Practice Exam #3

SECTION I

Time — 1 hour and 15 minutes
Number of questions — 40
Percent of total grade — 50

1. Given the declarations

```
int p = 5, q = 3;
```

which of the following expressions evaluate to 7.5?

I. `(double)p * (double)q / 2;`
II. `(double)p * (double)(q / 2);`
III. `(double)(p * q / 2);`

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

2. Consider the following method.

```
public void mystery(int a, int b)
{
  System.out.print(a + " ");
  if (a <= b)
    mystery(a + 5, b - 1);
}
```

What is the output when `mystery(0, 16)` is called?

(A) `0`
(B) `0 5`
(C) `0 5 10`
(D) `0 5 10 15`
(E) `0 5 10 15 20`

3. Consider the following method `fun2`.

```
public int fun2(int x, int y)
{
  y -= x;
  return y;
}
```

What are the values of the variables `a` and `b` after the following code is executed?

```
int a = 3, b = 7;
b = fun2(a, b);
a = fun2(b, a);
```

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

4. Assuming that `a` and `b` are `boolean` variables, when is the following expression true?

```
!(!a || b) || (!a && b)
```

(A) If and only if `a` and `b` have different values
(B) If and only if `a` and `b` have the same value
(C) If and only if both `a` and `b` are true
(D) If and only if both `a` and `b` are false
(E) Never

5. 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 cast objects of type *X* into *Y* and vice-versa
(B) Being able to use some common code accessible in classes *X* and *Y* without duplication
(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 in the future to implement another class that reuses some code from *A*

6. The method

```
private void transpose(int[][] m)
{
  < implementation not shown >
}
```

flips the elements of `m` symmetrically over the diagonal. For example:

```
1 2 3                    1 4 7
4 5 6   transpose  ⟶     2 5 8
7 8 9                    3 6 9
```

Which of the following implementations of `transpose` will work as specified?

I.
```
for (int r = 0; r < m.length; r++)
{
  for (int c = 0; c < m[0].length; c++)
  {
    int temp = m[r][c];
    m[r][c] = m[c][r];
    m[c][r] = temp;
  }
}
```

II.
```
for (int c = m[0].length - 1; c > 0; c--)
{
  for (int r = c-1; r >= 0; r--)
  {
    int temp = m[r][c];
    m[r][c] = m[c][r];
    m[c][r] = temp;
  }
}
```

III.
```
for (int c = 0; c < m[0].length - 1; c++)
{
  for (int r = c + 1; r < m.length; r++)
  {
    int temp = m[r][c];
    m[r][c] = m[c][r];
    m[c][r] = temp;
  }
}
```

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

7. What is the value of `v[4]` after the following code is executed?

```
int d = 1;
int[] v = {1, 1, 1, 1, 1};

for (int i = 0; i < v.length; i++)
{
  d *= 2;
  v[i] += d;
}
```

(A) 16
(B) 32
(C) 33
(D) 64
(E) 65

8. Suppose we have the following interface `Game`.

```
public interface Game
{
  void playWith(Fun other);
}
```

We have found a compiled Java class, `Fun.class`. We do not have its source code, but we have discovered that a statement

```
Fun fun = new Fun(100);
```

compiles with no errors. Which of the following statements, if it compiles correctly, will convince us that `Fun` implements `Game`?

I. `Game game = fun;`
II. `System.out.print(fun.playWith(new Fun(99)));`
III. `System.out.print(fun.playWith(fun));`

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

9. What is printed when the following code segment is executed?

```
List<Integer> lst = new ArrayList<Integer>();
int k = 2;
while (lst.size() < 5)
{
  boolean found = false;
  for (Integer n : lst)
    if (k % n.intValue() == 0)
      found = true;
  if (!found)
    lst.add(new Integer(k));
  k++;
}
System.out.println(lst);
```

(A) `[2, 3, 4, 5, 6]`
(B) `[2, 3, 5, 7, 11]`
(C) `[2, 3, 4, 5, 6, 7]`
(D) `[2, 3, 5, 7, 11, 13]`
(E) Nothing is printed — the program goes into an infinite loop.

