Practice Exam #4

SECTION I

Time — 1 hour and 30 minutes
Number of questions — 40
Percent of total grade — 50

1. Given the declarations

```
int p = 5;
int 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 code segment.

```
String band = "anamanaguchi";

System.out.println(band.substring(1, 4).
                        compareTo(band.substring(5, 8)));
```

What is printed when the code segment is executed?

(A) `true`
(B) `false`
(C) `0`
(D) A negative integer
(E) A positive integer

3. Consider the following class.

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

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

Which of the following statements about this code is true?

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

4. Consider the following code segment.

```
ArrayList<String> cities = new ArrayList<String>();
cities.add("Atlanta");
cities.add("Boston");
cities.add("Chicago");

for (String city : cities)
  city = city.substring(1);

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

What is printed when the code segment is executed?

(A) `[A, B, C]`
(B) `[C, B, A]`
(C) `[a, n, o]`
(D) `[tlanta, oston, hicago]`
(E) `[Atlanta, Boston, Chicago]`

5. Consider the following method.

```
public String filter(String str, String pattern)
{
  int pos = str.indexOf(pattern);

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

What is printed when the following statement is executed?

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

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

6. Consider the following statements.

```
String str = "\\*\n*\\";
System.out.println(str + " " + str.length());
```

What is printed when the statements are executed?

(A)
```
\*\n*\ 5
```
(B)
```
\**\*\" 6
```
(C)
```
\*\n*\ 8
```
(D)
```
\*
*\ 5
```
(E)
```
\\*
*\\ 7
```

7. Consider the following code segment.

```
int[] arr = {4, 3, 2, 1, 0};

for (int k = 1; k < arr.length; k++)
{
  arr[k-1] += arr[k];
}
```

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

(A) 4, 7, 5, 3, 1
(B) 4, 7, 9, 10, 10
(C) 7, 5, 3, 1, 0
(D) 7, 3, 2, 1, 0
(E) 10, 6, 3, 1, 0

8. Consider the following method.

```
public int countSomething(int[] arr)
{
  int m = arr[0];
  int count = 1;

  for (int k = 1; k < arr.length; k++)
  {
    int a = arr[k];
    if (a > m)
    {
      m = a;
      count = 1;
    }
    else if (m == a)
      count++;
  }

  return count;
}
```

For which of the following arrays will `countSomething` return 3?

(A) `int[] arr = {0, 1, 1, 1, 1};`
(B) `int[] arr = {1, 6, 5, 4, 0};`
(C) `int[] arr = {1, 0, 5, 6, 1};`
(D) `int[] arr = {3, 2, 1, 0, 5};`
(E) None of the above

9. Consider the following code segment.

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

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

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

10. Assume that the `boolean` variables `a`, `b`, and `c` have been properly declared and initialized. Which of the following expressions is `true` if and only if NOT all three variables `a`, `b`, and `c` have the same value?

(A) `a != b || b != c`
(B) `a != b && b != c`
(C) `a >= b && b >= c && c >= a`
(D) `a > b || b > c || a > c`
(E) `!(a == b || b == c || a == c)`

11. Consider the following method.

```
public void change(double[] nums, int n)
{
  for (int k = 0; k < n; k++)
  {
    nums[k] = 5.4;
  }
  n = 2;
}
```

What will be stored in `samples` and `len` after the following statements are executed?

```
double[] samples = {1.0, 2.1, 3.2, 4.3};
int len = samples.length;
change(samples, len);
```

(A) `samples` contains `5.4, 5.4, 5.4, 5.4`; `len` is `2`
(B) `samples` contains `5.4, 5.4, 5.4, 5.4`; `len` is `4`
(C) `samples` contains `1.0, 2.1, 3.2, 4.3`; `len` is `4`
(D) `samples` contains `5.4, 5.4`; `len` is `2`
(E) `samples` contains `1.0, 2.1`; `len` is `2`

12. Consider the following code segment.

