CS200: Programming I **Instructor:** Sam Vinitsky

P10: Definitely Not a Calculator

(40 points)

Due: Thursday, August 1 @ 11:59pm

Introduction

Brrrring brriiiiiing the phone rings – it's J. Pierrepont Finch, local businessman and vice president of advertising! Finch is a world-renowned businessman, but is harboring a terrible secret – he is terrible at basic math. If his rivals ever found this out, it would be the end of him. He has a very important math-related business meeting coming up next weekend, but is terrified that his lack of math skills will be on full display! In order to prevent this, he has commissioned *you* to help him!

Finch needs you to make a (secret) calculator for him. This program should be simple enough that someone who knows nothing about math should be able to use it. Finch has sent you a letter (given below) describing the program he desires, and how he would like to be able to use it. However, since Finch is not a programmer, he has not included *how* to make the program. This is up to you, and will require some creativity and problem solving on your part.

When making this program, you will need to use several built-in Java classes: Scanner, String, Double, Math, Random, etc... This will require you to import the classes properly! You can use any classes that seem relevant to the task at hand.

Before attempting this homework, work through the practice problems from Wednesday and Thursday of this week to get a handle on the built-in Java classes you will need to use for this assignment.

As usual, each program you turn in should include a comment at the top with (1) your full name, (2) your student ID number, (3) your netID, and (4) the name of anyone you discussed the homework with (excluding Sam and Alex).

As always: **start early, ask questions, and have fun!**

Finch's Letter

To whom it may concern,

As you are aware, I have requisitioned your help in creating an interactive calculator for my per-

sonal use. I have some very specific requests for how it works that you **must** adhere to, since I am a very important and powerful man. Make sure to read my instructions carefully, since I get angry very easily.

First and foremost, I wish to maintain my outwards appearance of being a very smart man. Please put the calculator in a file called `DefinitelyNotACalculator.java`. When I run this file, the program should **greet me with a pleasant message.** Then, it should **prompt me to type in an expression** on a new line. I will then type in an expression, which follows the rules attached to this letter (see addendum 1). Then, your program should print out the results of evaluating that expression. The program will then prompt me **again** for an expression. This process will repeat until I type in the word `quit` as the expression. When I do, the program should say goodbye kindly.

If at any point I enter an expression that is *invalid* for whatever reason, the program **must not crash**. It should instead print an error message describing the issue, and prompt me for another expression.

Notice that this program is **interactive**. It will need to prompt me for my input, and do something with that input. This means that each time I run the program, something *different* might happen, depending on which inputs I entered as the program was running.

I have also included an example session of what I desire my interaction with this program to look like (see addendum B). *Yours may differ slightly* in terms of the exact ways you greet, prompt, and say goodbye, but the general structure should be the same.

My secretary has included a list of hints for you in addendum 3.

Thank you very much for your hard work. If you do a sufficiently good job, I will make sure to put in a good word for you with my superiors.

Good day,
J. Pierrepont Finch

Addendum 1: List of Expressions

Listed here is the first addendum to Finch's letter: a table describing the various expressions his calculator must be able to evaluate. Each expression is a "command" followed by a list of arguments for that command, separated by spaces. For example, the expression `+ 4 5` should add `4` and `5`, resulting in `9` being returned.

Below is a list of the commands, along with their usage, and a what they evaluate to. Notice that here `x` and `y` are used to stand for any arbitrary numbers – either with or without decimal points. For example, the command "+" can be called as "`+ 4 5`", but NOT "`+ 5 y`" or "`+ x y`".

Note that these commands should work no matter how the word is capitalized. For example, `max`, `Max`, `MAX`, and `mAx` should all evaluate to the `max` expression. Unless otherwise specified, every command should work for both doubles AND integers.

