1. Assume that `a`, `b`, and `c` are variables of type `int`. Consider the following three conditions.

   I. `(a == b) && (a == c) && (b == c)`

   II. `(a == b) || (a == c) || (b == c)`

   III. `((a - b) * (a - c) * (b - c)) == 0`

   Assume that subtraction and multiplication never overflow. Which of the conditions above is (are) always true if at least two of `a`, `b`, and `c` are equal?

   (A) I only

   (B) II only

   (C) III only

   (D) I and II

   (E) II and III

---

2. Consider the following two data structures for storing several million words.

   I. An array of words, not in any particular order

   II. An array of words, sorted in alphabetical order

   Which of the following statements most accurately describes the time needed for operations on these data structures?

   (A) Inserting a word is faster in II than in I.

   (B) Finding a given word is faster in I than in II.

   (C) Finding a given word is faster in II than in I.

   (D) Finding the longest word is faster in II than in I.

   (E) Finding the first word in alphabetical order is faster in I than in II.

---

3. Which of the following is a reason to use an `ArrayList` instead of an array?

   (A) An `ArrayList` allows faster access to its *k*th item than an array does.

   (B) An `ArrayList` always uses less memory than an array does.

   (C) An `ArrayList` can store objects and an array can only store primitive types.

   (D) An `ArrayList` resizes itself as necessary when items are added, but an array does not.

   (E) An `ArrayList` provides access to the number of items it stores, but an array does not.

4. Consider the following method, `between`, which is intended to return `true` if `x` is between `lower` and `upper`, inclusive, and `false` otherwise.

```
// precondition:  lower <= upper
// postcondition: returns true if x is between lower and upper,
//                inclusive; otherwise, returns false
public boolean between(int x, int lower, int upper)
{
  /* missing code */
}
```

Which of the following can be used to replace `/* missing code */` so that `between` will work as intended?

(A) `return (x <= lower) && (x >= upper);`

(B) `return (x <= lower) || (x >= upper);`

(C) `return lower <= x <= upper;`

(D) `return (x >= lower) && (x <= upper);`

(E) `return (x >= lower) || (x <= upper);`

5. Consider the following class definitions.

```
public class Data
{
  private int x;

  public void setX(int n)
  {
    x = n;
  }

  // ... other methods not shown

}

public class EnhancedData extends Data
{
  private int y;

  public void setY(int n)
  {
    y = n;
  }

  // ... other methods not shown

}
```

Assume that the following declaration appears in a client program.

```
EnhancedData item = new EnhancedData();
```

Which of the following statements would be valid?

I. `item.y = 16;`

II. `item.setY(16);`

III. `item.setX(25);`

(A) I only

(B) II only

(C) I and II only

(D) II and III only

(E) I, II, and III

**Questions 6-7 refer to the following data field and method.**

```
private int[] arr;

// precondition: arr.length > 0
public void mystery()
{
  int s1 = 0;
  int s2 = 0;

  for (int k = 0; k < arr.length; k++)
  {
    int num = arr[k];

    if ((num > 0) && (num % 2 == 0))
      s1 += num;
    else if (num < 0)
      s2 += num;
  }

  System.out.println(s1);
  System.out.println(s2);
}
```

6. Which of the following best describes the value of `s1` output by the method `mystery` ?

   (A) The sum of all positive values in `arr`

   (B) The sum of all positive even values in `arr`

   (C) The sum of all positive odd values in `arr`

   (D) The sum of all values greater than 2 in `arr`

   (E) The sum of all values less than 2 in `arr`

---

7. Which of the following best describes the value of `s2` output by the method `mystery` ?

   (A) The sum of all positive values in `arr`

   (B) The sum of all positive even values in `arr`

   (C) The sum of all negative values in `arr`

   (D) The sum of all negative even values in `arr`

   (E) The sum of all negative odd values in `arr`

8. When designing classes, which of the following would be the best reason to use inheritance?

   (A) Inheritance allows you to write applications that require fewer base and super classes.