```
for (int r = 1; r < mat.length; r++)
{
  for (int c = 1; c < mat[0].length; c++)
  {
    if ((r + c) % 2 == 0)
      mat[r][c] = 2 * mat[r - 1][c - 1] + c;
  }
}
System.out.println(mat[2][2]);
```

Suppose `mat` is declared as

```
int[][] mat = new int[3][4];
```

and holds the values

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

What is printed when the code segment is executed?

(A) 5
(B) 11
(C) 12
(D) 15
(E) 16

13. Consider the following code segment.

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

What is printed when the code segment is executed?

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

14. What happens when the size of an `ArrayList<Integer> numbers` has reached its capacity and `numbers.add(0)` is called?

    (A) The program throws an `IndexOutOfBoundsException`.
    (B) `numbers` remains unchanged; `add` returns `false`.
    (C) The first element of `numbers` is set to hold an `Integer` object with the value 0. The size of `numbers` remains unchanged.
    (D) The values in `numbers` are shifted toward the beginning (with the first element overwritten), and an `Integer` object with the value 0 is appended at the end of `numbers`. The size of `numbers` remains unchanged.
    (E) A larger array is allocated to hold the elements of `numbers`; the values from the old `numbers` array are copied into the new array, and an `Integer` object with the value 0 is appended at the end of `numbers`. The size of `numbers` is incremented by 1.

15. Suppose an array `arr` contains 127 different random values arranged in ascending order, and the most efficient searching algorithm is used to find a target value. How many elements of the array will be examined when the target equals `arr[39]`?

    (A) 4
    (B) 5
    (C) 7
    (D) 63
    (E) 64

16. Consider the following statement.

```
System.out.println(Math.pow(2, 3));
```

    What is the result when the statement is compiled/executed?

    (A) `8` is printed
    (B) `8.0` is printed
    (C) `9` is printed
    (D) Syntax error: "pow(double, double) in java.lang.Math cannot be applied to (int, int)"
    (E) `ArithmeticException` at run time

17. Consider the following code segment.

```
Integer n = 3;                // Line 1
Double x = 0.14;              // Line 2
System.out.println(n + x);    // Line 3
```

What is the result when the code segment is compiled/executed?

(A) `3` is printed
(B) `3.14` is printed
(C) Syntax error on Line 1
(D) Syntax error on Line 2
(E) `ArithmeticException` on Line 3

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

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

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

Consider the following code segment.

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

What is the result when the code segment is executed?

(A) `Anya` is printed
(B) `Cathy` is printed
(C) `Cathy Anya` is printed
(D) `Anya Cathy` is printed
(E) `IndexOutOfBoundsException`

19. Suppose we have two classes `Person` and `Student`, and the statements

```
double gpa = 3.8;
Person s = new Student("Isabella", gpa);
```

compile with no errors. Which of the following must be true?

I. `Person` has a constructor that takes two parameters: a `String` and a `double`.
II. `Student` has a constructor that takes two parameters: a `String` and a `double`.
III. `Student` has private fields `String name` and `double gpa`.

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

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

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

21. What is the result when the following code segment is executed?

```
Object[] nums = {null, null, null};
nums[0] = new Integer(2);
nums[1] = new Double(0.718);

System.out.println(nums[0].toString() +
        nums[1].toString() + nums[2].toString());
```

(A) `20.718` is printed
(B) `20.718null` is printed
(C) `NullPointerException`
(D) `StringIndexOutOfBoundsException`
(E) `ArrayIndexOutOfBoundsException`

Questions 22 and 23 refer to the following method.

```
private int product(int n)
{
  if (n <= 1)
    return 1;
  else
    return n * product(n-2);
}
```

22. What value does `product(6)` return?

(A) 1
(B) 8
(C) 12
(D) 48
(E) 720

23. When you call `product(25)`, it returns -1181211311, a negative number. Which of the following best explains this result?

(A) Integer arithmetic overflow
(B) Logic error that shows up for odd values of `n`
(C) Stack overflow error in recursive calls
(D) Small range of integers in the Java Virtual Machine installed on your computer
(E) A loss of precision in calculations

24. Consider the following code segment.