10. What is the result from the following code segment?

```
List<String> xyz = new ArrayList<String>();
xyz.add("X");
xyz.add("Y");
xyz.add("Z");

int count = 0;
for (String s1 : xyz)
{
  for (String s2 : xyz)
  {
    if (s1.equals(s2))
    {
      count++;
    }
  }
}

System.out.print(count);
```

(A) Syntax error
(B) 0 is displayed
(C) 1 is displayed
(D) 3 is displayed
(E) `NullPointerException`

11. What are the smallest and the largest possible values of `x` after the following statement has been executed?

```
int x = (int)(Math.sqrt(4*Math.random()) + 0.5);
```

(A) 0 and 1
(B) 0 and 2
(C) 0 and 3
(D) 1 and 2
(E) 1 and 3

12. A class `Point` has a `void` method `move(int dx, int dy)` that changes the coordinates of the point.

A class `Polygon` has a list of points, its vertices:

```
private ArrayList<Point> vertices;
```

`Polygon`'s method `move` is supposed to move all its vertices. Alex programmed it with a "for each" loop —

```
public void move(int dx, int dy)
{
  for (Point p : vertices)
    p.move(dx, dy);
}
```

— but Pat decided to replace it with a regular `for` loop:

```
public void move(int dx, int dy)
{
  for (int i = 0; i < vertices.size(); i++)
    vertices.get(i).move(dx, dy);
}
```

What is a good reason for this change?

(A) No good reason; Pat just wasn't sure how "for each" loops work.
(B) The `for` loop with `get(i)` is more conventional programming style.
(C) The "for each" loop didn't work because it cannot change objects in a list.
(D) The `for` loop with `get(i)` is more efficient.
(E) The "for each" loop won't work if the list `vertices` holds objects that belong to a subclass of `Point`.

13. Suppose class `D` extends class `B` and has one constructor. In which of the following situations the first statement in `D`'s constructor must be `super(...)`?

(A) `B` has private fields
(B) `B` has no constructors
(C) `B` has only one constructor, which takes parameters
(D) `B` has only one constructor, which takes no parameters
(E) `D`'s constructor takes parameters

14. Which of the following statements about Java's platform independence are true?

I. The value of the `MAX_VALUE` constant in the `java.lang.Integer` class is the same on any computer.

II. Java source code is compiled into bytecode, which then may be run on any computer that has a Java Virtual Machine installed.

III. Overflow in arithmetic operations occurs at the same values regardless of the platform on which the Java program is running.

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

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

```
String s = "ban";
ArrayList<String> words = new ArrayList<String>();
words.add(s);
words.add(s.substring(1));
words.add(s.substring(1, 2));
String total = "";
for (String w : words)
{
  total += w;
}
System.out.print(total.indexOf("an"));
```

(A) `1`
(B) `2`
(C) `3`
(D) `ana`
(E) `banana`

Questions 16-17 refer to the method `smile` below.

```
public static void smile(int n)
{
  if (n == 0)
    return;
  for (int k = 1; k <= n; k++)
  {
    System.out.print("smile!");
  }
  smile(n-1);
}
```

16. What is the output when `smile(4)` is called?

(A) `smile!`
(B) `smile!smile!`
(C) `smile!smile!smile!`
(D) `smile!smile!smile!smile!`
(E) `smile!smile!smile!smile!smile!smile!smile!smile!smile!smile!`

17. When `smile(4)` is called, how many times will `smile` actually be called, including the initial call?

(A) 2
(B) 3
(C) 4
(D) 5
(E) 10

18. Consider the following code segment, intended to find the position of an integer `targetValue` in `int[] a`.

```
int i = 0;
while (a[i] != targetValue)
{
  i++;
}
int position = i;
```

When will this code work as intended?

(A) Always
(B) Only when `targetValue == a[0]`
(C) Only when `0 <= targetValue < a.length`
(D) Only when `targetValue` equals `a[i]` for some `i`, such that `0 <= i < a.length`
(E) Only when `targetValue` is not equal to `a[i]` for any `i`, such that `0 <= i < a.length`

19. Given two initialized `String` variables, `str1` and `str2`, which of the following conditions correctly tests whether the value of `str1` is greater than or equal to the value of `str2` (in lexicographical order)?