   (B) Inheritance allows the creation of a subclass that can use the methods of its superclass without rewriting the code for those methods.

   (C) Inheritance allows for data encapsulation, while noninherited classes do not allow for data encapsulation.

   (D) Inheritance reduces the number of polymorphic structures encapsulated in applications.

   (E) Inheritance guarantees that the applications will compile and execute much more quickly.

---

9. Consider the following method.

```
public void conditionalTest(int a, int b)
{
  if ((a > 0) && (b > 0))
  {
    if (a > b)
      System.out.println("A");
    else
      System.out.println("B");
  }
  else if ((b < 0) || (a < 0))
    System.out.println("C");
  else
    System.out.println("D");
}
```

What is printed as a result of the call `conditionalTest(3, -2)` ?

(A) `A`

(B) `B`

(C) `C`

(D) `D`

(E) Nothing is printed.

10. Consider the following class declaration.

```
public class IntCell
{
  private int myStoredValue;

  // constructor not shown

  public int getValue()
  {
    return myStoredValue;
  }

  public String toString()
  {
    return "" + myStoredValue;
  }
}
```

Assume that the following declaration appears in a client class.

```
IntCell m = new IntCell();
```

Which of these statements can be used in the client class?

I. `System.out.println(m.getValue());`

II. `System.out.println(m.myStoredValue);`

III. `System.out.println(m);`

(A) I only

(B) II only

(C) III only

(D) I and II

(E) I and III

11. Consider the following static method.

```
public static int calculate(int x)
{
  x = x + x;
  x = x + x;
  x = x + x;

  return x;
}
```

Which of the following can be used to replace the body of `calculate` so that the modified version of `calculate` will return the same result as the original version for all `x` ?

(A) `return 3 + x;`

(B) `return 3 * x;`

(C) `return 4 * x;`

(D) `return 6 * x;`

(E) `return 8 * x;`

---

12. Consider the following code segment.

```
int k = 1;

while (k < 20)
{
  if ((k % 3) == 1)
    System.out.print(k + "  ");

  k++;
}
```

What is printed as a result of executing this code segment?

(A) 2 5 8 11 14 17

(B) 3 6 9 12 15 18

(C) 1 4 7 10 13 16 19

(D) 1 3 5 7 9 11 13 15 17 19

(E) 2 4 6 8 10 12 14 16 18 20

13. Consider the following code segment. The code is intended to read nonnegative numbers and compute their product until a negative number is read; however, it does not work as intended. (Assume that the `readInt` method correctly reads the next number from the input stream.)

```
int k = 0;
int prod = 1;

while (k >= 0)
{
  System.out.print("enter a number: ");
  k = readInt(); // readInt reads the next number from input
  prod = prod * k;
}

System.out.println("product: " + prod);
```

Which of the following best describes the error in the program?

(A) The variable `prod` is incorrectly initialized.

(B) The while condition always evaluates to `false`.

(C) The while condition always evaluates to `true`.

(D) The negative number entered to signal no more input is included in the product.

(E) If the user enters a zero, the computation of the product will be terminated prematurely.

14. Consider the following declarations.

```
int valueOne, valueTwo;
```

Assume that `valueOne` and `valueTwo` have been initialized. Which of the following evaluates to `true` if `valueOne` and `valueTwo` contain the same value?

(A) `valueOne.equals((Object) valueTwo)`

(B) `valueOne == valueTwo`

(C) `valueOne.compareTo((Object) valueTwo) == 0`

(D) `valueOne.compareTo(valueTwo) == 0`

(E) `valueOne.equals(valueTwo)`

---

15. Consider the following method.

```
public int someCode(int a, int b, int c)
{
  if ((a < b) && (b < c))
    return a;
  if ((a >= b) && (b >= c))
    return b;
  if ((a == b) || (a == c) || (b == c))
    return c;
}
```

Which of the following best describes why this method does not compile?