```
String abc = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
< missing statement >
System.out.println(abc.substring(k, k+1));
```

It is intended to print a randomly chosen letter of the alphabet. Any of the 26 letters can be chosen with equal probability. Which of the following can replace *< missing statement >* for the code to work that way?

(A) `int k = Math.random(25);`
(B) `int k = Math.random(0, 25);`
(C) `int k = Math.random(25) + 1;`
(D) `int k = (int)(26*Math.random());`
(E) `int k = (int)(25*Math.random()) + 1;`

25. Consider the following method.

```
/** Rearranges the elements in words according to
 *  the values stored in an integer array indices,
 *  so that the element of words at index indices[k]
 *  is moved to the element at index k.
 *  Precondition: words.size() == indices.length
 */
public void permute(ArrayList<String> words, int[] indices)
{
  ArrayList<String> temp = new ArrayList<String>();
  < missing code >
}
```

For example, after executing the code segment

```
ArrayList<String> words = new ArrayList<String>();
words.add("I");
words.add("am");
words.add("Sam");

int[] indices = {2, 0, 1};
permute(words, indices);
```

`words` will become the list `["Sam", "I", "am"]`. Which of the following code segments could replace < *missing code* > in the `permute` method?

I.
```
for (String word : words)
  temp.add(word);
for (int k = 0; k < indices.length; k++)
  words.set(k, temp.get(indices[k]));
```

II.
```
for (int j : indices)
  temp.add(words.get(j));
for (int k = 0; k < indices.length; k++)
  words.set(k, temp.get(k));
```

III.
```
while (words.size() > 0)
  temp.add(words.remove(0));
for (int j : indices)
  words.add(temp.get(j));
```

(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 class `Person` and its subclass `Friend`.

```
public class Person
{
  private String name;

  public Person(String nm) { name = nm; }

  public String getName() { return name; }
  public String toString { return getName(); }
}

public class Friend extends Person
{
  public Friend(String name) { super(name); }

  < methods not shown >
}
```

Which methods have to be defined in the `Friend` class for

```
System.out.println(new Friend("Li Min"));
```

to print `Li Min`?

(A) No particular methods are needed
(B) Only `getName()`
(C) Only `toString()`
(D) Both `getName()` and `toString()`
(E) No combination of methods will work because `name` is undefined in `Friend`

27. Given two arrays of `double` values sorted in ascending order, one with 100 elements and the other with 10 elements, how many comparisons will it take in an optimal algorithm to merge these arrays into one sorted array, in the best case and in the worst case?

	Best case	Worst case
(A)	10	109
(B)	50	110
(C)	100	110
(D)	109	999
(E)	100	1000

28. 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`

29. Consider the following method, intended to use Binary Search to find the location of `target` within `ArrayList<String> a`.

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

    if (compResult == 0)
      return middle;
    if (compResult < 0)
      last = middle - 1;
    else
      first = middle + 1;
  }
  return -1;
}
```

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

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

30. 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) 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

31. 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.

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

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

for (int k = 0; k <= 9; k++)
  digits.add("" + k);

for (int k = 0; k <= 4; k++)
{
  String d1 = digits.remove(k);
  String d2 = digits.remove(k);
  digits.add(k, d1 + "+" + d2);
}

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

(A) `[0+1, 1+2, 2+3, 3+4, 4+5]`
(B) `[0+1, 2+3, 4+5, 6+7, 8+9]`
(C) `[0+1, 1+2, 2+3, 3+4, 5, 6, 7, 8, 9]`
(D) `[0+1, 1+2, 2+3, 3+4, 4+5, 6, 7, 8, 9]`
(E) `[0+0, 1+1, 2+2, 3+3, 4+4, 5, 6, 7, 8, 9]`

33. A class `Turtle` has a `void` method `forward(int d)` that moves the turtle forward by `d` units.

A class `TurtleRace` has a list of `Turtle` objects:

```
private ArrayList<Turtle> runners;
```

`TurtleRace`'s method `move` is supposed to move all the turtles. Kim programmed it with a "for each" loop —

```
public void move(int d)
{
  for (Turtle t : runners)
    t.forward((int)(d*Math.random()));
}
```

— but Zane decided to replace that loop with a regular `for` loop:

```
public void move(int d)
{
  for (int i = 0; i < runners.size(); i++)
    runners.get(i).forward((int)(d*Math.random()));
}
```

What is a good reason for this change?

