COMPUTER SCIENCE A
SECTION I
Time—1 hour and 15 minutes
Number of questions—40
Percent of total score—50

Directions: Determine the answer to each of the following questions or incomplete statements, using the available space for any necessary scratch work. Then decide which is the best of the choices given and fill in the corresponding circle on the answer sheet. No credit will be given for anything written in the examination booklet. Do not spend too much time on any one problem.

Notes:
- Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
- Assume that declarations of variables and methods appear within the context of an enclosing class.
- Assume that method calls that are not prefixed with an object or class name and are not shown within a complete class definition appear within the context of an enclosing class.
- Unless otherwise noted in the question, assume that parameters in method calls are not null and that methods are called only when their preconditions are satisfied.

1. Consider the following incomplete method, which is intended to return the number of integers that evenly (the integer inputVal. Assume that inputVal is greater than 0.

```
public static int numDivisors(int inputVal)
{
   int count = 0;
   for (int k = 1; k <= inputVal; k++)
   {
      if ( /* condition */ )
      {
         count++;
      }
   }
   return count;
}
```

Which of the following can be used to replace /* condition */ so that numDivisors will work as intended?

(A) k % inputVal == 0

(B) inputVal / k == 0

(C) inputVal % k != 0

(D) k / inputVal > 0

(E) inputVal % k == 0

2.   Consider the following two classes.

```
public class A
{
  public void show()
  {
    System.out.print("A");
  }
}


public class B extends A
{
  public void show()
  {
    System.out.print("B");
  }
}
```

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

```
A obj = new B();
obj.show();
```

(A)   The code results in a runtime error

(B)   A

(C)   B

(D)   AB

(E)   BA

3. Consider the following instance variable and method.

```
private int[] arr;

/** Precondition: arr.length > 0
 *   @return the largest value in array arr
 */
public int findMax()
{
  int maxVal = 0;

  for (int val : arr)
  {
    if (val > maxVal)
    {
      maxVal = val;
    }
  }

  return maxVal;
}
```

Method `findMax` is intended to return the largest value in the array `arr`. Which of the following best describes the conditions under which the method `findMax` will <u>not</u> work as intended?

(A) The largest value in `arr` occurs only once and is in `arr[arr.length - 1]`.

(B) The largest value in `arr` is negative.

(C) The largest value in `arr` occurs only once and is in `arr[0]`.

(D) The largest value in `arr` occurs more than once.

(E) The largest value in `arr` is zero.

4.   Consider the following code segment.

```
for (int r = 3; r > 0; r--)
{
  int c;

  for (c = 1; c < r; c++)
  {
    System.out.print("-");
  }
  for (c = r ; c <= 3; c++)
  {
    System.out.print("*");
  }

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

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

(A)   * _ _

       * * _

       * * *

(B)   * * *

       _ * *

       _ _ *

(C)   _ _ *

       * * *

       _ _ *

(D)   * * *

       * * _

       * _ _

(E)   _ _ *

       _ * *

       * * *

5.   Assume that x and y are boolean variables and have been properly initialized.

    (x || y) && x

Which of the following always evaluates to the same value as the expression above?

(A) x && y

(B) y

(C) x

(D) x || y

(E) x && y

6. Consider the following code segment.

```
for (int outer = 1; outer <= 6; outer++)
{
  for (int inner = outer; inner <= 6; inner++)
  {
    if (inner % 2 == 0)
    {
      System.out.print(inner + "  ");
    }
  }
  System.out.println();
}
```

What will be printed as a result of executing the code segment?

(A) 2   4   6
     2   4   6
     2   4   6

(B) 2   4   6
     2   4   6
     4   6
     4   6
     6
     6

(C) 2   4   6
     2   4   6
     2   4   6
     2   4   6
     2   4   6
     2   4   6

(D) 2   4
     2   4
     4
     4

(E) 2   4   6
     4   6
     6

7. Consider the following method, which is intended to return `true` if at least one of the three strings `s1`, `s2`, or `s3` contains the substring "art". Otherwise, the method should return `false`.

```
public static boolean containsArt(String s1, String s2, String s3)
{

    String all = s1 + s2 + s3;

    return (all.indexOf("art") != -1);

}
```

Which of the following method calls demonstrates that the method does <u>not</u> work as intended?

(A) containsArt("start", "article", "Bart")

(B) containsArt("harm", "chortle", "crowbar")

(C) containsArt("darkroom", "cartoon", "articulate")

(D) containsArt("rattrap", "similar", "today")

(E) containsArt("matriculate", "carat", "arbitrary")

8. A pair of number cubes is used in a game of chance. Each number cube has six sides, numbered from 1 to 6, inclusive, and there is an equal probability for each of the numbers to appear on the top side (indicating the cube's value) when the number cube is rolled. The following incomplete statement appears in a program that computes the sum of the values produced by rolling two number cubes.

