unit7-mcq

1. Consider the following code segment.

```
ArrayList<String> items = new ArrayList<String>();

items.add("A");

items.add("B");

items.add("C");

items.add(0, "D");

items.remove(3);

items.add(0, "E");

System.out.println(items);
```

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

(A) [A, B, C, E]

(B) [A, B, D, E]

(C) [E, D, A, B]

(D) [E, D, A, C]

(E) [E, D, C, B]

2. Consider the following code segment.

```
ArrayList<Integer> numList = new ArrayList<Integer>();
numList.add(3);
numList.add(2);
numList.add(1);
numList.add(1, 0);
numList.set(0, 2);
System.out.print(numList);
```

What is printed by the code segment?

(A) `[1, 3, 0, 1]`

(B) `[2, 0, 2, 1]`

(C) `[2, 0, 2, 3]`

(D) `[2, 3, 2, 1]`

(E) `[3, 0, 0, 1]`

unit7-mcq

3. Consider the following code segment.

```
ArrayList<String> animals = new ArrayList<>();
animals.add("fox");
animals.add(0, "squirrel");
animals.add("deer");
animals.set(2, "groundhog");
animals.add(1, "mouse");
System.out.println(animals.get(2) + " and " + animals.get(3));
```

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

(A) `mouse and fox`

(B) `fox and groundhog`

(C) `groundhog and deer`

(D) `fox and deer`

(E) `squirrel and groundhog`

4. Consider the following code segment.

```
ArrayList<Integer> oldList = new ArrayList();
oldList.add(100);
oldList.add(200);
oldList.add(300);
oldList.add(400);
ArrayList<Integer> newList = new ArrayList();
newList.add(oldList.remove(1));
newList.add(oldList.get(2));
System.out.println(newList);
```

What, if anything, is printed as a result of executing the code segment?

(A) `[100, 300, 400]`

(B) `[200, 300]`

(C) `[200, 400]`

(D) Nothing is printed because the code segment does not compile.

(E) Nothing is printed because an `IndexOutOfBoundsException` will occur.

5. Consider the following code segment.

```
ArrayList<Double> conditionRating = new ArrayList<Double>();
conditionRating.add(9.84);
conditionRating.add(8.93);
conditionRating.add(7.65);
conditionRating.add(6.24);
conditionRating.remove(2);
conditionRating.set(2, 7.63);
System.out.println(conditionRating);
```

What is printed when this code segment is executed?

(A) `[9.84, 7.63, 6.24]`

(B) `[9.84, 7.63, 7.65, 6.24]`

(C) `[9.84, 8.93, 7.63]`

(D) `[9.84, 8.93, 7.63, 6.24]`

(E) `[9.84, 8.93, 7.65, 7.63]`

6. Consider the following code segment.

```
ArrayList<Integer> nums = new ArrayList<Integer>();

nums.add(new Integer(37));

nums.add(new Integer(3));

nums.add(new Integer(0));

nums.add(1, new Integer(2));

nums.set(0, new Integer(1));

nums.remove(2);

System.out.println(nums);
```

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

(A) [1, 2, 0]

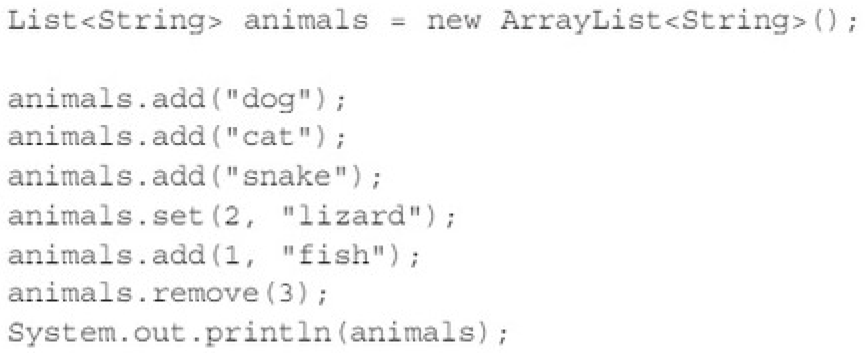
(B) [1, 3, 0]

(C) [1, 3, 2]

(D) [1, 37, 3, 0]

(E) [37, 0, 0]

7. Consider the following code segment.



```
List<String> animals = new ArrayList<String>();

animals.add("dog");
animals.add("cat");
animals.add("snake");
animals.set(2, "lizard");
animals.add(1, "fish");
animals.remove(3);
System.out.println(animals);
```

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

(A) [dog, fish, cat]

(B) [dog, fish, lizard]

(C) [dog, lizard, fish]

(D) [fish, dog, cat]

(E) The code throws an ArrayIndexOutOfBoundsException exception.

8. Consider the following code segment.

```
ArrayList<String> colors = new ArrayList<String>();

colors.add("Red");
colors.add("Orange");
colors.set(1, "Yellow");
colors.add(1, "Green");
colors.set(colors.size() - 1, "Blue");
colors.remove(0);
System.out.println(colors);
```

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

(A) `[Red, Orange]`

(B) `[Red, Green]`

(C) `[Yellow, Blue]`

(D) `[Green, Blue]`

(E) `[Blue, Yellow]`

unit7-mcq

9. Consider the following code segment.

```
ArrayList<String> numbers = new ArrayList<String>();
numbers.add("one");
numbers.add("two");
numbers.add(0, "three");
numbers.set(2, "four");
numbers.add("five");
numbers.remove(1);
```

Which of the following represents the contents of `numbers` after the code segment has been executed?