(A) The reserved word `return` cannot be used in the body of an `if` statement.

(B) It is possible to reach the end of the method without returning a value.

(C) The `if` statements must have `else` parts when they contain `return` statements.

(D) Methods cannot have multiple `return` statements.

(E) The third `if` statement is not reachable.

16. Consider the following code segment.

```
int num1 = 0;
int num2 = 3;

while ((num2 != 0) && ((num1 / num2) >= 0))
{
  num1 = num1 + 2;
  num2 = num2 - 1;
}
```

What are the values of `num1` and `num2` after the while loop completes its execution?

(A) `num1 = 0, num2 = 3`

(B) `num1 = 8, num2 = -1`

(C) `num1 = 4, num2 = 1`

(D) `num1 = 6, num2 = 0`

(E) The loop will never complete its execution because a division by zero will generate an `ArithmeticException`.

17. A bear is an animal and a zoo contains many animals, including bears. Three classes `Animal`, `Bear`, and `Zoo` are declared to represent animal, bear, and zoo objects. Which of the following is the most appropriate set of declarations?

(A)
```
public class Animal extends Bear
{
  ...
}

public class Zoo
{
  private Animal[] myAnimals;
  ...
}
```

(B)
```
public class Bear extends Animal
{
  ...
}

public class Zoo
{
  private Animal[] myAnimals;
  ...
}
```

(C)
```
public class Animal extends Zoo
{
  private Bear myBear;
  ...
}
```

(D)
```
public class Bear extends Animal, Zoo
{
  ...
}
```

(E)
```
public class Bear extends Animal implements Zoo
{
  ...
}
```

18. Consider the static method `selectSort` shown below. Method `selectSort` is intended to sort an array into increasing order; however, it does not always work as intended.

```
// precondition:  numbers.length > 0
// postcondition: numbers is sorted in increasing order
public static void selectSort(int[] numbers)
{
  int temp;
Line 1:  for (int j = 0; j < numbers.length - 1; j++)
  {
Line 2:    int pos = 0;
Line 3:    for (int k = j + 1; k < numbers.length; k++)
    {
Line 4:      if (numbers[k] < numbers[pos])
      {
Line 5:        pos = k;
      }
    }
    temp = numbers[j];
    numbers[j] = numbers[pos];
    numbers[pos] = temp;
  }
}
```

Which of the following changes should be made so that `selectSort` will work as intended?

(A) Line 1 should be changed to
```
for (int j = 0; j < numbers.length - 2; j++)
```

(B) Line 2 should be changed to
```
int pos = j;
```

(C) Line 3 should be changed to
```
for (int k = 0; k < numbers.length; k++)
```

(D) Line 4 should be changed to
```
if (numbers[k] > numbers[pos])
```

(E) Line 5 should be changed to
```
k = pos;
```

19. Consider the following two static methods, where `f2` is intended to be the iterative version of `f1`.

```
public static int f1(int n)
{
  if (n < 0)
  {
    return 0;
  }
  else
  {
    return (f1(n - 1) + n * 10);
  }
}

public static int f2(int n)
{
  int answer = 0;

  while (n > 0)
  {
    answer = answer + n * 10;
    n--;
  }

  return answer;
}
```

The method `f2` will always produce the same results as `f1` under which of the following conditions?

I. $n < 0$

II. $n = 0$

III. $n > 0$

(A) I only

(B) II only

(C) III only

(D) II and III only

(E) I, II, and III

20. Consider the following class definitions.

```
public class A
{
  private int a1;

  public void methodA()
  {
    methodB();          // Statement I
  }
}

public class B extends A
{
  public void methodB()
  {
    methodA();          // Statement II
    a1 = 0;             // Statement III
  }
}
```

Which of the labeled statements in the methods shown above will cause a compile-time error?