(A) No good reason; Zane 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`.

34. Suppose class *D* extends class *B* and has one constructor. In which of the following situations must the first statement in *D*'s constructor 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

35. In a regular pentagon, the ratio of the length of a diagonal to the length of a side is equal to the Golden Ratio (defined as $\frac{1+\sqrt{5}}{2} \approx 1.618$). Consider the following class `Pentagon`, which represents a regular pentagon.

```
public class Pentagon
{
  public static final double GOLDEN_RATIO =
                                   (1 + Math.sqrt(5.0)) / 2;
  private double side;

  public Pentagon (double x)
  {
    side = x;
  }

  public double getDiagonalLength()
  {
    return side * GOLDEN_RATIO;
  }
}
```

Which of the following code segments will compile with no errors and print the correct length of a diagonal in a regular pentagon with side 3.0?

I.
```
Pentagon p = new Pentagon(3);
System.out.println(p.getDiagonalLength());
```

II.
```
System.out.println(3.0 * Pentagon.GOLDEN_RATIO);
```

III.
```
System.out.println(3/2 * (1 + Math.sqrt(5.0)));
```

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

Questions 36-39 refer to the following class `House` and its subclass `HouseForSale`.

```
public class House
{
  private int mySize;

  public House(int size) { mySize = size; }

  public int getSize() { return mySize; }

  public void setSize(int size) { mySize = size; }

  public int compareToOther(House other)
  {
    return getSize() - other.getSize();
  }
}

public class HouseForSale extends House
{
  private int myPrice;

  public HouseForSale(int size, int price)
  {
    < missing statement >
    myPrice = price;
  }

  public int getPrice() { return myPrice; }

  public int compareToOther(House other)
  {
    return getPrice() - ((HouseForSale)other).getPrice();
  }

  /* other constructors, methods, and fields not shown */
}
```

36. Which of the following is the most appropriate replacement for < *missing statement* > in `HouseForSale`'s constructor?

(A) `mySize = size;`
(B) `setSize(size);`
(C) `super.setSize(size);`
(D) `super(size);`
(E) `super = new House(size);`

37. Suppose that while writing the `HouseForSale` class the programmer accidentally misspelled "compareToOther" in his code. What will happen when he tries to compile and run his class and the following statements in a client class?

```
HouseForSale house1 = new HouseForSale(2000, 129000);
HouseForSale house2 = new HouseForSale(1800, 149000);
System.out.println(house1.compareToOther(house2));
```

(A) The code compiles with no errors and prints `200`.
(B) The compiler fixes the spelling error based on the method name in the superclass; the code prints `-20000`.
(C) The compiler reports the syntax error "method compareToOther not found."
(D) The code compiles with no errors but throws a `NullPointerException`.
(E) The code compiles with no errors but throws a `ConcurrentModificationException`.

38. If the classes `House` and `HouseForSale` compile with no errors, which of the following declarations will result in a syntax error?

(A)
```
House[] houses = new House[2];
```

(B)
```
HouseForSale[] houses = {new House(2000), new House(1800)};
```

(C)
```
House[] houses = {new HouseForSale(2000, 129000),
                  new HouseForSale(1800, 149000)};
```

(D)
```
HouseForSale[] houses = {new HouseForSale(2000, 129000),
                         new HouseForSale(1800, 149000)};
```

(E) All of the above compile with no errors.