(A) `["one", "four", "five"]`

(B) `["three", "two", "five"]`

(C) `["three", "four", "two"]`

(D) `["three", "four", "five"]`

(E) `["one", "two", "three", "four", "five"]`

10. Consider the following statement, which is intended to create an `ArrayList` named `theater_club` to store elements of type `Student`. Assume that the `Student` class has been properly defined and includes a no-parameter constructor.

```
ArrayList<Student> theater_club = new /* missing code */;
```

Which choice can replace `/*` *missing code* `*/` so that the statement compiles without error?

(A) `Student()`

(B) `Student ArrayList()`

(C) `ArrayList(Student)`

(D) `ArrayList<Student>()`

(E) `ArrayList<theater_club>()`

11. Consider the following method `countNegatives`, which searches an `ArrayList` of `Integer` objects and returns the number of elements in the list that are less than `0`.

```
public static int countNegatives(ArrayList<Integer> arr)
{
     int count = 0;
     for (int j = 0; j < arr.size(); j++)   // Line 4
     {
         if (arr.get(j) < 0)
         {
            count++;
         }
     }
     return count;
}
```

Which of the following best explains the impact to the `countNegatives` method when, in line 4, `j < arr.size()` is replaced with `j <= arr.size() - 1` ?

(A) It has no impact on the behavior of the method.

(B) It causes the method to ignore the last element in `arr`.

(C) It causes the method to throw an `IndexOutOfBounds` exception.

(D) It reduces the size of `arr` by `1` and the last element will be removed.

(E) It changes the number of times the loop executes, but all indexes in `arr` will still be accessed.

12. Consider the following method `findValue`, which takes an `ArrayList` of `String` elements and a `String` value as parameters and returns `true` if the `String` value is found in the list and `false` otherwise.

```
public static boolean findValue(ArrayList<String> arr, String key)
{
     for (int j = 0; j < arr.size(); j++)     // Line 3
     {
         if (arr.get(j).equals(key))
         {
            return true;
         }
     }
     return false;
}
```

Which of the following best explains the impact to the `findValue` method when, in line 3, `int j = 0` is replaced by `int j = 1` ?

(A) It has no impact on the behavior of the method.

(B) It will cause the method to return a different result when the `key` value is not in the list.

(C) It will cause the method to return a different result when the `key` value is found only at the first index in the list.

(D) It will cause the method to return a different result when the `key` value is found only at the last index in the list.

(E) It will cause the method to throw an array index out of bounds exception.

13. Consider the following method, `inCommon`, which takes two `Integer ArrayList` parameters. The method returns `true` if the same integer value appears in both lists at least one time, and `false` otherwise.

```
public static boolean inCommon(ArrayList<Integer> a, ArrayList<Integer> b)
{
     for (int i = 0; i < a.size(); i++)
     {
         for (int j = 0; j < b.size(); j++)    // Line 5
         {
            if (a.get(i).equals(b.get(j)))
            {
        return true;
            }
         }
     }
     return false;
}
```

Which of the following best explains the impact to the `inCommon` method when line 5 is replaced by `for (int j = b.size() - 1; j > 0; j--)` ?

(A) The change has no impact on the behavior of the method.

(B) After the change, the method will never check the first element in list `b`.

(C) After the change, the method will never check the last element in list `b`.

(D) After the change, the method will never check the first and the last elements in list `b`.

(E) The change will cause the method to throw an `IndexOutOfBounds` exception.

14. 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.

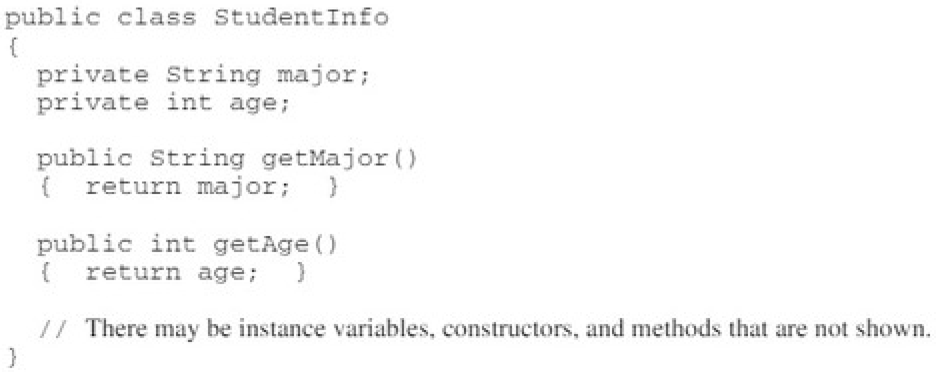
(B) An ArrayList always uses less memory than an array does.

(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.