(A) I only

(B) III only

(C) I and II

(D) I and III

(E) II and III

21. Consider the following class definitions.

```
public class Animal
{
  public void eat()
  { /* implementation not shown */ }

  // constructors and other methods not shown

}

public class Tiger extends Animal
{
  public void roar()
  { /* implementation not shown */ }

  // constructors and other methods not shown

}
```

Assume that the following declaration appears in a client class.

```
Animal a = new Tiger();
```

Which of the following statements would compile without error?

I. `a.eat();`

II. `a.roar();`

III. `((Tiger) a).roar();`

(A) I only

(B) II only

(C) III only

(D) I and III only

(E) I, II, and III

---

22. Assume `obj1` and `obj2` are object references. Which of the following best describes when the expression `obj1 == obj2` is true?

(A) When `obj1` and `obj2` are defined within the same method

(B) When `obj1` and `obj2` are instances of the same class

(C) When `obj1` and `obj2` refer to objects that contain the same data

(D) When `obj1` and `obj2` refer to the same object

(E) When `obj1` and `obj2` are private class variables defined in the same class

23. Assume `mat` is defined as follows.

```
int dim = 4;
int[][] mat = new int[dim][dim];
```

Consider the following code segment.

```
int sum = 0;

for (int row = 0; row < dim; row++)
{
  sum = sum + mat[row][dim - 1];
}
```

Assume that `mat` contains the following values before the code segment is executed. Note that `mat[0][3]` is 2.

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| **0** | 1 | 1 | 2 | 2 |
| **1** | 1 | 2 | 2 | 4 |
| **2** | 1 | 3 | 2 | 6 |
| **3** | 1 | 4 | 2 | 8 |

What value will `sum` contain after the code segment is executed?

(A) 6

(B) 8

(C) 13

(D) 15

(E) 20

**24.** Consider the following data field and method.

```
private ArrayList list;

public void mystery(int n)
{
  for (int k = 0; k < n; k++)
  {
    Object obj = list.remove(0);
    list.add(obj);
  }
}
```

Assume that `list` has been initialized with the following `Integer` objects.

```
[12, 9, 7, 8, 4, 3, 6, 11, 1]
```

Which of the following represents the list as a result of a call to `mystery(3)` ?

(A) `[12, 9, 8, 4, 3, 6, 11, 1, 7]`

(B) `[12, 9, 7, 8, 4, 6, 11, 1, 3]`

(C) `[12, 9, 7, 4, 3, 6, 11, 1, 8]`

(D) `[8, 4, 3, 6, 11, 1, 12, 9, 7]`

(E) `[1, 11, 6, 12, 9, 7, 8, 4, 3]`

25. Consider the following data field and method.

```
private int[][] mat;

public void mystery()
{
  for (int row = 1; row < mat.length; row++)
  {
    for (int col = 0; col < mat[0].length; col++)
    {
      if (row != col)
        mat[row][col] = mat[row - 1][col];
    }
  }
}
```

Assume that `mat` contains the following values. Note that `mat[0][4]` is 2.

```
4  1  3  4  2
1  8  7  5  3
7  4  6  9  2
3  8  1  2  4
5  6  7  0  3
```

What values does `mat` contain after a call to `mystery`?

(A)
```
4  1  3  4  2
4  8  3  4  2
4  8  6  4  2
4  8  6  2  2
4  8  6  2  3
```

(B)
```
4  1  3  4  2
4  1  3  4  2
4  1  3  4  2
4  1  3  4  2
4  1  3  4  2
```

(C)
```
4  1  3  4  2
4  1  3  4  2
1  8  7  5  3
7  4  6  9  2
3  8  1  2  4
```

(D)
```
4  4  4  4  4
1  1  1  1  1
7  7  7  7  7
3  3  3  3  3
5  5  5  5  5
```

(E)
```
4  8  6  2  3
4  8  6  2  3
4  8  6  2  3
4  8  6  2  3
4  8  6  2  3
```

26. Assume that methods `f` and `g` are defined as follows.

```
public int f(int x)
{
  if (x <= 0)
  {
    return 0;
  }
  else
  {
    return g(x - 1);
  }
}

public int g(int x)
{
  if (x <= 0)
  {
    return 0;
  }
  else
  {
    return (f(x - 1) + x);
  }
}
```

What value is returned as a result of the call `f(6)` ?