```
int sum = /* missing code */ ;
```

Which of the following replacements for `/* missing code */` would best simulate the value produced as a result of rolling two number cubes?

(A) 2 * (int) (Math.random() * 6)

(B) 2 * (int) (Math.random() * 7)

(C) (int) (Math.random() * 13)

(D) 2 + (int) (Math.random() * 6) + (int) (Math.random() * 6)

(E) (int) (Math.random() * 6) + (int) (Math.random() * 6)

9. Consider the following interface and class declarations.

```
public interface Student
{   /* implementation not shown */   }

public class Athlete
{   /* implementation not shown */   }

public class TennisPlayer extends Athlete implements Student
{   /* implementation not shown */   }
```

Assume that each class has a zero-parameter constructor. Which of the following is NOT a valid declaration?

(A) `Athlete c = new TennisPlayer();`

(B) `Student a = new TennisPlayer();`

(C) `TennisPlayer b = new TennisPlayer();`

(D) `Athlete e = new Athlete();`

(E) `Student d = new Athlete();`

10. Consider the following instance variable and method.

```
private int[] numbers;

public void mystery(int x)
{
   for (int k = 1; k < numbers.length; k = k + x)
   {
     numbers[k] = numbers[k - 1] + x;
   }
}
```

Assume that numbers has been initialized with the following values.

`{17, 34, 21, 42, 15, 69, 48, 25, 39}`

Which of the following represents the order of the values in numbers as a result of the call mystery(3) ?

(A) `{17, 20, 23, 26, 29, 32, 35, 38, 41}`

(B) `{17, 37, 21, 42, 18, 69, 48, 28, 39}`

(C) `{20, 23, 21, 42, 45, 69, 51, 54, 39}`

(D) `{20, 34, 21, 45, 15, 69, 51, 25, 39}`

(E) `{17, 20, 21, 42, 45, 69, 48, 51, 39}`

11. Consider the following method.

```
public static void showMe(int arg)
{
  if (arg < 10)
  {
    showMe(arg + 1);
  }
  else
  {
    System.out.print(arg + " ");
  }
}
```

What will be printed as a result of the call showMe(0) ?

(A) 0 1 2 3 4 5 6 7 8 9

(B) 0 1 2 3 4 5 6 7 8 9

(C) 0 1 2 3 4 5 6 7 8 9 10

(D) 11

(E) 10

12. Consider the following method.

```
/** Precondition: values has at least one row */
public static int calculate(int[][] values)
{
  int found = values[0][0];
  int result = 0;
  for (int[] row : values)
  {
    for (int y = 0; y < row.length; y++)
    {
      if (row[y] > found)
      {
        found = row[y];
        result = y;
      }
    }
  }
  return result;
}
```

Which of the following best describes what is returned by the calculate method?

(A) The smallest value in the two-dimensional array

(B) The row index of an element with the smallest value in the two-dimensional array

(C) The column index of an element with the largest value in the two-dimensional array

(D) The largest value in the two-dimensional array

(E) The row index of an element with the largest value in the two-dimensional array

13. Consider the following method.

```
//* Precondition: num > 0 */
public static int doWhat(int num)
{
  int var = 0;

  for (int loop = 1; loop <= num; loop = loop + 2)
  {
    var += loop;
  }

  return var;
}
```

Which of the following best describes the value returned from a call to `doWhat` ?

(A) The sum of all integers between 1 and num, inclusive

(B) The sum of all even integers between 1 and num, inclusive

(C) The sum of all odd integers between 1 and num, inclusive

(D) No value is returned because of an infinite loop.

(E) num

14. Consider the following method.

```
//* Precondition: num > 0 */
public static int doWhat(int num)
{
  int var = 0;

  for (int loop = 1; loop <= num; loop = loop + 2)
  {
    var += loop;
  }

  return var;
}
```

Which of the following best describes the value returned from a call to `doWhat` ?

(A) The sum of all integers between 1 and num, inclusive

(B) The sum of all even integers between 1 and num, inclusive

(C) The sum of all odd integers between 1 and num, inclusive

(D) No value is returned because of an infinite loop.

(E) num

15. Consider the following code segment.