15. Consider the following class declaration.



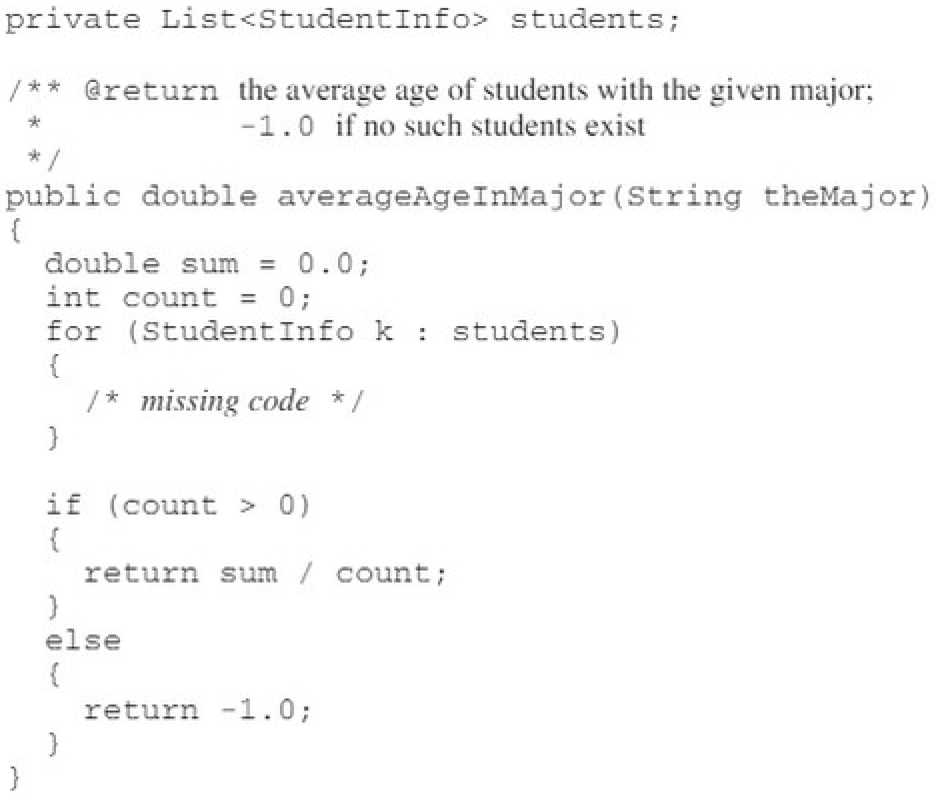
```
public class StudentInfo
{
  private String major;
  private int age;

  public String getMajor()
  {  return major;  }

  public int getAge()
  {  return age;  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

The following instance variable and method appear in another class.



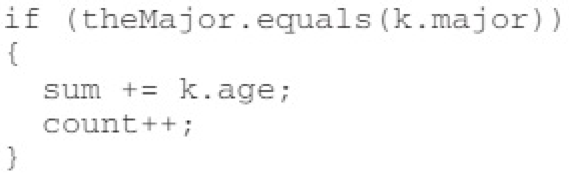
```
private List<StudentInfo> students;

/** @return  the average age of students with the given major;
 *           -1.0  if no such students exist
 */
public double averageAgeInMajor(String theMajor)
{
   double sum = 0.0;
   int count = 0;
   for (StudentInfo k : students)
   {
     /* missing code */
   }

   if (count > 0)
   {
     return sum / count;
   }
   else
   {
     return -1.0;
   }
}
```

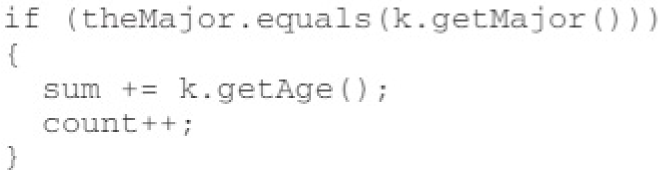
Which of the following could be used to replace /* *missing code* */ so that averageAgeInMajor will compile without error?

(A)



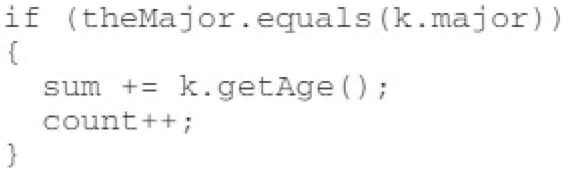
```
if (theMajor.equals(k.major))
{
  sum += k.age;
  count++;
}
```

(B)



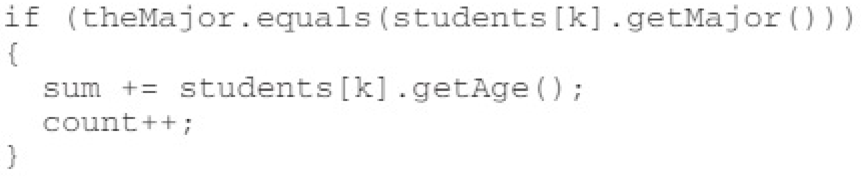
```
if (theMajor.equals(k.getMajor()))
{
  sum += k.getAge();
  count++;
}
```

(C)



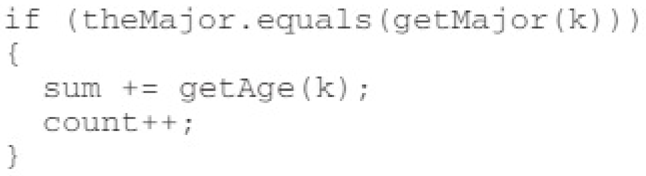
```
if (theMajor.equals(k.major))
{
  sum += k.getAge();
  count++;
}
```

(D)



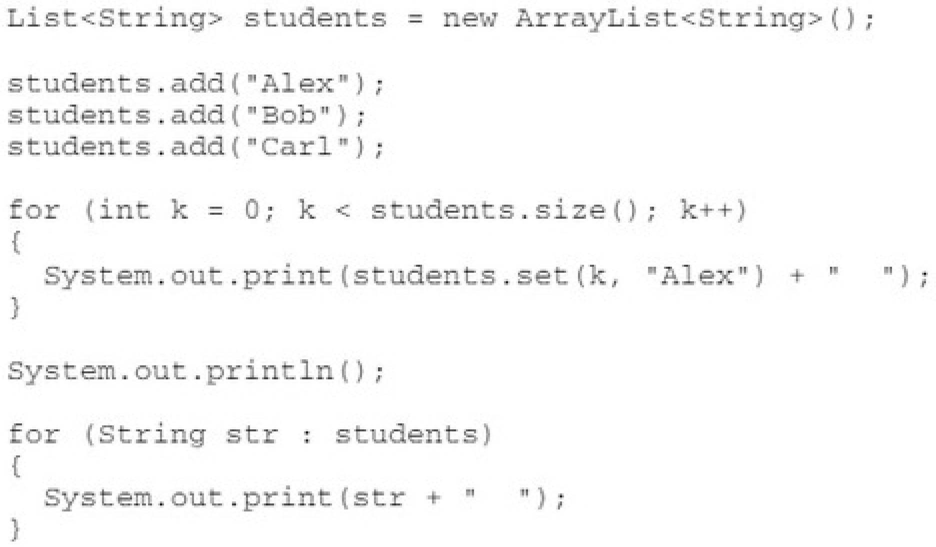
```
if (theMajor.equals(students[k].getMajor()))
{
  sum += students[k].getAge();
  count++;
}
```

(E)