(A) 0

(B) 3

(C) 6

(D) 9

(E) 12

27. Assume that `class Vehicle` contains the following method.

```
public void setPrice(double price)
{ /* implementation not shown */ }
```

Also assume that `class Car extends Vehicle` and contains the following method.

```
public void setPrice(double price)
{ /* implementation not shown */ }
```

Assume `Vehicle v` is initialized as follows.

```
Vehicle v = new Car();
v.setPrice(1000.0);
```

Which of the following is true?

(A) The code above will cause a compile-time error, because a subclass cannot have a method with the same name and the same signature as its superclass.

(B) The code above will cause a run-time error, because a subclass cannot have a method with the same name and the same signature as its superclass.

(C) The code above will cause a compile-time error because of type mismatch.

(D) The code `v.setPrice(1000.0);` will cause the `setPrice` method of the `Car` class to be called.

(E) The code `v.setPrice(1000.0);` will cause the `setPrice` method of the `Vehicle` class to be called.

28. Consider the following static method.

```
private static void recur(int n)
{
  if (n != 0)
  {
    recur(n - 2);
    System.out.print(n + " ");
  }
}
```

What numbers will be printed as a result of the call `recur(7)` ?

(A) `-1  1  3  5  7`

(B) `1  3  5  7`

(C) `7  5  3  1`

(D) Many numbers will be printed because of infinite recursion.

(E) No numbers will be printed because of infinite recursion.

---

29. Integers can be represented using different bases. Base 10 (decimal), and base 16 (hexadecimal) are indicated with the subscripts $_{dec}$ and $_{hex}$, respectively. For example, the decimal number 23 can also be represented in base 16 as shown below.

$23_{dec} = 17_{hex}$

Which of the following is equal to $100_{hex} - 10_{hex}$?

(A) $15_{dec}$

(B) $90_{dec}$

(C) $144_{dec}$

(D) $240_{dec}$

(E) $256_{dec}$

**Questions 30-31 are based on the following incomplete declaration of the class `BoundedIntArray` and its constructor definitions.**

A `BoundedIntArray` represents an indexed list of integers. In a `BoundedIntArray` the user can specify a size, in which case the indices range from `0` to `size - 1`. The user can also specify the lowest index, `low`, in which case the indices can range from `low` to `low + size - 1`.

```
public class BoundedIntArray
{
  private int[] myItems;        // storage for the list
  private int myLowIndex;       // lowest index

  public BoundedIntArray(int size)
  {
    myItems = new int[size];
    myLowIndex = 0;
  }

  public BoundedIntArray(int size, int low)
  {
    myItems = new int[size];
    myLowIndex = low;
  }

  // other methods not shown
}
```

30. Which of the following is the best reason for declaring the data fields `myItems` and `myLowIndex` to be private rather than public?

(A) This permits `BoundedIntArray` objects to be initialized and modified.

(B) This permits `BoundedIntArray` methods to be written and tested before code that uses a `BoundedIntArray` is written.

(C) This helps to prevent clients of the `BoundedIntArray` class from writing code that would need to be modified if the implementation of `BoundedIntArray` were changed.

(D) This prevents compile-time errors whenever public methods are called that access the private data fields.

(E) This prevents run-time errors whenever public methods are called that access the private data fields.

---

31. Consider the following statements.

```
BoundedIntArray arr1 = new BoundedIntArray(100, 5);
BoundedIntArray arr2 = new BoundedIntArray(100);
```

Which of the following best describes `arr1` and `arr2` after these statements?

(A) `arr1` and `arr2` both represent lists of integers indexed from `0` to `99`.

(B) `arr1` and `arr2` both represent lists of integers indexed from `5` to `104`.

