CHAPTER 12 ~ ARRAYS AND ARRAYLISTS Name _____

EXERCISES 1-9, 11-14, 17, 18, 20, 26, 27, 29 Period _____

1. (a) Write a statement that declares an array of three integers, initialized to 1, 2, and 4.

   (b) Write an expression that represents the sum of the three elements of the above array (regardless of their current values)

2. Mark true or false and explain:

   (a) The following array has 101 elements:

   ```
   int[] x = new int[100];
   ```
   _____

   (b) Java syntax allows programmers to use any expression of the `int` data type as an array subscript. ____

   (c) The program, when running, verifies that all array subscripts fall into the valid range. _____

   (d) Any one-dimensional array object has a `length` method that returns the size of the array. _____

3. Write a method that takes an array of integers and swaps the first element with the last one.

4. An array of integers scores has at least two elements, and its elements are arranged in ascending order (i.e. `scores[i] ≤ scores[i+1]`). Write a condition that tests whether all the elements in scores have the same values. (Hint: you do not need iterations.)

5. Write a method `getRandomRps` that returns a character 'r', 'p', or 's', chosen randomly with odds of 3 : 5 : 6, respectively. (Hint: declare an array of chars and initialize it with values 'r', 'p', and 's', with each value occurring a number of times proportional to its desired odds. Return a randomly chosen element of the array.)

6. What does the `mysteryCount` method count?

```
private int mysteryCount(int[] v)
{
  int n = v.length, count = 0;

  for (int i = 0; i < n; i++)
  {
    if (v[i] != 0) break;
    count++;
  }
  return count;
}
```

7. If you take any two positive integers m and n ($m > n$), then the numbers a, b, and c, where

   $a = m^2 - n^2$; $b = 2mn$; $c = m^2 + n^2$

   form a Pythagorean triple:

   $a^2 + b^2 = c^2$

   You can use algebra to prove that this is always true. Write a method `makePythagoreanTriple` that takes two integer arguments, `m` and `n`, swaps them if necessary to make `m > n`, calculates the Pythagorean triple using the above expressions, places the resulting values a, b, and c into a new array of three elements, and returns that array. Test your method in a simple program.

8. Complete the following method:

```
// Returns an array filled with values
// 1, 2, ..., n-1, n, n-1, ..., 2, 1.
public static int[] createWedge(int n)
{
   ...
}
```

9. In SCRABBLE®, different letters are assigned different numbers of points:

```
A - 1   E - 1   I - 1   M - 3   Q - 10  U - 1   X - 8
B - 3   F - 4   J - 8   N - 1   R - 1   V - 4   Y - 4
C - 3   G - 2   K - 5   O - 3   S - 1   W - 4   Z - 10
D - 2   H - 4   L - 1   P - 3   T - 1
```

Write a method `computeScore(String word)` that returns the score for a word without using either `if` or `switch` statements. (Hint: find the position of a given letter in the alphabet string by calling `indexOf`; get the score for that letter from the array of point values, and add to the total.)

11. Mark true or false and explain:

(a) An `ArrayList` can contain multiple references to the same object. _____

(b) The same object may belong to two different `ArrayLists`. _____

(c) `ArrayList`'s `remove` method destroys the object after it has been removed from the list. _____

(d) `ArrayList`'s `add` method makes a copy of the object and adds it to the list. _____

(e) Two variables can refer to the same `ArrayList`. _____

12. Write a method that takes an `ArrayList` and returns a new `ArrayList` in which the elements are stored in reverse order. The original list should remain unchanged.

13. Write a method that removes the smallest value from an `ArrayList<Integer>`. (Hint: `Integer` has a method `compareTo(Integer other)` that returns the difference of this `Integer` and other.)

14. Write and test a method.

```
public void filter(ArrayList<Object> list1, ArrayList<Object> list2)
```

that removes from `list1` all objects that are also in `list2`. Your method should compare the objects using the == operator, not **`equals`**. (Hint: the `contains` and `indexOf` methods cannot be used.)

17. Find and fix the bug in the following code:

```
char[] hello = {' ', 'h', 'e', 'l', 'l', 'o'};
int i = 0;
// Shift to the left and append '!':
while (i < 6)
{
  hello[i-1] = hello[i];
  i++;
}
hello[5] = '!';
```

18. Write a method that determines whether a given number is a median for values stored in an array:

```
// Returns true if m is a median for values in the array
// sample, false otherwise. (Here we call m a median if
// the number of elements that are greater than m is the
// same as the number of elements that are less than m)
public boolean isMedian(double[] sample, double m)
```

20. Fill in the blanks in the following method that returns the average of the two largest elements of an array:

```
// Finds the two largest elements in scores
// and returns their average.
// Precondition: the size of the array is >= 2.
public static double averageTopTwo(int[] scores)
{
  int i, size = scores.length;
  int iMax1 = 0;        // index of the largest element
  int iMax2 = 1;        // index of the second largest element
  // If scores[iMax2] is bigger than scores[iMax1] --
  //   swap iMax1 and iMax2
  if (scores[iMax2] > scores[iMax1])
  {
    i = iMax1;
    _____________________________________
    _____________________________________
  }
  for (i = 2; i < size; i++)
  {
    if (scores[i] > scores[iMax1])
    {
      _____________________________________

      _____________________________________
    }
    else if ( _____________________________ )
    {
      _____________________________
    }
  }
  return _____________________;
}
```

26. A two-dimensional array `matrix` represents a square matrix with the number of rows and the number of columns both equal to `n`. Write a condition to test that an element `matrix[i][j]` lies on one of the diagonals of the matrix.

27. Write a method that returns the value of the largest positive element in a 2-D array, or 0 if all its elements are negative:

```
// Returns the value of the largest positive element in
// the matrix m, or 0, if all its elements are negative.
private static double positiveMax(double[][] m)
```

29. Let us say that a matrix (a 2-D array of numbers) `m1` "covers" a matrix `m2` (with the same dimensions) if `m1[i][j] > m2[i][j]` for at least half of all the elements in `m1`. Write the following method:

```
// Returns true if m1 "covers" m2, false otherwise.
// Precondition: m1 and m2 have the same dimensions.
private static boolean covers(double[][] m1, double[][] m2)
{
  ...
}
```