39. Which of the following is the most appropriate way to define the `getSize` method in `HouseForSale`?

(A) No definition is necessary, because the same code is already written in `House`.
(B) `public int getSize() { return mySize; }`
(C) `public int getSize() { return super.mySize; }`
(D) `public int getSize() { return super(mySize); }`
(E) `public int getSize() { return super.getSize(); }`

40. Which of the following code segments correctly traverses row-by-row a rectangular two-dimensional `int` array `m`?

(A)

```
for (int x : m)
{
  ...
}
```

(B)

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

(C)

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

(D)

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

(E)

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

Practice Exam #4

SECTION II

Time — 1 hour and 30 minutes
Number of questions — 4
Percent of total grade — 50

1. DNA sequencing and matching has become a standard technique in medical research, forensics, and recreational genealogy. A DNA molecule encodes genetic information as a sequence of nucleobases (or simply bases) of 4 kinds: adenine, cytosine, guanine, and thymine, denoted, respectively, by the letters A, C, G, and T. In this question you will work with strings that are sequences of these four letters.

   (a) Write a `boolean` method `isValidDNA` that checks whether a given string is a valid DNA string. A valid DNA string includes only the letters A, C, G, T. For example, `isValidDNA("GTACGTAATG")` should return `true`, while `isValidDNA("GXACGTAATG")` and `isValidDNA("GTAC_GTAATG")` should return `false`.

   Complete the `isValidDNA` method below.

```
/** Checks whether str is a valid DNA string. Returns
 *  true if str contains only the letters A, C, G, or T;
 *  otherwise returns false.
 *  Precondition: str is not null and not empty.
 */
public static boolean isValidDNA(String str)
```

   (b) A gene is a segment of DNA. A single gene can have hundreds, even thousands of bases; in our examples here we will use very short "genes." Write a method `matchTwoGenes` that takes two genes, `gene1` and `gene2`, and attempts to find `gene1` followed by `gene2` in a given DNA string. If it finds `gene1` followed by `gene2`, `matchTwoGenes` returns the gap (the number of letters) between `gene1` and `gene2` in the DNA string. The method always looks for the first occurrence of `gene1` and the first occurrence of `gene2`, which must follow `gene1`. If one of the two genes is not found, or `gene2` precedes `gene1` or overlaps with `gene1`, then the method returns `-1`. The table below shows a few examples with the values returned by the calls to `matchTwoGenes(dna, "ACA", "ACG")`.

dna	Return	Reason
"GT**ACA**TAATG**ACG**GT" (5)	5	
"GT**ACA**TAACAAG**ACG**GAC" (7)	7	
"GT**ACAACG**GTAA"	0	
"AAT**ACG**TAATG**ACA**GT"	-1	"ACG" found out of order
"GT**ACACG**AACAGT"	-1	"ACG" overlaps with "ACA"
"GT**ACA**TATTG**ACA**GT"	-1	"ACG" not found

Complete the method `matchTwoGenes` below.

```
/** Finds the first occurrence in a given DNA string of
 *  gene1 and gene2. Returns the gap between the matched
 *  genes if they are found in the correct order with no
 *  overlap; otherwise returns -1.
 *  Precondition: gene1 and gene2 are non-empty strings.
 */
public static int matchTwoGenes(String dna,
                                String gene1, String gene2)
```

For additional practice, write a method `matchGenes` that takes a DNA string and an `ArrayList` of genes (strings) and returns the sum of the gaps between the consecutive genes, if all of them are found in the DNA string in the correct order, with no overlaps. If one of the genes is not found or the order is wrong or two genes overlap, your method should return -1.

2. This question involves an implementation of a system, represented by the class `Car`, that tracks a car's fuel usage and average gas mileage. The class `Car` defines a constant (a `final double` variable) that represents the size of the car's gas tank (in gallons). `Car`'s constructor initializes that constant to a given value of the type `double`. The constructor also sets the amount of fuel left in the tank to full tank.

   The class `Car` provides the following methods:

   - `getGasLeft` — returns the amount of gas left in the tank
   - `fillUp` — resets the amount of fuel in the tank to full tank
   - `logTrip` — takes two parameters: the number of miles driven on a trip and the amount of gas used, represented as a fraction of tank size, and updates the total number of miles driven and the total amount of gas used by the car (assume the car had enough gas for the trip)
   - `mpg` — returns the average gas mileage in miles per gallon, rounded to the nearest integer

   For example, the statements

```
Car car = new Car(12.0);
car.logTrip(65, 1.0/4);
car.logTrip(35, 1.0/8);
System.out.println("MPG: " + car.mpg() +
        "  Gas left: "  + car.getGasLeft() + " gallons");
```

   print

```
MPG: 22  Gas left: 7.5 gallons
```

   because the car has driven $65+35=100$ miles and used $12\cdot\left(\frac{1}{4}+\frac{1}{8}\right)=4.5$ gallons of gas (with 7.5 gallons remaining in the tank). The average gas mileage after the two trips is $\frac{100}{4.5}=22.222\approx 22$ miles per gallon.

