Java Journal Template

Directions: Follow the directions for each part of the journal template. Include in your response all the elements listed under the Requirements section. Prompts in the Inspiration section are not required; however, they may help you to fully think through your response.

Remember to review the Touchstone page for entry requirements, examples, and grading specifics.

Name: Warren Spears

Date: 4/13/2024

Final Replit Program Join Link: https://replit.com/join/ephoekbdxo-warrenspearsii

Complete the following template. Fill out all entries using complete sentences.

PART 1: Defining Your Problem

Task State the problem you are planning to solve. **Requirements** ● Describe the problem you are trying to solve. ● Describe any input data you expect to use. ● Describe what the program will do to solve the problem. ● Describe any outputs or results the program will provide. **Inspiration** When writing your entry below, ask yourself the following questions: ● Is your problem clearly defined? ● Why do you want to solve this particular problem? ● What source(s) of data do you believe you will need? Will the user need to supply that data, or will you get it from an external file or another source? ● Will you need to interact with the user throughout the program? Will users continually need to enter data in and see something to continue? ● What are your expected results or what will be the end product? What will you need to tell a user of your program when it is complete?
This will be a calculator program used to calculate the corrected gravity of a liquid to 60 Degrees F. This is used throughout the Oil and Gas industry to correct the volume to a standard 60 degrees for all sales points. It will receive from the user two inputs. The first input will be the Observed Temperature and the second input will be the Observed Gravity. The program will then perform calculations to correct the gravity. This will be a handy project to quickly calculate the information for reference only. The output to the user will be the Corrected Gravity at 60 Degrees F.

PART 2: Working Through Specific Examples

Task
Write down clear and specific steps to solve a simple version of your problem you identified in Part 1.

Requirements
Complete the three steps below **for at least two distinct examples/scenarios**.

- State any necessary input data for your simplified problem.
- Write clear and specific steps in English (not Java) detailing what the program will do to solve the problem.
- Describe the specific result of your example/scenario.

Inspiration
When writing your entry below, ask yourself the following questions:

- Are there any steps that you don't fully understand? These are places to spend more time working out the details. Consider adding additional smaller steps in these spots.
- Remember that a computer program is very literal. Are there any steps that are unclear? Try giving the steps of your example/scenario to a friend or family member to read through and ask you questions about parts they don't understand. Rewrite these parts as clearly as you can.
- Are there interesting edge cases for your program? Try to start one of your examples/scenarios with input that matches this edge case. How does it change how your program might work?

Explain the project to the user:

1. Instruct the user on the program
2. Define the data needed from the user
3. Receive data from user

Calculation:

1. Create and initialize private constant variables
2. Create private variables for calculations
3. Calculate initial variables
4. Iterate over the remaining calculations the required 10 iterations.

Scenario 1:

1. User inputs are correctly formatted Observed Temp
2. User inputs are correctly formatted Observed Gravity
3. Calculate the output

Scenario 2:

1. User inputs are incorrectly formatted Observed Temp
2. User inputs are incorrectly formatted Observed Gravity
3. Display message to user that values are incorect
4. Calculate based on private final variables to prevent errors

PART 3: Generalizing Into Pseudocode

Task
Write out the general sequence your program will use, including all specific examples/scenarios you provided in Part 2.

Requirements
- Write pseudocode for the program in English but refer to Java program elements where they are appropriate. The pseudocode should represent the full functionality of the program, not just a simplified version. Pseudocode is broken down enough that the details of the program are no longer in any paragraph form. One statement per line is ideal.