```
if (theMajor.equals(getMajor(k)))
{
  sum += getAge(k);
  count++;
}
```

16. Consider the following code segment.



```
List<String> students = new ArrayList<String>();

students.add("Alex");
students.add("Bob");
students.add("Carl");

for (int k = 0; k < students.size(); k++)
{
  System.out.print(students.set(k, "Alex") + "  ");
}

System.out.println();

for (String str : students)
{
  System.out.print(str + "  ");
}
```

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

(A) Alex Alex Alex
Alex Alex Alex

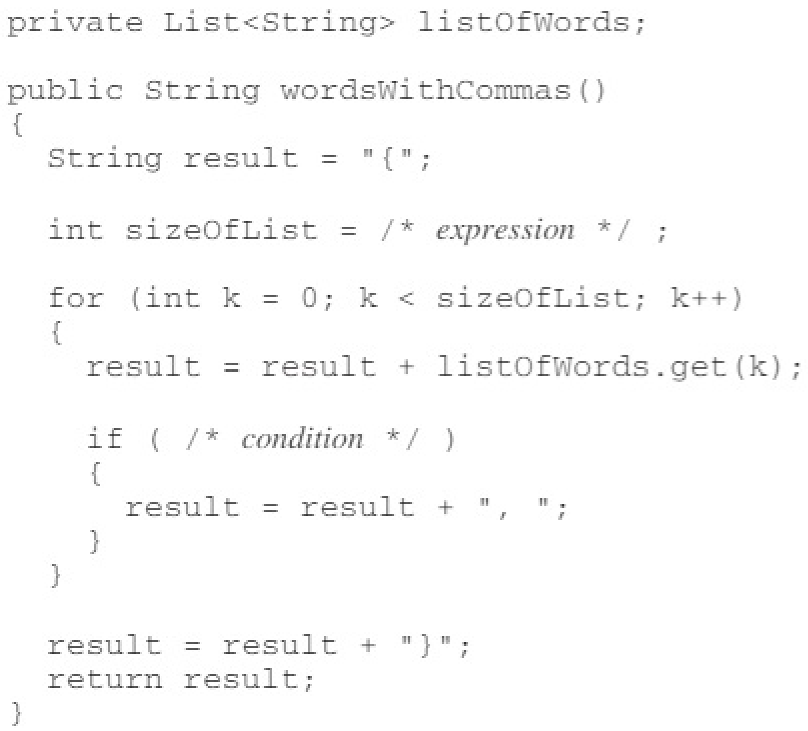
(B) Alex Alex Alex
Alex Bob Carl

(C) Alex Bob Carl
Alex Alex Alex

(D) Alex Bob Carl
Alex Bob Carl

(E) Nothing is printed because the first print statement will cause a runtime exception to be thrown.

17. Consider the following instance variable and method. Method wordsWithCommas is intended to return a string containing all the words in listOfWords separated by commas and enclosed in braces.For example, if listOfWords contains ["one", "two", "three"], the string returned by the call wordsWithCommas () should be "{one, two, three}".



```
private List<String> listOfWords;

public String wordsWithCommas()
{
  String result = "{";

  int sizeOfList = /* expression */ ;

  for (int k = 0; k < sizeOfList; k++)
  {
    result = result + listOfWords.get(k);

