CSE 2221 - Project 2

Task

Gain familiarity of **double** variables, **while** loops, and **static** methods by creating a program that computes the root of a number using Newton iteration

Original Project Instructions

Project 2 Instructions from CSE2221 Project Site

Program Requirements

- Repeatedly ask the user of the program if they wish to calculate another square root
- If the user’s response is ”y” then proceed, any other value should terminate the program
- If the program proceeds, the program should prompt the user to enter a positive **double** value, you may assume the user will always enter a positive **double**
- Compute the square root of this user-inputted value using Newton iteration using a relative error of 0.01% (or 0.0001 in decimal)
- Display the result of this computation
- Program should then ask the user if they wish to compute another square root, and so on

Summary of Newton Iteration

- Start with an initial guess. As mentioned in the original instructions, $r = x$ is a good initial guess
- Repeatedly assign $r = (r + x \ / \ r) \ / \ 2$ until $(\ | \ r^2 - x \ | \ / \ x \ ) < \epsilon^2$
- Note: the above iteration is for solving a SQUARE root, the algorithm will change slightly to compute any k-th root (which you will do if you choose to do the additional activities)

Steps

1. Copy and paste *ProjectTemplate* to create a new project folder for this project
2. Name the project *Newton*
3. Open the *src* folder, then open *(default package)*
4. Rename *ProgramWithIOAndStaticMethod.java* to *Newton1.java*
5. Delete the other files
6. Open *Newton1.java*
7. Update the JavaDoc comments above the class declaration (i.e. program description and author name)
8. Replace the `myMethod` static method with the `sqrt` static method provided below (also provided in the original project instructions)

```
/**
* Computes estimate of square root of x to within relative error 0.01%.
*
* @param x
*            positive number to compute square root of
* @return estimate of square root
*/
private static double sqrt(double x) {

}
```

9. Add the necessary code to satisfy the project requirements (i.e. computing the square root of a number using Newton iteration given some error)

10. Copy and paste *Newton1.java* to create a new file, name the new file *Newton2.java*

11. Open *Newton2.java*

12. Edit the `sqrt` static method in *Newton2.java* to allow $x = 0$ (i.e. no division by 0).

    Note: Your project may appear to work without making any changes, this is incorrect. You must add the necessary check to ensure you are not dividing by 0 ever. This should be a very minor edit, no part of your Newton iteration loop algorithm should change, you should only be ensuring x does not equal 0 before entering the loop

13. Copy and paste *Newton2.java* to create a new file, name the new file *Newton3.java*

14. Open *Newton3.java*

15. Edit the main method in *Newton3.java* so that it prompts the user to enter the value for the error (ϵ), rather than assuming it to be 0.0001. Only prompt the user once in the main method

16. Edit the `sqrt` static method in *Newton3.java* accordingly so that the user-inputted value for ϵ is passed to the `sqrt` function

17. Copy and paste *Newton3.java* to create a new file, name the new file *Newton4.java*

18. Open *Newton4.java*

19. Edit the `main` method of *Newton4.java* so that it no longer asks the user if they wish to compute another square root. Instead, the program asks the user to enter a new value for x and interprets a negative input as an indication to terminate the program

20. Create a zip file of your *Newton* project

21. Rename the zip file (not your project folder) using the naming scheme
    ”FirstName_LastName_DotNumber_ProjectNumber.zip”, for example mine would be
    ”Logan_Frank_580_2.zip”

22. Submit to Carmen