(C) `arr1` represents a list of integers indexed from `0` to `104`, and `arr2` represents a list of integers indexed from `0` to `99`.

(D) `arr1` represents a list of integers indexed from `5` to `99`, and `arr2` represents a list of integers indexed from `0` to `99`.

(E) `arr1` represents a list of integers indexed from `5` to `104`, and `arr2` represents a list of integers indexed from `0` to `99`.

32. Which of the following is (are) true of an interface?

I. An interface can contain a constructor.

II. An interface can be instantiated.

III. All methods in an interface are abstract.

(A) I only

(B) II only

(C) III only

(D) I and II only

(E) I, II, and III

---

33. Consider the following three declarations.

I. `Integer obj1 = new Integer(7);`

II. `Comparable obj2 = new Integer(7);`

III. `Comparable obj3 = new Comparable(7);`

Which of these declarations is (are) legal?

(A) I only

(B) I and II only

(C) I and III only

(D) II and III only

(E) I, II, and III

**Questions 34-35 refer to the following declarations.**

```
public class Point
{
  private double myX;
  private double myY;

  // postcondition: this Point has coordinates (0,0)
  public Point()
  { /* implementation not shown */ }

  // postcondition: this Point has coordinates (x,y)
  public Point(double x, double y)
  { /* implementation not shown */ }

  // other methods not shown
}

public class Circle
{
  private Point myCenter;
  private double myRadius;

  // postcondition: this Circle has center at (0,0) and radius 0.0
  public Circle()
  { /* implementation not shown */ }

  // postcondition: this Circle has the given center and radius
  public Circle(Point center, double radius)
  { /* implementation not shown */ }

  // other methods not shown
}
```

34. In a client program which of the following correctly declares and initializes `Circle circ` with center at (29.5, 33.0) and radius 10.0 ?

(A)
```
Circle circ = new Circle(29.5, 33.0, 10.0);
```

(B)
```
Circle circ = new Circle((29.5, 33.0), 10.0);
```

(C)
```
Circle circ = new Circle(new Point(29.5, 33.0), 10.0);
```

(D)
```
Circle circ = new Circle();
circ.myCenter = new Point(29.5, 33.0);
circ.myRadius = 10.0;
```

(E)
```
Circle circ = new Circle();
circ.myCenter = new Point();
circ.myCenter.myX = 29.5;
circ.myCenter.myY = 33.0;
circ.myRadius = 10.0;
```

---

35. Which of the following would be the best specification for a `Circle` method `isInside` that determines whether a `Point` lies inside this `Circle`?

(A)
```
public boolean isInside()
```

(B)
```
public void isInside(boolean found)
```

(C)
```
public boolean isInside(Point p)
```

(D)
```
public void isInside(Point p, boolean found)
```

(E)
```
public boolean isInside(Point p, Point center, double radius)
```

36. Consider the following method.

```
public static void sort(String[] arr)
{
  for (int pass = arr.length - 1; pass >= 1; pass--)
  {
    String large = arr[0];
    int index = 0;

    for (int k = 0; k <= pass; k++)
    {
      if ((arr[k].compareTo(large)) > 0)
      {
        large = arr[k];
        index = k;
      }
    }

    arr[index] = arr[pass];
    arr[pass] = large;
  }
}
```

Assume `arr` is the following array.

| "Ann" | "Mike" | "Walt" | "Lisa" | "Shari" | "Jose" | "Mary" | "Bill" |
|---|---|---|---|---|---|---|---|

What is the intermediate value of `arr` after two iterations of the outer `for` loop in the call `sort(arr)` ?

