathy");
```

What is the result of the following code segment?

```
for (Integer i : numbers)
{
  names.remove(i.intValue());
}
for (String name : names)
{
  System.out.print(name + " ");
}
```

(A) `Cathy`
(B) `Cathy Anya`
(C) `Anya Cathy`
(D) `IndexOutOfBoundsException`
(E) `NoSuchElementException`

27. When does a class have to be declared `abstract`?

(A) When it has no constructors
(B) When it has no public methods
(C) When the class has no public or private instance variables
(D) When you need to derive another class from this class
(E) When one or more methods in the class are declared abstract

28. Consider the following interface `TV` and class `MyTV`:

```
public interface TV
{
  void tuneTo(String channel);
}

public class MyTV implements TV
{
  private ArrayList<String> myFavoriteChannels;

  public MyTV(ArrayList<String> channels)
  { /* implementation not shown */ }

  public void tuneTo(int k)
  { /* implementation not shown */ }

  public void tuneTo(int k, String name)
  { /* implementation not shown */ }
}
```

One of them has one or more errors and won't compile properly. Which of the following best describes the compiler errors reported for the statements that are shown?

(A) In the `TV` interface, the `tuneTo` declaration is missing the keyword `public`
(B) `MyTV` should be declared abstract; it does not define `tuneTo(String)` in `MyTV`
(C) `tuneTo` is defined more than once in `MyTV`
(D) Cannot convert `int` to `String` in the `tuneTo` method in `MyTV`
(E) Two errors: (1) `tuneTo` is defined more than once and (2) cannot convert `int` to `String` in the `tuneTo(int)` method in `MyTV`

29. Consider the following code segment:

```
if (!somethingIsFalse())
  return false;
else
  return true;
```

Which of the following replacements for this code will produce the same result?

(A) `return true;`
(B) `return false;`
(C) `return !somethingIsFalse();`
(D) `return somethingIsFalse();`
(E) none of the above

30. Consider the following class definitions:

```
public class Airplane
{
  private int fuel;

  public Airplane() { fuel = 0; }
  public Airplane(int g) { fuel = g; }

  public void addFuel() { fuel++; }
  public String toString() { return fuel + " "; }
}

public class Jet extends Airplane
{
  public Jet(int g) { super(2*g); }
}
```

What is the result when the following code is compiled and run?

```
Airplane plane = new Airplane(4);
Airplane jet = new Jet(4);
System.out.print(plane);
plane.addFuel();
System.out.print(plane);
System.out.print(jet);
jet.addFuel();
System.out.print(jet);
```

(A) A syntax error, "undefined addFuel," is reported for the `jet.addFuel();` statement.
(B) A run-time error, `ClassCastException`, occurs when `jet.addFuel()` is attempted.
(C) The code compiles and runs with no errors; the output is `4 5 5 6`
(D) The code compiles and runs with no errors; the output is `4 5 8 9`
(E) The code compiles and runs with no errors; the output is `8 9 9 10`

31. A programmer wants to create a swap method that swaps two integer values. Which of the following three ways of representing the values and corresponding methods successfully swap the values?

I.
```
// a and b are Integer objects that represent the values
// to be swapped
public static void swap(Integer a, Integer b)
{
  Integer temp = a; a = b; b = temp;
}
```

II.
```
// a[0] and a[1] contain the values to be swapped
public static void swap(int[] a)
{
  int temp = a[0]; a[0] = a[1]; a[1] = temp;
}
```

III.
```
// a[0] and b[0] contain the values to be swapped
public static void swap(int[] a, int[] b)
{
  int temp = a[0]; a[0] = b[0]; b[0] = temp;
}
```

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

32. Suppose an interface `Solid` specifies the `getVolume()` method. Two classes, `Cube` and `Pyramid`, implement `Solid`. Which Java feature makes it possible for the following code segment to print the correct values for the volume of a pyramid and a cube?

```
Solid[] solids = new Solid[2];
solids[0] = new Cube(100);
solids[1] = new Pyramid(150, 100);
System.out.println("Cube: " + solids[0].getVolume());
System.out.println("Pyramid: " + solids[1].getVolume());
```

(A) abstraction
(B) encapsulation
(C) polymorphism
(D) platform-independence
(E) method overloading

33. Consider the following code segment:

```
ArrayList<String> list = new ArrayList<String>();
list.add("One");
list.add("Two");
String[] msg = new String[2];
list.add(msg[0]);
< another statement >
```

Which of the following choices for < *another statement* > will cause a `NullPointerException`?

(A) `msg[0] = "Three";`
(B) `msg[0] = list.get(list.size());`
(C) `if (!"Three".equals(list.get(2))) msg[0] = "Three";`
(D) `list.add(2, msg[0]);`
(E) `msg[1] = msg[0].substring(0, 2);`

34. A programmer is trying to choose between an `ArrayList` and a standard one-dimensional array for representing data. Which of the following is NOT a correct statement?

(A) Both an `ArrayList` and a standard array allow direct access to the *k*-th element.
(B) A standard array may hold elements of a primitive data type, such as `int` or `double`; an `ArrayList` may only hold objects.
(C) An `ArrayList` may hold objects of different types, such as `Integer` and `Double`, simultaneously.
(D) An `ArrayList` has a convenient method for inserting a value at a specified location in the middle.
(E) Both an `ArrayList` and a standard array are expanded automatically when the number of values stored exceeds their size.

Questions 35-40 refer to the code from the GridWorld case study.

35. Which of the following code segments will compile with no errors?

I.
```
BoxBug bb = new BoxBug();
if (!bb.canMove())
  bb.turn();
```

II.
```
BoxBug bb = new BoxBug(5);
bb.setColor(Color.BLUE);
```

III.
```
BoxBug bb = new BoxBug(Color.BLUE);
bb.move();
```

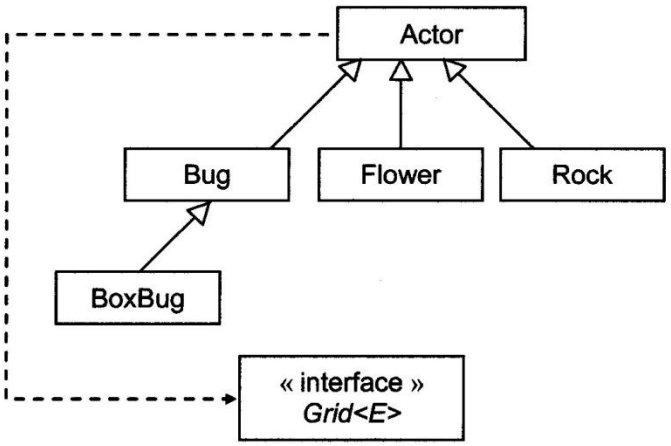
36. Assuming that `loc1` and `loc2` are `Location` objects that represent valid locations in a grid, which of the following conditions verifies that `loc2` lies to the north of `loc1`?

(A) `Grid.getDirection(loc1, loc2).equals(Location.NORTH)`
(B) `loc1.getAdjacentLocation(Location.NORTH)).equals(loc2)`
(C) `loc1.getDirectionTowardLocation(loc2) == Location.NORTH`
(D) `loc1.getDirectionTowardLocation(loc2).equals(Location.NORTH)`
(E) `loc1.getDirection(loc2).equals(new Location(Location.NORTH, 0))`

37. Suppose we want to create a variation of `Bug` that acts like a regular `Bug` but turns 45 degrees randomly left or right after each move. Which of the following is the most economical approach, in terms of the amount of code to be written?

(A) Extend `Actor` and override the `act` method
(B) Extend `Bug` and override the `move` method
(C) Extend `Bug` and override the `turn` method
(D) Extend `Bug` and override both `move` and `turn` methods
(E) Extend `Bug` and override both `move` and `canMove` methods

38. The diagram below shows the interactions between some of the GridWorld classes and interfaces.



An —▷ arrow from *A* to *B* indicates that *A* extends *B*. An - - → arrow from *A* to *B* indicates that *A* uses *B*; that is, *A* refers directly to variables of the type *B*. Two "uses" arrows can be added to the diagram to make it more accurate. Which ones?

(A) `Bug` uses `Grid` and `Bug` uses `Flower`
(B) `Bug` uses `Flower` and `Bug` uses `Rock`
(C) `Bug` uses `Grid` and `Flower` uses `Grid`
(D) `Bug` uses `Grid` and `BoxBug` uses `Grid`
(E) `Bug` uses `Grid` and `BoxBug` uses `Flower`

39. Given

```
Rock rock = new Rock();
```

and assuming that all the necessary `import` statements are present, which of the following statements will cause a syntax error?

(A) `rock.moveTo(null);`
(B) `rock.setDirection(0);`
(C) `rock.setColor(Color.GRAY);`
(D) `rock.setGrid(null);`
(E) `Grid<Actor> gr = rock.getGrid();`

40. Which of `Critter`'s methods calls `selectMoveLocation`?

(A) `act`
(B) `processActors`
(C) `makeMove`
(D) `moveTo`
(E) None of the above