   Write the complete class `Car`, including the constructor and any required instance variables and methods. Your implementation must meet the specifications and conform to the above example.

   For additional practice, write and test a `boolean` method `canMakeTrip(int miles)`, which returns `true` if the required amount of gas for the trip (based on the average mpg at the beginning of the trip) plus 10% does not exceed the amount of gas left in the tank.

3. All emails received by the Office of Complaints in Appaloosa County are assigned a priority rank from 0 to 9 and answered in order of priority; emails of higher priority are answered first; emails with the same priority are answered in the order in which they were received. In the Java application that handles email in Appaloosa, each message is represented by an object of the class `Message` whose `getPriority` method returns the priority of the message (an integer from 0 to 9, inclusive).

   Messages of the same priority are held in a queue represented by an object of the class `Queue`. `Queue` has the following methods:

   - `public void add(Message msg)` — adds `msg` at the end of the queue
   - `public Message remove()` — removes and returns the first message from the queue
   - `public boolean isEmpty()` — returns `true` if the queue is empty; `false` otherwise

   An object of the `MessagePriorityQueue` class is used to store all incoming messages and retrieve them in order of priority, and in order of arrival from a queue of messages of the same priority. A `MessagePriorityQueue` object holds an array of ten queues, one for messages of each priority, from 0 to 9. The `MessagePriorityQueue` class also defines an instance variable that holds the total number of messages stored in all ten queues combined.

   In this question you will write two methods of the `MessagePriorityQueue` class. An incomplete definition of this class is shown below.

```
public class MessagePriorityQueue
{
  /** Array of queues; each queue holds messages of
   *  a particular priority. */
  private Queue[] queues;

  /** Total number of messages in all ten queues. */
  private int numMessages;

  /** Constructs a priority queue with ten empty queues,
   *  one for each priority.
   */
  public MessagePriorityQueue()
  { /* implementation not shown */ }

  /** Returns the total number of messages in all ten queues.
   */
  public int size()
  { /* implementation not shown */ }
```

```
    /** Returns true if all ten queues are empty; otherwise returns
     *  false.
     */
    public boolean isEmpty()
    { /* implementation not shown */ }

    /** Adds a message to the priority queue, as described in
     *  part (a).
     */
    public void add(Message msg)
    { /* to be implemented in part(a) */ }

    /** Removes and returns the first message of the highest
     *  priority from this priority queue, as described in
     *  part (b).
     */
    public Message remove()
    { /* to be implemented in part(b) */ }
}
```

(a) Write the method `add(Message msg)` of the `MessagePriorityQueue` class. This method retrieves the priority of `msg` and adds `msg` at the end of the queue that holds messages of that priority. `add` also increments `numMessages`.

Complete the method `add` below.

```
/** Adds msg at the end of the queue that holds
 *  messages of the same priority as msg; increments the
 *  total number of messages stored in this priority queue.
 */
public void add(Message msg)
```

(b) Write the method `remove` of the `MessagePriorityQueue` class. If the priority queue is not empty, this method removes the first message of the highest priority and returns it; it also decrements `numMessages`. If the priority queue is empty, `remove` returns `null`.

Complete the method `remove` below.

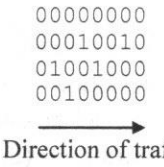
```
/** If this priority queue is not empty, removes and returns
 *  the first message of the highest priority from this
 *  priority queue and decrements the total number of
 *  messages in the priority queue; otherwise returns null.
 */
public Message remove()
```

For additional practice, write the class `Message`. Provide a constructor that accepts two parameters: the priority of the message and the text of the message. Include a suitable `toString` method.