```
int x = 1;
while ( /* condition */ )
{
  if (x % 2 == 0)
  {
    System.out.print(x + " ");
  }
  x = x + 2;
}
```

The following conditions have been proposed to replace /* condition */ in the code segment.

I.   x < 0
II. x <= 1
III. x < 10

For which of the conditions will nothing be printed?

(A) I, II, and III

(B) I and II only

(C) I and III only

(D) II only

(E) I only

16. Consider the following code segment.

```
int x = 1;
while ( /* condition */ )
{
  if (x % 2 == 0)
  {
    System.out.print(x + " ");
  }
  x = x + 2;
}
```

The following conditions have been proposed to replace /* condition */ in the code segment.

I.   x < 0

II. x <= 1

III. x < 10

For which of the conditions will nothing be printed?

(A) I, II, and III

(B) I and II only

(C) I and III only

(D) II only

(E) I only

17. Consider the following method.

```java
/** Precondition: arr.length > 0 */
public static int mystery(int[] arr)
{
   int index = 0;
   int count = 0;
   int m = -1;

   for (int outer = 0; outer < arr.length; outer++)
   {
      count = 0;
      for (int inner = outer + 1; inner < arr.length; inner++)
      {
         if (arr[outer] == arr[inner])
         {
            count++;
         }
      }

      if (count > m)
      {
         index = outer;
         m = count;
      }
   }

   return index;
}
```

Assume that nums has been declared and initialized as an array of integer values. Which of the following best describes the value returned by the call mystery(nums) ?

(A) An index of the maximum value that occurs in nums

(B) A value that occurs most often in nums

(C) An index of a value that occurs most often in nums

(D) The number of times that the maximum value occurs in nums

(E) The maximum value that occurs in nums

18.   Consider the following method.

```java
/** Precondition: arr.length > 0 */
public static int mystery(int[] arr)
{
   int index = 0;
   int count = 0;
   int m = -1;

   for (int outer = 0; outer < arr.length; outer++)
   {
      count = 0;
      for (int inner = outer + 1; inner < arr.length; inner++)
      {
         if (arr[outer] == arr[inner])
         {
            count++;
         }
      }

      if (count > m)
      {
         index = outer;
         m = count;
      }
   }

   return index;
}
```

Assume that nums has been declared and initialized as an array of integer values. Which of the following best describes the value returned by the call mystery(nums) ?

(A)   An index of the maximum value that occurs in nums

(B)   A value that occurs most often in nums

(C)   An index of a value that occurs most often in nums

(D)   The number of times that the maximum value occurs in nums

(E)   The maximum value that occurs in nums

19. Consider the following recursive method.

```
public static void whatsItDo(String str)
{
  int len = str.length();
  if (len > 1)
  {
    String temp = str.substring(0, len - 1);
    System.out.println(temp);
    whatsItDo(temp);
  }
}
```

What is printed as a result of the call `whatsItDo("WATCH")` ?

(A)   ATCH
      ATC
      AT
      A

(B)   WATC
      WAT
      WA
      W

(C)   WATCH
      WATC
      WAT
      WA

(D)   WATC

(E)   H

20. Consider the following definition.

```
int[][] numbers = {{1, 2, 3},
                   {4, 5, 6}};
```

Which of the following code segments produces the output 123456 ?

(A)
```
for (int[] row : numbers)
{
  for (int n : row)
  {
    System.out.print(row[n]);
  }
}
```

(B)
```
for (int rc = 0; rc < numbers.length; rc++)
{
  System.out.print(numbers[rc]);
}
```

(C)
```
for (int[] row : numbers)
{
  for (int n : row)
  {
    System.out.print(n);
  }
}
```

(D)
```
for (int c = 0; c < numbers[0].length; c++)
{
  for (int r = 0; r < numbers.length; r++)
  {
    System.out.print(numbers[r][c]);
  }
}
```

(E)
```
for (int r = 0; r < numbers[0].length; r++)
{
  for (int c = 0; c < numbers.length; c++)
  {
    System.out.print(numbers[r][c]);
  }
}
```

21. Consider the following class.