(A)

| "Ann" | "Mike" | "Walt" | "Lisa" | "Shari" | "Jose" | "Mary" | "Bill" |
|---|---|---|---|---|---|---|---|

(B)

| "Ann" | "Mike" | "Lisa" | "Shari" | "Jose" | "Mary" | "Bill" | "Walt" |
|---|---|---|---|---|---|---|---|

(C)

| "Ann" | "Bill" | "Jose" | "Lisa" | "Mary" | "Mike" | "Shari" | "Walt" |
|---|---|---|---|---|---|---|---|

(D)

| "Ann" | "Mike" | "Bill" | "Lisa" | "Mary" | "Jose" | "Shari" | "Walt" |
|---|---|---|---|---|---|---|---|

(E)

| "Walt" | "Shari" | "Ann" | "Lisa" | "Mike" | "Jose" | "Mary" | "Bill" |
|---|---|---|---|---|---|---|---|

37. The following incomplete method is intended to return the largest integer in the array `numbers`.

```
// precondition:  numbers.length > 0
public static int findMax(int[] numbers)
{
  int posOfMax = 0;

  for (int index = 1; index < numbers.length; index++)
  {
    if ( /* condition */ )
    {
      /* statement */
    }
  }
  return numbers[posOfMax];
}
```

Which of the following can be used to replace `/* condition */` and `/* statement */` so that `findMax` will work as intended?

| | `/* condition */` | `/* statement */` |
|---|---|---|
| (A) | `numbers[index] > numbers[posOfMax]` | `posOfMax = numbers[index];` |
| (B) | `numbers[index] > numbers[posOfMax]` | `posOfMax = index;` |
| (C) | `numbers[index] > posOfMax` | `posOfMax = numbers[index];` |
| (D) | `numbers[index] < posOfMax` | `posOfMax = numbers[index];` |
| (E) | `numbers[index] < numbers[posOfMax]` | `posOfMax = index;` |

**Questions 38-39 refer to the following information.**

Consider the following data field and method. The method `removeDups` is intended to remove all adjacent duplicate numbers from `myData`, but does not work as intended.

```
private ArrayList myData;

public void removeDups()
{
  int k = 1;
  while (k < myData.size())
  {
    if (myData.get(k).equals(myData.get(k - 1)))
    {
      myData.remove(k);
    }
    k++;
  }
}
```

For example, if `myData` has the values `3 3 4 4 4 8 7 7 7`, after calling `removeDups`, `myData` should have the values `3 4 8 7`.

38. Assume that `myData` has the following values.

```
2 7 5 5 5 5 6 6 3 3 3
```

Which of the following represents `myData` after the incorrect `removeDups` is executed?

(A) `2 7 5 6 3`

(B) `2 7 5 6 3 3`

(C) `2 7 5 5 6 3 3`

(D) `2 7 5 5 5 6 3 3`

(E) `2 7 5 5 5 5 6 6 3 3`

---

39. Which of the following best describes how to fix the error so that `removeDups` works as intended?

(A) `k` should be initialized to 0 at the beginning of the method.

(B) The `while` condition should be `(k < myData.size() - 1)`.

(C) The `if` test should be `(myData.get(k).equals(myData.get(k + 1)))`.

(D) The body of the `if` statement should be: `myData.remove(k - 1);`

(E) There should be an `else` before the statement `k++;`

40. Consider the following output.

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

Which of the following code segments will produce the output shown above?

(A)
```
for (int j = 1; j <= 6; j++)
{
  for (int k = 1; k < j; k++)
    System.out.print(" " + k);
  System.out.println();
}
```

(B)
```
for (int j = 1; j <= 6; j++)
{
  for (int k = 1; k <= j; k++)
    System.out.print(" " + j);
  System.out.println();
}
```

(C)
```
for (int j = 1; j <= 6; j++)
{
  for (int k = 1; k <= j; k++)
    System.out.print(" " + k);
  System.out.println();
}
```

(D)
```
for (int j = 1; j < 6; j++)
{
  for (int k = 1; k <= j; k++)
    System.out.print(" " + k);
  System.out.println();
}
```

(E)
```
for (int j = 1; j < 6; j++)
{
  for (int k = 1; k < j; k++)
    System.out.print(" " + k);
  System.out.println();
}
```