Code-Writing Quiz 5 Prep

The goal of code-writing quizzes is to help students gain fluency and confidence in their code-writing.

During the quiz students will be given 1 – 3 very short problems and about 15 min. to implement the solution with a pencil/pen on a piece of paper. Then each student will be randomly paired with a peer and the peers will assess each-other's work. All the errors must be marked with a red pen and the quiz will be graded according to a simple rubric. The rubric will be provided. Instructor will collect all the quizzes to quickly check on the works and to post grades in the gradebook.

Students are expected to write syntactically and logically correct Java code. Import statements and main() method heading will not be needed though.

The quiz is closed-book, no-computer-access type of exercise. Students are encouraged to prepare notes on a 3x5 inch index card using both sides. That card is the only reference students can use during the quiz and eventually during the midterm and final exams. It is necessary to spend time practicing for the quiz. It's close to impossible to do well on such a quiz without prior practice. Problems provided below are very similar to the one(s) that will be appearing on the quiz.

If the student missed the quiz and wants to get credit for it, he/she must submit to the instructor hand-written code solutions to all the sample problems listed in the quiz preparation guide.

Topics to Review

- Writing class definitions for classes with fields of non-primitive type
- Accessor and mutator methods.
- Accessor method for reference field (not of primitive type)
- Mutator methods that throw exceptions
- Writing copy constructor that creates deep copy of non-primitive fields
- Writing equals() method
- STATIC methods (make sure to know the limitations here!)
- STATIC fields

Sample Problems

Given a class described below:

```
public class Rectangle {

   private double length;

   private double width;

   public Rectangle(double len , double wid) {/*implemented*/}public
   Rectangle(){/*implemented*/}
```

```
    public Rectangle(Rectangle obj) {/*implemented*/}

    boolean equals(Rectangle) {/*implemented*/}

    public String toString(){/*implemented*/}

    /* accessor and mutator methods implemented*/

}
```

Write a class MyClass that has 2 fields: Rectangle r and int aField. Implement the following methods for the new class:

1. Accessor for field r. Make a deep copy of the field when returning it.
2. Constructor MyClass(int a, Rectangle b) that sets both fields. Make a deep copy of object b when copying it!
3. No-argument constructor that sets aField to 0, and r field to a rectangle object with length and width set to 1.
4. Mutator for aField. The value stored in that field must not be negative. Throw Illegal ArgumentException when the negative value is detected.
5. copy constructor for MyClass
6. toString() method for MyClass
7. equals() method for MyClass

I assume you know all topics from previous Quizzes and remember how to solve problems listed below.

Quiz 4 Topics to Review

- Writing simple classes
- Writing class constructors – no-argument constructors and constructors that take parameter
- Writing accessor and mutator methods
- Reading simple UML diagrams
- Array analysis algorithms:
    - Finding minimum element
    - Finding maximum element
    - Finding sum of all elements (and average)
    - Linear search: find given element in array
    - Count occurrences of a number in array
- Writing methods that take array as parameter and modify the array
- Writing methods that return array
- Creating an array of objects. Doing simple analysis on that array.

Quiz 4 Sample Problems

1. Write a class Box that represents a 3D box.
    - The class must have 3 fields that represent dimensions: length, width, height – all of type double
    - The class must have two constructors: non-argument constructor and constructor that takes all dimensions as parameters and sets the fields. When setting the dimensions check for values to be positive. Throw IllegalArgumentException if they are negative or 0.
    - Accessor and mutator methods for all fields (total of 6). Use exceptions to prevent negative values to be used when setting dimensions.
    - getVolume() method that calculates and returns volume of the box.
    - getPerimeter() method that returns the sum of all edges of the box.
2. Assume you have Box class described above. Declare an array of 100 boxes and an ArrayList for 5 elements of type Box.
3. Assume you have an array of 100 objects of type Box. Use linear search algorithm to find a box with volume 100.
4. Assume you have an array of 100 objects of type Box. Search the given array for all boxes that have a perimeter of 10. Store all of those objects in ArrayList named "boxes10".
5. Implement the following array algorithms as methods:
    - Finding minimum element
    - Finding maximum element
    - Finding sum of all elements
    - Finding an average of all elements
    - Linear search: find given element in array
    - Count occurrences of a number in array
    - Count occurrences of a string in array of strings

Quiz 3 Topics

- Writing methods that take parameters and don't return a value
- Writing methods that take parameters and return a value
- Methods that return numeric, string, and Boolean results.
- Calling methods using user input or random numbers generated in a loop.
- Turning segments of code you practiced to write for Quiz 1 and Quiz 2 into methods.
- Throwing exceptions from methods:
    - Propagating exception from the method. Example: opening file in a method and not handling exceptions in the method but propagating them up to the calling code
    - Throwing a new exception from the method. Example: argument validation and throwing IllegalArgumentException