```
public class SomeMethods
{
  public void one(int first)
  {  /* implementation not shown */  }

  public void one(int first, int second)
  {  /* implementation not shown */  }

  public void one(int first, String second)
  {  /* implementation not shown */  }
}
```

Which of the following methods can be added to the SomeMethods class without causing a compile-time error?

I. public void one(int value)
    { /* implementation not shown */ }

II. public void one(String first, int second)
     { /* implementation not shown */ }

III. public void one(int first, int second, int third)
      { /* implementation not shown */ }

(A) I, II, and III

(B) II and III

(C) I and II only

(D) I and III only

(E) I only

22. Consider the following two methods, which appear within a single class.

```java
public static void changeIt(int[] arr, int val, String word)
{
  arr = new int[5];
  val = 0;
  word = word.substring(0, 5);

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


public static void start()
{
  int[] nums = {1, 2, 3, 4, 5};
  int value = 6;
  String name = "blackboard";

  changeIt(nums, value, name);

  for (int k = 0; k < nums.length; k++)
  {
    System.out.print(nums[k] + " ");
  }

  System.out.print(value + " ");
  System.out.print(name);
}
```

What is printed as a result of the call `start()` ?

(A) 0 0 0 0 0 6 blackboard

(B) 1 2 3 4 5 6 black

(C) 0 0 0 0 0 0 black

(D) 1 2 3 4 5 6 blackboard

(E) 1 2 3 4 5 0 black

23. Consider the following recursive method.

```
/** Precondition: num ≥ 0 */
public static int what(int num)
{
   if (num < 10)
   {
      return 1;
   }
   else
   {
      return 1 + what(num / 10);
   }
}
```

Assume that `int val` has been declared and initialized with a value that satisfies the precondition of the method. Which of the following best describes the value returned by the call `what(val)` ?

(A) The sum of the digits in the decimal representation of `val` is returned.

(B) Nothing is returned. A run-time error occurs because of infinite recursion.

(C) The value `val/10` is returned.

(D) The number of digits in the decimal representation of `val` is returned.

(E) The value 1 is returned.

24. The price per box of ink pens advertised in an office supply catalog is based on the number of boxes ordered. The following table shows the pricing.

Number of Boxes	Price per Box
1 up to but not including 5	$5.00
5 up to but not including 10	$3.00
10 or more	$1.50

The following incomplete method is intended to return the total cost of an order based on the value of the parameter numBoxes.

```
/** Precondition: numBoxes > 0 */
public static double getCost(int numBoxes)
{
   double totalCost = 0.0;

   /* missing code */

   return totalCost;
}
```

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

```
I.  if (numBoxes >= 10)
    {
       totalCost = numBoxes * 1.50;
    }
    if (numBoxes >= 5)
    {
       totalCost = numBoxes * 3.00;
    }
    if (numBoxes > 0)
    {
       totalCost = numBoxes * 5.00;
    }


II. if (numBoxes >= 10)
    {
       totalCost = numBoxes * 1.50;
    }
    else if (numBoxes >= 5)
    {
       totalCost = numBoxes * 3.00;
    }
    else
    {
       totalCost = numBoxes * 5.00;
    }


III. if (numBoxes > 0)
     {
        totalCost = numBoxes * 5.00;
     }
     else if (numBoxes >= 5)
     {
        totalCost = numBoxes * 3.00;
     }
     else if (numBoxes >= 10)
     {
        totalCost - numBoxes * 1.50;
     }
```

(A) II and III

(B) I and II

(C) I only

(D) II only

(E) III only

25. Consider the following code segment.

```
String[][] board = new String[5][5];

for (int row = 0; row < 5; row++)
{
  for (int col = 0; col < 5; col++)
  {
    board[row][col] = "O";
  }
}