Help With Writing Pseudocode
- Here are a few links that can help you write pseudocode with examples. Remember to check out part 3 of the Example Journal Template Submission if you have not already. Note: everyone will write pseudocode differently. There is no right or wrong way to write it, other than to make sure you write it clearly and in as much detail as you can so that it should be easy to convert to code later.
    - https://www.geeksforgeeks.org/how-to-write-a-pseudo-code/
    - https://www.wikihow.com/Write-Pseudocode

Inspiration
When writing your entry below, ask yourself the following questions:
- Do you see common program elements and patterns in your specific examples/scenarios in Part 2, like variables, conditionals, functions, loops, and classes? These should be part of your pseudocode for the general sequence as well.
- Are there places where the steps for your examples/scenarios in Part 2 diverged? These may be places where errors may occur later in the project. Make note of them.
- When you are finished with your pseudocode, does it make sense, even to a person that does not know Java? Aim for the clearest description of the steps, as this will make it easier to convert into program code later.

```
Method Calculate(double Temperature, double Observed Gravity)

   Calculate the Delta of the Temp to 60 Degrees F

   Square the Delta

   Calculate the RHO based on the Observed Gravity

   Calculate the HYC

   Calculate the RHOT

   Calculate the initial Alpha

   Calculate the Corrected Gravity
```

```
for (I is = to 0; I is less than 10; add 1 to I)

  Calculate the Alpha Squared

  Calculate the VCF

  Calculate the RHO60

  Calculate the ALPH

  Calculate the final Corrected Gravity @60 Degrees F

  Return the final Corrected Gravit
```

PART 4: Testing Your Program

Task
While writing and testing your program code, describe your tests, record any errors, and state your approach to fixing the errors.

Requirements
- For at least one of your test cases, describe how your choices for the test helped you understand whether the program was running correctly or not.

For each error that occurs while writing and testing your code:
- Record the details of the error from Replit. A screenshot or copy-and-paste of the text into the journal entry is acceptable.
- Describe what you attempted in order to fix the error. Clearly identify which approach was the one that worked.

Inspiration
When writing your entry below, ask yourself the following questions:
- Have you tested edge cases and special cases for the inputs of your program code? Often these unexpected values can cause errors in the operation of your program.
- Have you tested opportunities for user error? If a user is asked to provide an input, what happens when they give the wrong type of input, like a letter instead of a number, or vice versa?
- Did the outcome look the way you expected? Was it formatted correctly?
- Does your output align with the solution to the problem you coded for?

During testing I entered an incorrect value into the Temperature field and received the error below:

```
Exception in thread "main" java.util.InputMismatchException
   at java.base/java.util.Scanner.throwFor(Scanner.java:939)
   at java.base/java.util.Scanner.next(Scanner.java:1594)
   at java.base/java.util.Scanner.nextDouble(Scanner.java:2564)
   at Main.main(Main.java:28)
```

This is a common error when entering data into project files. More commonly known as "Fat Fingering" the keyboard. I corrected with a simple try catch block with an error message for the user. After the try catch is executed the project completes with a standard computation.

Correction:

```
// Try Catch NAN values
    try{
      temp = myObj.nextDouble();
    }
    catch(Exception e){
      System.out.println(user.errorMessage);
    }
```

PART 5: Commenting Your Program

Task
Submit your full program code, including thorough comments describing what each portion of the program should do when working correctly.

Requirements
- The purpose of the program and each of its parts should be clear to a reader that does not know the Java programming language.