Name	Usage	Evaluates To	Notes
`+`	`+ x y`	$x + y$	None
`-`	`- x y`	$x - y$	None
`*`	`* x y`	$x * y$	None
`/`	`/ x y`	x/y	This is *double* division
`max`	`max x y`	The bigger number of x and y	None
`pow`	`pow x y`	x^y	None
`abs`	`abs x`	The absolute value of x	None
`log`	`log x`	The log (base 10) of x	None
`sqrt`	`sqrt x`	The square root of x	None
`rand`	`rand x y`	A random **integer** between x and y	x and y must be `int`s
`rand`	`rand x`	A random **double** between 0 and x	x is a `double` or `int`
`sum`	`sum x1 x2 x3 x4 ... xn`	The sum of all the arguments	No limit on number of args
`sort`	`sort x1 x2 x3 x4 ... xn`	The arguments in sorted order	No limit on number of args

---

Addendum B: Example Session

Listed here is the second addendum to Finch's letter: an example session showing how he expects the program to behave. The red lines represent the input that Finch gave to the program as it was running.

```
Good day, Mr. Finch! Welcome to your totally-not-a-calculator.
Please input an expression:
+ 4.0 4.0
8.0
Please input an expression:
- 4.0 5.0
-1.0
Please input an expression:
/ 4 5
0.8
Please input an expression:
rAnD 0 5
4.0
Please input an expression:
raND 10
5.55555
Please input an expression:
```

```
SQRT 16
4.0
Please input an expression:
sqrt 16.0 13.0 14.5
Error! Incorrect number of arguments.
Please input an expression:
sort 4 3 2 1 5000
1 2 3 4 5000
Please input an expression:
sdfjklsjdfkl 123 123
Error! Command not found.
Please input an expression:
quit
Have a nice day, Mr. Finch!
```

Addendum 3: Minor Hints

Finch's secretary is a computer programmer, and has included a series of small hints. However, you may choose to ignore any of these, if you find a better way of making your program work the way Finch wants.

- You will need a *loop* of some kind to prompt the user repeatedly...
- You will need to use the `Scanner` class to get the user input. The **only** method you need is `.nextLine()`.
- You will need to use the `String` class to parse the input.
- There are `String` methods that convert entire strings into lowercase
- In order to break the user's input into it's parts, try `String`'s `split()` method, with a single space as input...
- You will need to use the `Double` class to convert the inputted number into `doubles` rather than `Strings`.
- At some point, you will need to figure out *which* command you are evaluating. Notice that some commands do different things depending on the number of arguments...
- You will need to use the `Math`, `Random`, and `Arrays` classes to actually evaluate the expressions.
- Be careful with how you handle bad expressions. Make sure to print out *meaningful* error messages.

- Make sure to print out the result.
- When comparing `Strings`, make sure to use "`.equals()`", **not** "`==`"!
- When using built-in Java classes, consult the documentation (found via Google). Make sure this is the Java 8 documentation!!!
- If you have any questions or issues, *please* ask on Piazza or during office hours.

Addendum 4: Pseudo-code

Finch's secretary also secretly included some pseudocode for the program. You do **not** need to follow this pseudocode, as long as your program functions the way Finch requested (but trust me, it'll be easier if you follow this.) The pseudocode is as follows:[1]

1. Greet Mr. Finch
2. Prompt the user for input
3. Until "`quit`" is the input
   (a) Parse the input into (1) a command name and (2) the arguments
   (b) If the command name is not a valid, print an error
   (c) Determine *which* command is being used
   (d) If the number of arguments is wrong, print an error
   (e) Otherwise, parse each argument as a double or integer, as necessary[2]
   (f) Based on the command, do something with the arguments to produce a result
   (g) Print out the result
   (h) Prompt the user for input again
4. Say goodbye to Mr. Finch

[1] This can be accomplished much more cleanly with a "`do-while`" loop, which we will not cover in this course. You are welcome to use one if you desire! You can read about them here

[2] If the user inputs something *other* than a number, such as a letter, your program will crash when you try to cast it to a `double`/`int`, and throw an exception. If you desire, you can try to catch this using a `try`/`catch` block, which we will be covered in CS 300. See here for a detailed description. This is entirely optional.

Part ∞: Feedback Form [3 points]

Fill out this feedback form **after you submit** this assignment. Completion of this will count towards your grade, but your responses themselves will not affect your grade in any way (so be honest!).

What to turn in

On Canvas, turn in a zip folder called `<my_net_id>_P10.zip` with the following programs:

- `DefinitelyNotACalculator.java`

Also complete the feedback form. [3 points]