    if ( /* condition */ )
    {
      result = result + ", ";
    }
  }

  result = result + "}";
  return result;
}
```

Which of the following can be used to replace /* _expression_ */ and /* _condition_ */ so thatwordsWithCommas will work as intended?

(A) / * _expression_ * / / / * _condition_ * /
listOfWords.size() - 1 / k != 0

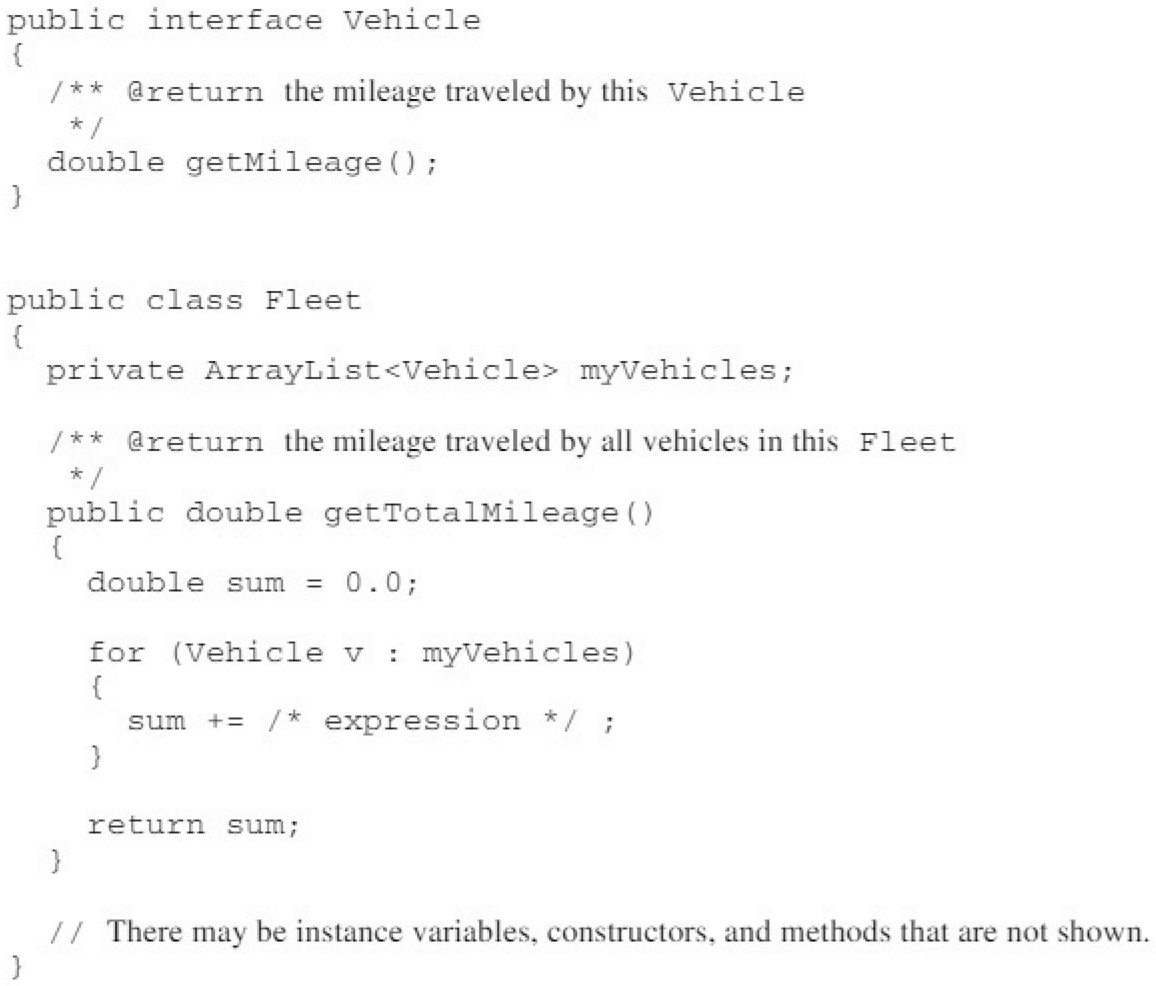
(B) / * _expression_ * / / / * _condition_ * /
listOfWords.size() / k != 0

(C) / * _expression_ * / / / * _condition_ * /
listOfWords.size() - 1 / k != sizeOfList - 1

(D) / * _expression_ * / / / * _condition_ * /
listOfWords.size() / k != sizeOfList - 1

(E) / * _expression_ * / / / * _condition_ * /
result.length() / k != 0

18. Consider the following interface and class declarations.



```
public interface Vehicle
{
  /** @return the mileage traveled by this Vehicle
   */
  double getMileage();
}


public class Fleet
{
  private ArrayList<Vehicle> myVehicles;

  /** @return the mileage traveled by all vehicles in this Fleet
   */
  public double getTotalMileage()
  {
    double sum = 0.0;

    for (Vehicle v : myVehicles)
    {
      sum += /* expression */ ;
    }

    return sum;
  }

  // There may be instance variables, constructors, and methods that are not shown.
}
```

Which of the following can be used to replace /* expression */ so that getTotalMileage returns the total of the miles traveled for all vehicles in the fleet?

(A) getMileage (v)

(B) myVehicles [v] .getMileage ()

(C) Vehicle.get (v) .getMileage ()

(D) myVehicles.get (v) .getMileage ()

(E) v.getMileage ()

unit7-mcq

19. Consider the following method.

```
/** Removes all occurrences of nameToRemove from nameList.