(A) `str1 >= str2`
(B) `str1.compareTo(str2) >= 0`
(C) `str1.compareTo(str2) == true`
(D) `str1.length() > str2.length() || str1 >= str2`
(E) `str1.equals(str2) || str1.compareTo(str2) == 1`

20. Consider the following method from `ClassX`.

```
private int modXY(int x, int y)
{
  r = x / y;
  return x % y;
}
```

If `ClassX` compiles with no errors, which of the following must be true?

I. `r` must have the type `double`.
II. `r` is not a local variable in the `modXY` method.
III. `r` must be a static variable in `ClassX`.

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

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

```
int[] a = {0, 1};
int[] b = a;
a[0] = 1;
b[0] = 2;
System.out.println(a[0] + b[0] + a[1] + b[1]);
```

(A) 3
(B) 4
(C) 5
(D) 6
(E) None of the above

22. What is the result when the following code segment is compiled and executed?

```
Object[] nums = {null, null, null};
nums[0] = new Integer(3);
nums[1] = new Double(0.14);
System.out.println(nums[0].toString() +
        nums[1].toString() + nums[2].toString());
```

(A) `30.14null` is displayed
(B) `NullPointerException`
(C) `IllegalArgumentException`
(D) `IndexOutOfBoundsException`
(E) `ArrayIndexOutOfBoundsException`

23. Consider the following three code segments.

I.
```
int  i = 1;
while (i <= 10)
{
  System.out.print(i);
  i += 2;
}
```

II.
```
for (int i = 0; i < 5; i++)
{
  System.out.print(2*i + 1);
}
```

III.
```
for (int i = 0; i < 10; i++)
{
  i++;
  System.out.print(i);
}
```

Which of the three segments produce the same output?

(A) I and II only
(B) II and III only
(C) I and III only
(D) All three outputs are different.
(E) All three outputs are the same.

24. Consider the following method with missing code.

```
public void zeroSomething(int[][] m)
{
  int numRows = m.length;
  int numCols = m[0].length;
  < missing code >
}
```

Which of the following three versions of < *missing code* > are equivalent, that is, result in the same values for a given two-dimensional array `m`?

I.
```
for (int k = 0; k < numRows; k++)
{
  m[k][0] = 0;
  m[k][numCols - 1] = 0;
}
for (int k = 0; k < numCols; k++)
{
  m[0][k] = 0;
  m[numRows - 1][k] = 0;
}
```

II.
```
for (int k = 0; k < numRows - 1; k++)
{
  m[k][0] = 0;
  m[numRows - k - 1][numCols - 1] = 0;
}
for (int k = 0; k < numCols - 1; k++)
{
  m[0][numCols - k - 1] = 0;
  m[numRows - 1][k] = 0;
}
```

III.
```
for (int k = 0; k < numCols; k++)
{
  m[0][k] = 0;
  m[numRows - 1][k] = 0;
}
for (int[] r : m)
{
  r[0] = 0;
  r[numCols - 1] = 0;
}
```

(A) All three are equivalent
(B) I and II only
(C) II and III only
(D) I and III only
(E) All three are different

25. Suppose a, b, and c are positive integers under 1000 and x satisfies the formula

$$\frac{a}{b} = \frac{c}{x}$$

The integer value d is obtained by truncating x to an integer. Which of the following code segments correctly calculates d?

I.
```
d = c * b / a;
```

II.
```
int temp = c * b;
d = temp / a;
```

III.
```
int temp = b / a;
d = c * temp;
```

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

26. Consider the following method with two missing statements.

```
/** Returns the sum of all positive odd values
 *  among the first n elements of arr
 *  Precondition: 1 <= n <= arr.length
 */
public static int addPositiveOddValues(int[] arr, int n)
{
  int sum = 0;
  < statement 1 >
  {
    < statement 2 >
      sum += arr[i];
  }
  return sum;
}
```

Which of the following are appropriate replacements for < *statement 1* > and < *statement 2* > so that the method works as specified?

	< *statement 1* >	< *statement 2* >
(A)	`for (int i = 1; i < n; i += 2)`	`if (arr[i] > 0)`
(B)	`for (int i = 0; i < n; i++)`	`if (arr[i] > 0 && arr[i] % 2 != 0)`
(C)	`for (int i = 1; i <= n; i += 2)`	`if (arr[i] > 0)`
(D)	`for (int i = 0; i <= n; i++)`	`if (arr[i] % 2 != 0)`
(E)	None of the above	

27. Consider the following class.

```
public class Question
{
  private boolean answer;

  public void flip(Question q)
  {
    < missing statement >
  }

  < constructors and other methods not shown >

}
```

Which of the following could replace < *missing statement* > in the `flip` method so that it compiles with no errors?