Quiz 3 Sample Problems

6. Write a method that calculates and returns area of rectangle. Call method using user input to pass as arguments.
   Method heading: double area (double length, double width)
   - Add code that checks for length and width to be positive numbers. Throw IllegalArgumentException if length or width are negative. Add code to the method header that announces that the method can throw exceptions in the code.
7. Assume you have a method described in problem 1. Generate 50 pairs of random integers in the range [1, 100], and calculate 50 areas of rectangle, passing a pair of random numbers into the method as arguments on each iteration of the loop. Use Java formatting to output randomly generated length, width, and calculated area in columns, to create table-like look on the screen. There is no need to catch IllegalArgumentException.
8. Write a method that compares two numbers and returns the largest one. Call method using user input to pass as arguments. Method heading: int max (int, int);
   Sample calling code that explains functionality:
   int k = max(10, 5);
9. Write a method that checks if the number is positive. Call method using user input to pass as arguments. Method heading: bool isPositive (double)
10. Write a method that takes a file name as a parameter, reads all numbers from the file, echo-prints them into the screen, finds and returns the sum of all the numbers it printed. The method must be throwing IOException. We assume that there are no errors in the file and each number is placed on a separate line.
    - Practice writing calling code for the method. Use try/catch block to catch exceptions the method throws.

Quiz 2 Topics

- Exceptions basics – how to catch an exception of specified type.
- User input validation
  - Validating the range of values
  - Validating the TYPE of value with the help of try/catch block and exceptions
- File I/O
  - Opening text file for reading, reading from a text file all the lines using a loop, closing the text file
  - Opening text file for writing, writing into the file using a loop, closing the text file
  - Catching I/O exceptions
- Generating random numbers in a given range
- Simple data analysis:
  - Find the total (sum) of all the numbers
    - coming from user input or

        - generated with random number generator
    - Find the smallest and largest number in a set coming from
        - coming from user input or
        - generated with random number generator

Quiz 2 Sample Problems

11. Ask user to provide an integer in the range from 1 to 100. Validate user input: make sure user provides number in the given range AND that the input provided is of integer type. Use validation loop to prevent user from moving on until the value of correct type and range is provided.
12. Generate 200 numbers in the range from 5 to 55. Print them out on the screen.
13. Generate 100 random numbers in the range from 1 to 10. Calculate the sum of the numbers and print it into the screen.
14. Generate 100 random numbers in the range from 1 to 10. Print into the screen only even numbers out of those that were generated.
15. Generate 100 random numbers in the range from 1 to 10. Find out how many times number 3 has been generated. Output that info into the screen.
16. Use loop to ask user to input an integer number 25 times. Find the total of all the numbers the user entered. No input validation is required.
17. Use loop to ask user to input an integer number 50 times. Find the smallest and the largest of all the numbers the user entered. No input validation is required.
18. Open text file named “myFile.txt” for reading. Print the contents of the file into the screen. (Close the file). Catch and handle I/O exceptions.
19. Open text file named “numbers.txt” for writing. Write all even integers in the range 1 – 100 into the file. (Close the file) Catch and handle I/O exceptions.

Quiz 1 Topics

- Input using Scanner object to input integers, doubles, and lines from keyboard
- Output using .println() method
- Calculation of any sort
- Use Java formatting to format output and use printf() method to manipulate the output appearance and formatting.
- Using if, if/else, is/else if, switch and nested if statements
- Logical operators and Boolean logic
- Using simple “while” and “for” loops for repetition.

Quiz 1 Sample Problems

1. Get two values from the user input, width and length of the rectangle, and calculate the rectangle area using formula area = width*length. Output the resulting value showing two digits after the decimal point.
2. Input three numbers from the user. Use nested “if” statements to print them out in order from the smallest to the largest.
3. Write a loop that prints “Hello WORLD!” 10 times.
4. Use a loop to ask user to input 20 integers, one at a time. Analyze the input as it comes and print out the number only if it is even.
5. Use loop to output numbers 1 – 100 and their squares. Use Java formatting and printf() to organize output in two columns: Number and Square.
6. Use “for” loop to output a table of Celsius temperatures and their Fahrenheit equivalents. Make Celsius temperatures change from -10 to 40 degrees with a step of 5. For conversion use the following formula. Beware of integer division!

$$T_{(°F)} = T_{(°C)} \times 9/5 + 32$$

7. Use Java formatting and printf() to output number double k = 345.6789 in a field that is 10 spaces wide and rounded to 2 decimal places.