* @param nameList a list of names

* @param nameToRemove a name to be removed from nameList

*/

public void removeName(List<String> nameList, String nameToRemove)

{

/* missing implementation */

}
```

Which of the following can be used to replace /* *missing implementation* */ so that removeName will work as intended?

I.
```
for (String name : nameList)

{

      if (name.equals(nameToRemove))

            name.remove();

}
```

II.
```
for (int k = 0; k < nameList.size(); k++)

{

      if (nameList.get(k).equals(nameToRemove))

            nameList.remove(k);
```

```
        }
```

III.
```
for (int k = nameList.size() - 1; k >= 0; k--)

{

if (nameList.get(k).equals(nameToRemove))

    nameList.remove(k);

}
```

(A) I only

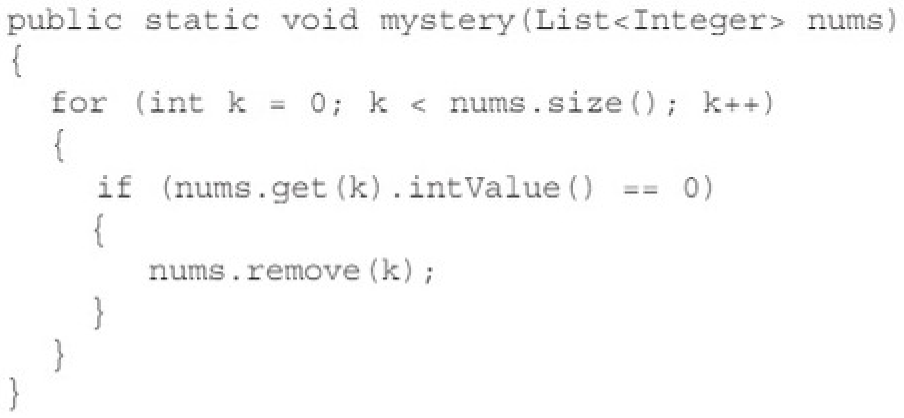
(B) II only

(C) III only

(D) II and III only

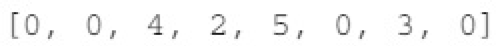
(E) I, II, and III

20. Consider the following method.



```
public static void mystery(List<Integer> nums)
{
  for (int k = 0; k < nums.size(); k++)
  {
    if (nums.get(k).intValue() == 0)
    {
      nums.remove(k);
    }
  }
}
```

Assume that a List<Integer> values initially contains the following Integer values.



```
[0, 0, 4, 2, 5, 0, 3, 0]
```

What will values contain as a result of executing mystery(values) ?

(A) [0, 0, 4, 2, 5, 0, 3, 0]

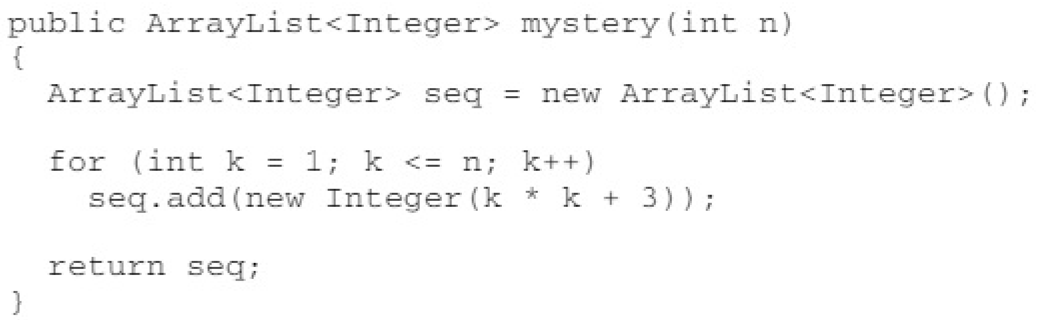
(B) [4, 2, 5, 3]

(C) [0, 0, 0, 0, 4, 2, 5, 3]

(D) [0, 4, 2, 5, 3]

(E) The code throws an ArrayIndexOutOfBoundsException exception.

21. Consider the following method.



```
public ArrayList<Integer> mystery(int n)
{
  ArrayList<Integer> seq = new ArrayList<Integer>();

  for (int k = 1; k <= n; k++)
    seq.add(new Integer(k * k + 3));

  return seq;
}
```

Which of the following is printed as a result of executing the following statement?
System.out.println(mystery (6)) ;

(A) [3, 4, 7, 12, 19, 28]

(B) [3, 4, 7, 12, 19, 28, 39]

(C) [4, 7, 12, 19, 28, 39]

(D) [39, 28, 19, 12, 7, 4]

(E) [39, 28, 19, 12, 7, 4, 3]

Consider the following correct implementation of the insertion sort algorithm. The `insertionSort` method correctly sorts the elements of `ArrayList data` into increasing order.

```
public static void insertionSort(ArrayList<Integer> data)
{
     for (int j = 1; j < data.size(); j++)
     {
         int v = data.get(j);
         int k = j;

         {
            data.set(k, data.get(k - 1)); /* Statement 1 */
            k--;
         }
         data.set(k, v); /* Statement 2 */
         /* End of outer loop */
     }
}
```

22. Assume that `insertionSort` has been called with an `ArrayList` parameter that has been initialized with the following `Integer` objects.

```
[5, 2, 4, 1, 3, 6]
```

What will the contents of `data` be after three passes of the outside loop (i.e., when `j == 3` at the point indicated by `/*` *End of outer loop* `*/`) ?

(A) `[1, 2, 3, 4, 5, 6]`

(B) `[1, 2, 3, 5, 4, 6]`

(C) `[1, 2, 4, 5, 3, 6]`

(D) `[2, 4, 5, 1, 3, 6]`

(E) `[5, 2, 1, 3, 4, 6]`

23. Assume that `insertionSort` is called with an `ArrayList` parameter that has been initialized with the following `Integer` objects.

```
[1, 2, 3, 4, 5, 6]
```

How many times will the statements indicated by `/*` *Statement 1* `*/` and `/*` *Statement 2* `*/` execute?

unit7-mcq

(A)

Statement 1	*Statement 2*
0	0

(B)

Statement 1	*Statement 2*
0	5

(C)

Statement 1	*Statement 2*
0	6

(D)

Statement 1	*Statement 2*
5	5

(E)

Statement 1	*Statement 2*
6	6

unit7-mcq

24. In the following code segment, assume that the `ArrayList wordList` has been initialized to contain the `String` values `["apple", "banana", "coconut", "lemon", "orange", "pear"]`.