Then write a class `Queue`. The easiest way to implement it is to simply derive `Queue` from `ArrayList<Message>`, adding one parameterless method `remove`, based on `ArrayList`'s `remove`.

Finally, implement the constructor and the `size` and `isEmpty` methods of the `MessagePriorityQueue` class, generate a few "complaint messages" of various priorities, and test your priority queue.

4. In a traffic modeling application, a segment of a highway is represented as a two-dimensional array of integers, in which each row represents one lane and each column represents a multilane cross-section at a given position. A value of 1 signifies a car is present, and a value of 0 signifies an empty slot. All cars move in the same direction. Cars can also make lateral moves ("switch lanes"). For example, the stretch of highway below has 4 lanes.



To simulate continuous traffic in a stretch of highway in the model, a car that goes out of bounds on the right reappears in the leftmost position of the same lane.

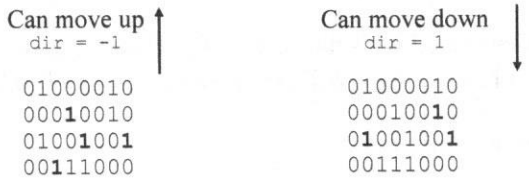
The model is implemented with the help of the class `Highway`. A partial definition of this class is shown below. You will write two methods of this class.

```
public class Highway
{
  /** hwy[lane][x] = 1 if there is a car at position x
   *  in lane; hwy[lane][x] = 0 if there is no car. */
  private int[][] hwy;

  /** Constructs an empty Highway with a given number of
   *  lanes and a given length.
   */
  public Highway(int lanes, int length)
  { /* implementation not shown */ }

  /** Places a car in a given lane at a given position. */
  public void addCar(int lane, int x)
  { /* implementation not shown */ }

  /** Returns true if the car can switch to an adjacent lane
   *  -- see part (a).
   */
  public boolean canSwitchLane(int lane, int x, int dir)
  { /* to be implemented in part (a) */ }

  /** All cars in hwy attempt to switch lanes with
   *  the probability 0.1 up and the same probability
   *  down; all cars that can make such a lateral move
   *  do so.
   */
  public void moveAllSideways()
  { /* implementation not shown */ }
```

Continued

```
    /** Moves all cars in all lanes forward by one position;
     *  a car in the last (rightmost) position moves into the
     *  first position (position 0) in the same lane.
     */
    public void moveAllForward()
    { /* to be implemented in part (b) */ }
}
```

(a) Write the `boolean` method `canSwitchLane` of the `Highway` class. A car at `hwy[lane][x]` can move in direction -1 (up) if `lane-1` exists and there are no cars at position `x` in any lanes above the current lane (lanes `lane-1`, `lane-2`, and so on). A car can move in direction +1 (down) if `lane+1` exists and there are no cars at position `x` in any lane below (lanes `lane+1`, `lane+2`, and so on). In the example below, the cars that can change lanes are shown in bold.



Complete the method `canSwitchLane` below.

```
/** A car can switch to an adjacent lane if that lane is
 *  within hwy bounds and there are no cars at the same
 *  position in other lanes in the direction of the lateral
 *  move.
 */
public boolean canSwitchLane(int lane, int x, int dir)
```

(b) Write the method `moveAllForward` of the `Highway` class. In this simplified version of the model, each car moves forward by one position. A car in the last position (rightmost column) is placed into the same lane in the leftmost position (column 0). You may not use any temporary arrays or lists. If a temporary array or list is used, your solution may not receive full credit.

Complete the method `moveAllForward` below.

```
/** Moves all cars in all lanes forward by one position;
 *  a car in the last (rightmost) position moves into the
 *  first position (position 0) in the same lane.
 */
public void moveAllForward()
```

For additional practice, write the constructor and the methods of the `Highway` class whose code is not shown. Then implement a more sophisticated version of `moveAllForward` in which each car moves halfway toward the current position of the next car in the same lane. (If a car has no other cars in front of it, use the first car in its lane as the "next" car.) Do not use any temporary arrays or lists.