I. `answer = !answer;`

II. `answer = !q.answer;`

III. `q.answer = !q.answer;`

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

28. Which of the following statements will compile with no errors?

I. `ArrayList<Integer> nums = new ArrayList<Integer>();`
II. `List<Integer> nums = new ArrayList<Integer>();`
III. `ArrayList<Integer> nums = new List<Integer>();`

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

29. What is the output from the following code segment?

```
double pi = 3.14159;
int r = 100;
int area = (int)(pi * Math.pow(r, 2));
System.out.println(area);
```

(A) 30000
(B) 31415
(C) 31416
(D) 314159
(E) Depends on the particular computer system

30. Which of the following best describes the return value for the method `propertyX` below?

```
/** Precondition: v.length >= 2
 */
public boolean propertyX(int[] v)
{
  boolean flag = false;

  for (int i = 0; i < v.length - 1; i++)
  {
    flag = flag || (v[i] == v[i+1]);
  }

  return flag;
}
```

(A) Returns `true` if the elements of `v` are sorted in ascending order, `false` otherwise
(B) Returns `true` if the elements of `v` are sorted in descending order, `false` otherwise
(C) Returns `true` if `v` has two adjacent elements with the same value, `false` otherwise
(D) Returns `true` if `v` has two elements with the same value, `false` otherwise
(E) Returns `true` if all elements in `v` have different values, `false` otherwise

31. Which of the following is NOT a good reason to use comments in programs?

(A) To describe the parameters of a method
(B) To explain a convoluted piece of code
(C) To document which methods of a class are private
(D) To document requirements for correct operation of a method
(E) To document the names of the programmers and the date of the last change

32. Consider the following classes.

```
public class A
{
  public A() { methodOne(); }

  public void methodOne() { System.out.print("A"); }
}

public class B extends A
{
  public B() { System.out.print("*"); }

  public void methodOne() { System.out.print("B"); }
}
```

What is the output when the following code statement is executed?

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

(A) *
(B) *A
(C) *B
(D) A*
(E) B*

33. The following method is intended to remove from `List<Integer> list` all elements whose value is less than zero.

```
public void removeNegatives(List<Integer> list)
{
  int i = 0, n = list.size();

  while (i < n)
  {
    if (list.get(i) < 0)
    {
      list.remove(i);
      n--;
    }
    i++;
  }
}
```

For which lists of `Integer` values does this method work as intended?

(A) Only an empty list
(B) All lists that do not contain negative values in consecutive positions
(C) All lists where all the negative values occur before all the positive values
(D) All lists where all the positive values occur before all the negative values
(E) All lists

Questions 34-36 involve reasoning about classes and objects used in an implementation of a library catalog system.

An object of the class `BookInfo` represents information about a particular book, and an object of the class `LibraryBook` represents copies of a book on the library's shelves.

```
public class BookInfo
{
  private String title;
  private String author;
  private int numPages;

  < constructors not shown >

  public String toString()
  {
    return title + " by " + author;
  }

  public String getTitle() { return title; }
  public int getNumPages() { return numPages; }
}

public class LibraryBook
{
  private BookInfo info;
  private int numCopies;   // Number of copies on shelf

  < constructors not shown >

  public int getNumCopies() { return numCopies; }
  public void setNumCopies(int num)
                          { numCopies = num; }
  public BookInfo getInfo() { return info; }

  /** If there are copies on shelf, decrements
   *  the number of copies left and returns true;
   *  otherwise returns false
   */
  public boolean checkOut() { /* implementation not shown */ }
}
```

34. If `catalog` is declared in a client class as

```
LibraryBook[] catalog;
```

which of the following statements will correctly display *title* by *author* of the third book in `catalog`?