```
int count = 0;
for (String word : wordList)
{
     if (word.indexOf("a") >= 0)
     {
         count++;
     }
}
System.out.println(count);
```

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

(A) `1`

(B) `2`

(C) `3`

(D) `4`

(E) `5`

25. Consider the following method, `remDups`, which is intended to remove duplicate consecutive elements from `nums`, an `ArrayList` of integers. For example, if `nums` contains `{1, 2, 2, 3, 4, 3, 5, 5, 6}`, then after executing `remDups(nums)`, `nums` should contain `{1, 2, 3, 4, 3, 5, 6}`.

```
public static void remDups(ArrayList<Integer> nums)
{
     for (int j = 0; j < nums.size() - 1; j++)
     {
         if (nums.get(j).equals(nums.get(j + 1)))
         {
            nums.remove(j);
            j++;
         }
     }
}
```

The code does not always work as intended. Which of the following lists can be passed to `remDups` to show that the method does NOT work as intended?

(A) `{1, 1, 2, 3, 3, 4, 5}`

(B) `{1, 2, 2, 3, 3, 4, 5}`

(C) `{1, 2, 2, 3, 4, 4, 5}`

(D) `{1, 2, 2, 3, 4, 5, 5}`

(E) `{1, 2, 3, 3, 4, 5, 5}`

unit7-mcq

26. The `removeElement` method is intended to remove all instances of `target` from the `ArrayList` object `data` passed as a parameter. The method does not work as intended for all inputs.

```
public void removeElement(ArrayList<Integer> data, int target)
{
     for (int j = 0; j < data.size(); j++)
     {
          if (data.get(j).equals(target))
          {
             data.remove(j);
          }
     }
}
```

Assume that the `ArrayList` object `scores` and the `int` variable `low_score` have been properly declared and initialized. In which of the following cases will the method call `removeElement(scores, low_score)` fail to produce the intended result?

(A) When `scores` is `[0, 2, 0, 2, 0, 6]` and `low_score` is `0`

(B) When `scores` is `[2, 4, 0, 5, 7, 0]` and `low_score` is `0`

(C) When `scores` is `[3, 4, 5, 7, 7, 2]` and `low_score` is `1`

(D) When `scores` is `[8, 8, 4, 3, 3, 6]` and `low_score` is `3`

(E) When `scores` is `[9, 9, 5, 9, 7, 7]` and `low_score` is `5`

27. In the following code segment, assume that the `ArrayList numList` has been properly declared and initialized to contain the `Integer` values `[1, 2, 2, 3]`. The code segment is intended to insert the `Integer` value `val` in `numList` so that `numList` will remain in ascending order. The code segment does not work as intended in all cases.

```
int index = 0;
while (val > numList.get(index))
{
     index++;
}
numList.add(index, val);
```

For which of the following values of `val` will the code segment not work as intended?

(A) `0`

(B) `1`

(C) `2`

(D) `3`

(E) `4`

unit7-mcq

28. Consider the following correct implementation of the selection sort algorithm.

```
public static void selectionSort(int[] elements)
{
     for (int j = 0; j < elements.length - 1; j++)
     {
         int minIndex = j;
         for (int k = j + 1; k < elements.length; k++)
         {
            if (elements[k] < elements[minIndex])
            {
        minIndex = k;
            }
         }
         if (j != minIndex)
         {
            int temp = elements[j];
            elements[j] = elements[minIndex];
            elements[minIndex] = temp;    // Line 19
         }
     }
}
```

The following declaration and method call appear in a method in the same class as `selectionSort`.

```
int[] arr = {9, 8, 7, 6, 5};
selectionSort(arr);
```

How many times is the statement `elements[minIndex] = temp;` in line 19 of the method executed as a result of the call to `selectionSort` ?

(A) 1

(B) 2

(C) 3

(D) 4

(E) 5

29. Consider the following statement, which is intended to create an `ArrayList` named `values` that can be used to store `Integer` elements.

```
/* missing code */ = new ArrayList<>();
```

Which of the following can be used to replace `/*` *missing code* `*/` so that the statement compiles without error?

I. `ArrayList values`
II. `ArrayList<int> values`
III. `ArrayList<Integer> values`

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

30. Consider the following statement, which is intended to create an `ArrayList` named `years` that can be used to store elements both of type `Integer` and of type `String`.

```
/* missing code */ = new ArrayList();
```

Which of the following can be used to replace `/*` *missing code* `*/` so that the statement compiles without error?

(A) `ArrayList years`
(B) `ArrayList years()`
(C) `ArrayList years[]`
(D) `ArrayList<Integer> years`
(E) `ArrayList<String> years`

31. 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]

32. Consider the following correct implementation of the insertion sort algorithm.

```
public static void insertionSort(int[] elements)
{
     for (int j = 1; j < elements.length; j++)
     {
         int temp = elements[j];
         int possibleIndex = j;

         {
            elements[possibleIndex] = elements[possibleIndex - 1];
            possibleIndex--;    // Line 10
         }
         elements[possibleIndex] = temp;
     }
}
```

The following declaration and method call appear in a method in the same class as `insertionSort`.