Inspiration
When writing your entry, you are encouraged to consider the following:
- Is each section or sub-section of your code commented to describe what the code is doing?
- Give your code with comments to a friend or family member to review. Add additional comments to spots that confuse them to make it clearer.

```
import java.util.Scanner;  // Import the Scanner class

public class Main {
  public static void main(String[] args) {
    // Standard Inputs
    double temp = 60.0f;
    double obsGravity = 32.0f;


    // Create Scanner Object to get inputs from user
    Scanner myObj = new Scanner(System.in);
    // Create a User Object
    User user = new User();
    // Print out Greetings and Instructions
    System.out.println(user.getGreeting());
    System.out.println(user.getInstructions());
    System.out.println(user.getExampleData());

    // Get and Set user inputs
    System.out.println(user.setTemp());

    // Try Catch NAN values
    try{
      temp = myObj.nextDouble();
    }
    catch(Exception e){
      System.out.println(user.errorMessage);
    }

    System.out.println(user.setObsGravity());

    // Try Catch NAN values
    try
      {
      obsGravity = myObj.nextDouble();
      }
    catch(Exception e)
      {
        System.out.println(user.errorMessage);
      }

    // Create a FiveATable Object and perform the calculations
    FiveATable fiveATable = new FiveATable();
    double cg = fiveATable.CalcFiveATable(temp, obsGravity);
    System.out.println(user.setOutput() + cg);

    // Close Scanner Object
    myObj.close();
  }
}
```

```
public class FiveATable {
  // Private Variables
  // Private Variables
  private double OTEMP;
  private double OG;
  private double RHO;
  private double HYC;
  private double RHOT;
  private double VCF;
  private double ALPHA;
  private double ALPHASQD;
  private double RHO60;
  private double DELTA;
  private double DELTASQD;
  private double CG;
  private double temp;

  // Constant Variables
  private final double C1 = 141.5f;
  private final double C2 = 999.012f;
  private final double C3 = 131.5f;
  private final double C4 = 0.00001278f;
  private final double C5 = 0.0000000062f;
  private final double K0 = 341.0957f;
  private final double CE = 2.71828182845904f;

  public double CalcFiveATable(double OTEMP, double OG) {
    this.OTEMP = OTEMP;
    this.OG = OG;

    // Calculate the Delta of the Temp to 60 Degrees F
    DELTA = OTEMP - 60.0F;
    DELTASQD = (double) Math.pow(DELTA, 2.0F);

    // Calculate the RHO based on the Observed Gravity
    RHO = C1 * C2 / (C3 + OG);

    // Calculate the HYC
    HYC = 1 - C4 * DELTASQD - C5 * DELTASQD;

    // Calculate the RHOT
    RHOT = HYC * RHO;

    // Calculate the initial Alpha
    ALPHA = K0 / RHOT / RHOT;
    // (double) Math.pow(CE, VCF[i])

    // Calculate the Corrected Gravity
    for (int i = 0; i < 10; i++) {

      // Calculate the Alpha Squared
      ALPHASQD = (double) Math.pow(ALPHA, 2.0f);

      // Calculate the VCF
      temp = (ALPHA * -1.0f) * DELTA - 0.8f * ALPHASQD * DELTASQD;
      VCF = (double) Math.pow(CE, temp);

      // Calculate the RHO60
      RHO60 = RHOT / VCF;

      // Calculate the ALPHA
      ALPHA = K0 / RHO60 / RHO60;

      // Calculate the final Corrected Gravity @60 Degrees F
      CG = (C1 * C2 / RHO60) - C3;

    }
    // Return the final Corrected Gravit
    return CG;
  }
}

public class User {
  // Greeting
  String greeting = "Welcome: This program will calculate the corrected gravity of a liquid from     an
observed Temperature to a corrected 60 Degrees F.";
  // Instructions
  String instructions = "You will be asked to enter some informaiton: \n" + "1. The Observed
Temperature \n" + "2. The Observed Gravity \n";
  // Example Data to Enter:
  String exampleData = "Example Data: \n" + "1.  Temperature: 89 \n" + "2.  Gravity: 38 \n";

  // Enter Temperature
  String enterTemp = "Enter the Observed Temperature: ";

  // Enter Gravity
  String etnerObsGravity = "Enter the Observed Gravity: ";

  // Output Message
  String output = "The corrected gravity is: ";

  // Error Message
  String errorMessage = "Please enter a number in the format from the instructions" + "\n" + "The
Corected Gravity is defaulted to a: 32 Gravity @ 60 Degrees F";

  // Greeting
  public String getGreeting() {
    return greeting;
  }

  // Instructions
  public String getInstructions() {
    return instructions;
  }

  // Example Data to Enter
  public String getExampleData() {
    return exampleData;
  }

  // Enter Temperature
  public String setTemp() {
    return enterTemp;
  }

  // Enter Gravity
  public String setObsGravity() {
    return etnerObsGravity;
  }

  // Display Output Message
  public String setOutput() {
    return output;
  }

  // Error Message
  public String setErrorMessage(){
    return errorMessage;
  }

}
```

PART 6: Your Completed Program

Task
Provide the Replit link to your full program code.

Requirements
- The program must work correctly with all the comments included in the program.

Inspiration
- Check before submitting your Touchstone that your final version of the program is running successfully.

https://replit.com/join/ephoekbdxo-warrenspearsii