I. `System.out.println(catalog[2]);`

II. `System.out.println(catalog[2].getInfo());`

III. `System.out.println(catalog[2].getInfo().toString());`

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

35. Consider the following method from another class, a client of `LibraryBook`.

```
/** Returns the total number of pages in all
 *  books in catalog that are on the shelves
 */
public int totalPages(LibraryBook[] catalog)
{
  int count = 0;

  for (LibraryBook bk : catalog)
  {
    < statement >
  }
  return count;
}
```

Which of the following replacements for < *statement* > completes the method as specified?

(A) `count += bk.numCopies * bk.info.numPages;`
(B) `count += bk.getNumCopies() * bk.getNumPages();`
(C) `count += bk.(numCopies * info.getNumPages());`
(D) `count += bk.getNumCopies() * bk.getInfo().getNumPages();`
(E) None of the above

36. Which of the following code segments will correctly complete the `checkOut()` method of the `LibraryBook` class?

I.
```
if (getNumCopies() == 0)
{
  return false;
}
else
{
  setNumCopies(getNumCopies() - 1);
  return true;
}
```

II.
```
int n = getNumCopies();
if (n == 0)
{
  return false;
}
else
{
  setNumCopies(n - 1);
  return true;
}
```

III.
```
if (numCopies == 0)
{
  return false;
}
else
{
  numCopies--;
  return true;
}
```

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

Questions 37-38 refer to the following class `SortX`.

```
public class SortX
{
  public static void sort(String[] items)
  {
    int n = items.length;
    while (n > 1)
    {
      sortHelper(items, n - 1);
      n--;
    }
  }

  private static void sortHelper(String[] items, int last)
  {
    int m = last;
    for (int k = 0; k < last; k++)
    {
      if (items[k].compareTo(items[m]) > 0)
        m = k;
    }
    String temp = items[m];
    items[m] = items[last];
    items[last] = temp;
  }
}
```

37. Suppose `names` is an array of `String` objects:

```
String[] names =
          {"Dan", "Alice", "Claire", "Evan", "Boris"};
```

If `SortX.sort(names)` is running, what is the order of the values in `names` after two complete iterations through the `while` loop in the `sort` method?

(A) `"Boris", "Alice", "Claire", "Dan", "Evan"`
(B) `"Alice", "Claire", "Boris", "Dan", "Evan"`
(C) `"Alice", "Boris", "Claire", "Evan", "Dan"`
(D) `"Alice", "Claire", "Dan", "Evan", "Boris"`
(E) None of the above

38. If `items` contains five values and `SortX.sort(items)` is called, how many times, total, will `items[k].compareTo(items[m])` be called in the `sortHelper` method?

(A) 5
(B) 10
(C) 15
(D) 25
(E) Depends on the values in `items`

39. Consider the following two implementations of the method `getValue(double[] c, int n, double x)` that computes and returns the value of $c_0 + c_1x + c_2x^2 + ... + c_nx^n$.

Implementation 1

```
double getValue(double[] c, int n, double x)
{
  double value = 0.0, powx = 1.0;
  for (int k = 0; k <= n; k++)
  {
    value += powx * c[k];
    powx *= x;
  }
  return value;
}
```

Implementation 2

```
public static double getValue(double[] c, int n, double x)
{
  double value = c[n];
  for (int k = n-1; k >= 0; k--)
  {
    value = value * x + c[k];
  }
  return value;
}
```

What is the total number of arithmetic operations on floating-point numbers (additions and multiplications combined) that are performed within the `for` loop in Implementation 1 and Implementation 2, when $n = 5$?

	Implementation 1	Implementation 2
(A)	10	5
(B)	12	6
(C)	15	10
(D)	18	10
(E)	18	12

40. Consider the following method.

```
/** Returns the location of the target value
 *  in the array a, or -1 if not found
 *  Precondition: a[0] ... a[a.length - 1] are
 *                sorted in ascending order
 */
public static int search(int[] a, int target)
{
  int first = 0;
  int last = a.length - 1;

  while (first <= last)
  {
    int middle = (first + last) / 2;
    if (target == a[middle])
      return middle;
    else if (target < a[middle])
      last = middle;
    else
      first = middle;
  }
  return -1;
}
```

This method fails to work as expected under certain conditions. If the array has five elements with the values 3, 4, 35, 42, 51, which of the following values of `target` would make this method fail?

(A) 3
(B) 4
(C) 35
(D) 42
(E) 51