```
int[] arr = {4, 12, 4, 7, 19, 6};
insertionSort(arr);
```

How many times is the statement `possibleIndex--;` in line 10 of the method executed as a result of the call to `insertionSort` ?

(A) 2

(B) 3

(C) 4

(D) 5

(E) 6

33. Consider the following correct implementation of the selection sort algorithm.

```
public static void selectionSort(int[] elements)
{
     for (int j = 0; j < elements.length - 1; j++)
     {
         int minIndex = j;
         for (int k = j + 1; k < elements.length; k++)
         {
            if (elements[k] < elements[minIndex])
            {
        minIndex = k;    // Line 11
            }
         }
         if (j != minIndex)
         {
            int temp = elements[j];
            elements[j] = elements[minIndex];
            elements[minIndex] = temp;
         }
     }
}
```

The following declaration and method call appear in the same class as `selectionSort`.

```
int[] vals = {5, 10, 2, 1, 12};
selectionSort(vals);
```

How many times is the statement `minIndex = k;` in line 11 of the method executed as a result of the call to `selectionSort` ?

(A) 0

(B) 1

(C) 2

(D) 3

(E) 4

34. 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.

35. In the following code segment, assume that the `ArrayList data` has been initialized to contain the `Integer` values `[4, 3, 4, 5, 3, 4]`.

```
int j = 0;
while (j < data.size() - 1)
{
     if (data.get(j) > data.get(j + 1))
     {
         System.out.print(data.get(j + 1) + " ");
     }
     j++;
}
```

What, if anything, is printed as a result of executing the code segment?

(A) `3 3`

(B) `4 5`

(C) `4 5 4`

(D) Nothing is printed because the code segment does not compile.

(E) Nothing is printed because an `IndexOutOfBoundsException` occurs.

36. In the code segment below, assume that the `ArrayList` object `numbers` has been properly declared and initialized to contain `[0, 2, 4, 5]`.

```
for (int k = numbers.size() - 1; k >= 0; k--)
{
     if (numbers.get(k) > k)
     {
         System.out.print(k + " ");
     }
}
```

What, if anything, is printed as a result of executing the code segment?

(A) `1 2 3`

(B) `2 4 5`

(C) `3 2 1`

(D) `5 4 2`

(E) Nothing will be printed because an `IndexOutOfBoundsException` will occur.

unit7-mcq

37. Consider the following method, which is intended to return a list containing the elements of the parameter myList with all even elements removed.

```
public static ArrayList<Integer> removeEvens(ArrayList<Integer> myList)
{
     for (int i = 0; i < myList.size(); i++)
     {
         if (myList.get(i) % 2 == 0)
         {
            myList.remove(i);
         }
     }
     return myList;
}
```

Which of the following best explains why the code segment does not work as intended?

(A) The code segment causes an IndexOutOfBoundsException for all lists because of an incorrect Boolean expression in the for loop.

(B) The code segment causes an IndexOutOfBoundsException for lists with at least one even element because the indexes of all subsequent elements change by one when a list element is removed.

(C) The code segment returns a list with fewer elements than intended because it fails to consider the last element of myList.

(D) The code segment removes the wrong elements of myList because the condition in the if statement to test whether an element is even is incorrect.

(E) The code segment skips some elements of myList because the indexes of all subsequent elements change by one when a list element is removed.

38. The following method is intended to remove all elements of an `ArrayList` of integers that are divisible by `key` and add the removed elements to a new `ArrayList`, which the method returns.

```
public static ArrayList<Integer> match(ArrayList<Integer> numList, int
key)
{
     ArrayList<Integer> returnList = new ArrayList<Integer>();
     int i = 0;
     while (i < numList.size())
     {
         int num = numList.get(i);
         if (num % key == 0)
         {
            numList.remove(i);
            returnList.add(num);
         }
         i++;
     }
     return returnList;
}
```

As an example, if the method is called with an `ArrayList` containing the values `[5, 2, 10, 20, 16]` and the parameter `key` has the value `5`, then `numList` should contain `[2, 16]` at the end of the method and an `ArrayList` containing `[5, 10, 20]` should be returned.

Which of the following best explains why the method does not always work as intended?

(A) The method attempts to add an element to `returnList` after that element has already been removed from `numList`.

(B) The method causes a `NullPointerException` to be thrown when no matches are found.

(C) The method causes an `IndexOutOfBoundsException` to be thrown.

(D) The method fails to correctly determine whether an element of `numList` is divisible by `key`.

(E) The method skips some elements of `numList` during the traversal.

unit7-mcq

39. The following question 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.

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. The following question 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.

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

unit7-mcq

41. Consider the following class definition.

```
public class Value
{
     private int num;
     public int getNum()
     {
         return num;
     }
     // There may be instance variables, constructors, and methods not
  shown.
}
```

The following method appears in a class other than `Value`. It is intended to sum all the `num` instance variables of the `Value` objects in its `ArrayList` parameter.

```
/** Precondition: valueList is not null */
public static int getTotal(ArrayList<Value> valueList)
{
     int total = 0;
     /* missing code */
     return total;
}
```

Which of the following code segments can replace `/*` *missing code* `*/` so the `getTotal` method works as intended?

```
  I. for (int x = 0; x < valueList.size(); x++)
     {
        total += valueList.get(x).getNum();
     }
 II. for (Value v : valueList)
     {
        total += v.getNum();
     }
III. for (Value v : valueList)
     {
        total += getNum(v);
     }
```

(A) I only

(B) II only

(C) III only

(D) I and II

(E) I and III