for (int val = 0; val < 5; val++)
{
  if (val % 2 == 1)
  {
    int row = val;
    int col = 0;
    while (col < 5 && row >= 0)
    {
      board[row][col] = "X";
      col++;
      row--;
    }
  }
}
```

Which of the following represents board after this code segment is executed?

(A)

	0	1	2	3	4
0	O	X	O	X	O
1	X	O	X	O	X
2	O	X	O	X	O
3	X	O	X	O	X
4	O	X	O	X	O

(B)

	0	1	2	3	4
0	X	O	O	O	X
1	O	X	O	X	O
2	O	O	X	O	O
3	O	X	O	X	O
4	X	O	O	O	X

(C)

	0	1	2	3	4
0	X	O	X	O	X
1	O	X	O	X	O
2	X	O	X	O	X
3	O	X	O	X	O
4	X	O	X	O	X

(D)

	0	1	2	3	4
0	O	X	O	X	O
1	X	O	X	O	O
2	O	X	O	O	O
3	X	O	O	O	O
4	O	O	O	O	O

(E)

	0	1	2	3	4
0	O	X	O	O	O
1	O	O	X	O	O
2	X	O	O	X	O
3	O	X	O	O	X
4	O	O	X	O	O

26. Consider the problem of finding the maximum value in an array of integers. The following code segments are proposed solutions to the problem. Assume that the variable arr has been defined as an array of int values and has been initialized with one or more values.

I.
```
int max = Integer.MIN_VALUE;
for (int value : arr)
{
  if (max < value)
  {
    max = value;
  }
}
```

II.
```
int max = 0;
boolean first = true;
for (int value : arr)
{
  if (first)
  {
    max = value;
    first = false;
  }
  else if (max < value)
  {
    max = value;
  }
}
```

III.
```
int max = arr[0];
for (int k = 1; k < arr.length; k++)
{
  if (max < arr[k])
  {
    max = arr[k];
  }
}
```

Which of the code segments will always correctly assign the maximum element of the array to the variable max ?

(A) I, II, and III

(B) II and III only

(C) III only

(D) II only

(E) I only

27. Consider the problem of finding the maximum value in an array of integers. The following code segments are proposed solutions to the problem. Assume that the variable arr has been defined as an array of int values and has been initialized with one or more values.

I.
```
int max = Integer.MIN_VALUE;
for (int value : arr)
{
   if (max < value)
   {
      max = value;
   }
}
```

II.
```
int max = 0;
boolean first = true;
for (int value : arr)
{
   if (first)
   {
      max = value;
      first = false;
   }
   else if (max < value)
   {
      max = value;
   }
}
```

III.
```
int max = arr[0];
for (int k = 1; k < arr.length; k++)
{
   if (max < arr[k])
   {
      max = arr[k];
   }
}
```

Which of the code segments will always correctly assign the maximum element of the array to the variable max ?

(A)   I, II, and III

(B)   II and III only

(C)   III only

(D)   II only

(E)   I only

28. 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 that `wordsWithCommas` will work as intended?

	/* *expression* */	/* *condition* */
(A)	`listOfWords.size() - 1`	`k != sizeOfList - 1`
(B)	`listOfWords.size() - 1`	`k != 0`
(C)	`listOfWords.size()`	`k != 0`
(D)	`result.length()`	`k != 0`
(E)	`listOfWords.size()`	`k != sizeOfList - 1`

29. Consider the following `binarySearch` method. The method correctly performs a binary search.

```
/** Precondition: data is sorted in increasing order. */
public static int binarySearch(int[] data, int target)
{
    int start = 0;
    int end = data.length - 1;
    while (start <= end)
    {
        int mid = (start + end) / 2;       /* Calculate midpoint */
        if (target < data[mid])
        {
            end = mid - 1;
        }
        else if (target > data[mid])
        {
            start = mid + 1;
        }
        else
        {
            return mid;
        }
    }
    return -1;
}
```

Consider the following code segment.

```
int[] values = {1, 2, 3, 4, 5, 8, 8, 8};
int target = 8;
```

What value is returned by the call `binarySearch(values, target)` ?

(A) 5

(B) -1

(C) 3

(D) 8

(E) 6

30. Consider the following incomplete method that is intended to return a string formed by concatenating elements from the parameter `words`. The elements to be concatenated start with `startIndex` and continue through the last element of `words` and should appear in reverse order in the resulting string.

```
/** Precondition: words.length > 0;
  *               startIndex >= 0
  */
public static String concatWords(String[] words, int startIndex)
{
    String result = "";

    /* missing code */

    return result;
}
```

For example, the following code segment uses a call to the `concatWords` method.

```
String[] things = {"Bear", "Apple", "Gorilla", "House", "Car"};
System.out.println(concatWords(things, 2));
```

When the code segment is executed, the string `"CarHouseGorilla"` is printed.

The following three code segments have been proposed as replacements for /* missing code */.

I.
```
for (int k = startIndex; k < words.length; k++)
{
    result += words[k] + words[words.length - k - 1];
}
```

II.
```
int k = words.length - 1;
while (k >= startIndex)
{
    result += words[k];
    k--;
}
```

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

for (int k = 0; k < temp.length - startIndex; k++)
{
    result += temp[k];
}
```

Which of these code segments can be used to replace /* missing code */ so that `concatWords` will work as intended?

(A) II and III

(B) I and II

(C) III

(D) II

(E) I