Problem Set Four AP Computer Science

Unit 4 - Arrays **Due: 1/14/2019**

Instructions

Create a Java Project called LastNameFirstNamePS4. Within that project, create a class called LastNameExercises. Be sure that it has a main method.

Your work is due the day of the unit exam, but you are expected to answer questions after each class.

Scoring

You may score up to 6 points for each question.
-2 pts: it works
-2 pts: properly uses new techniques
-2 pts: follows the style guide (see below)

Submission

Zip the file and submit it to our assignment in Google Classroom. Be sure the zip file is named LastNameFirstNamePS1. Visit the class website for further instructions.

Style Guide

Unit 4

Arrays

- -All data should be of same type.
- -Simplest way to traverse: for(int i = 0; i < arr.length; i++) { something(arr[i]);}
- -Use for-each when you don't want to change the array itself, for loop when you do.
- -Avoid returning arrays as a fix for returning multiple values from a method.
- -If you know the array's values up front, initialize in one line with the values:

```
int[] arr = {3, 5, 2, 8};
```

Unit 3

Parameters

- -Include parameters that can make a method more general.
- -Don't include parameters that are never used or that are the same every time.

If/else statements

- -Should never have any code in common.

While loops

- -Use when you're not sure how many time code will be executed.
- -Use sentinels to keep your eye on a particular target value for stopping the loop.

-Use fencepost design to make the first or last iteration different.

Booleans

-Never say A==true or A!=true; just use A or !A.

-Never code if(A==true){return true;} else{return false;}. Just: return A;

-Remember that && and || only operate on booleans. Therefore you must:

replace: A || B == 1 with: A == 1 || B == 1

-Avoid needless parentheses.

Objects

-Scanner, Random objects should be made in main method and passed to methods.

-Place all import statements at the very top of the class, only after identifying comments.

<u>Unit 2</u>

Variables

-Use the correct data type - int for whole numbers, double for more precision.

-Create descriptive variable names.

-Variable names should follow the same conventions as method names.

Class Constants

-Use these for a value that is the same for the whole run of the program.

-Naming convention: all capital letters separated by underscores

-Keywords: public static final

For loops

-Use for code that repeats a set number of times (more than once).

<u>Unit 1</u>

General

-Place each statement on its own line.

-Use spaces and indentation to make code legible.

-Use comments to explain the purpose of complicated code.

Naming Conventions

-Classes: Start with a capital letter and capitalize every word that follows. Ex: ClassName

-Methods: Start with a lowercase letter and capitalize every word that follows. Ex: methodName

Methods

-Use static methods to break your program into reusable pieces to reduce redundancy.

-Avoid trivial methods that do not accomplish much.

-Leave a blank line between methods.

-Pairs of curly braces should be easily identifiable.

Printing

-For a blank line, use System.out.println(); not System.out.println("");

-To print text on the same line, only use one print statement.

-Use println() rather than including \n at the end of a print() statement.

Formatting

Indentation

-Within a set of curly braces, indent one level with tab or 3 spaces.

-Be consistent at each level.

-Keep braces visible. Do so by closing the curly brace on the same level as the header with the opening {.

-Loops or if/else with just line don't need curly braces, but should be indented.

Spacing

-Use spaces liberally.

Good spacing	Bad spacing
`int i = methodCall() * j + j % 4;`	`int i=methodCall()*j+j%4;`
`for (int i = 0; i < 10; i++) {`	`for(int i=0;i<10;i++){`
`while (sum < target) {`	`while(sum<target){`
`public void myMethod(int x, int y) {`	`public void myMethod (int x,int y){`
`methodCall(param1, param2);`	`methodCall (param1,param2);`
`public class Foo {`	`public class Foo{`
`int[] myArray = new int[x + 1];`	`int[] myArray=new int[x+1];`

Commenting

-Begin each class with a comment of your name and a description of the program.

-Comment each method with a description of what it does, what the parameters mean, what it will return, and whether there are any parameter restrictions.

-Comments describe what something does, not how it does it.

-Unless you are doing something unexpected or unusual! Then explain it.

-Single line: //comment

-Multiple lines: /* comment
comment */

Problems

1. December 12th
Write your code in a method named: printArray under the comment:
//Exercise 1

Your method will accept no parameters and return nothing.
Create an array of 5 booleans and print each one out using a for-loop.

Optional further exercises: Practice-It: 7.1 - arrayDeclarationSyntax, 7.3 - dataArray

2. December 13
Write your code in a method named: checkOver100 under the comment:
//Exercise 2

Ask a user to input ten numbers into an array. Use a for-each loop to keep track of whether any of the numbers are over 100. If any numbers are over 100, print "Value over 100 found." If no numbers are over 100, print "No values over 100 found."

Optional further exercises: Practice-It: 7.1 - arrayDeclarationSyntax, 7.3 - dataArray

3.
Write your code in a method named: sideSwap under the comment:
//Exercise 3

Write a method that swaps each element of the array with the one after it. If there is an odd number of elements, leave the last one as it is.

Optional further exercises: Practice-It: 7.22 - arrayCodeTracing3

4.
Write your code in a method named: doubleArray under the comment:
//Exercise 4
Write a method that takes in one array as a parameter and returns an array that is two copies of the parameter. Ex: [3, 4, 5] → [3, 4, 5, 3, 4, 5]

5.
Write your code in a method named: mode under the comment:
//Exercise 5

Write a method called mode that returns the most frequently occurring element of an array of integers. Assume the array has at least one element and that every element in the array has a value between 0 and 100 (inclusive). Break ties by choosing the lower value. For example, if the array passed contains the values [27, 15, 11, 15, and 27], your method should return 15.

Hint: You may wish to look at the Tally program in your textbook (Chapter 7) to get an idea how to solve this problem.