This introductory assignment is intended to be done by you as a solo effort. The goal of the assignments is first, to review basic OO terms and second, to get set up to perform basic development tasks with Java and Git.

Project 1.1: OO Concepts and Definitions – 10 points

Provide definitions for each of the following six terms. Remember to cite sources, even if the source is the textbook. You may want to look for web sources, open courseware, etc. OO books (including the textbook) can be found via search on the CU Libraries link, or in the O'Reilly-Safari e-books, under the Sciences tab here: https://libguides.colorado.edu/strategies/ebooks. Please note that Wikipedia is not usable as a primary source, neither (in this case) are the class slides.

- abstraction
- encapsulation
- polymorphism
- coupling
- cohesion
- identity

For each term above provide:

- the definition of each term (supported by citation)
- an example of the term applied in the design of a class or a set of classes – this can be shown as code, psuedo-code, text/description, or graphic/UML examples).

Capture your answers in a PDF with your name for submission to Project 1.1

Project 1.2: Java exercise – 15 points

These coding exercises are designed to let you familiarize yourself with basic Java and ensure your GitHub and Java development environments are set up appropriately. You should use a Java 8 or later environment to develop this code.

Program 1: Write a basic numerical analysis program that will perform as follows:

- Enters a loop asking for numeric input from the user (should accept any real or integer value)
    - Each number entered should be appended to an ArrayList, and the current ArrayList should be printed to the console
    - When a user enters a null input (i.e. hits return with no number) the input loop should end
- At that point, apply a number these descriptive statistics methods against the ArrayList, including:
    - median
    - mean
    - mode
    - variance
    - standard deviation
    - minimum value
    - maximum value
    - maximum occurrences – value and count (if the number 2 was in the list 3 times, and other numbers were not duplicated more than 3 times) the answer is value 2, count 3.

- The results of each of the operations should be printed to the console, ending the interaction

The calculations of these values **may not** use a math or statistics library (i.e. math.stddev(list)), you must write the code to perform each of the operations to the provided ArrayList of numbers. There should be at least two classes (Reader, Analyzer) that will be instantiated to perform the work. Each of the calculations (sum, median, mean, etc.) should be methods inside the Analyzer class. A method called analyze in the Analyzer class should perform the eight operations listed above.

Capture the output console text from the data entry and calculations (cut and paste to a text file is fine) from at least two test runs – one with a set of integers, one with a set of real numbers – the results file should be named Results1.txt.

Program 2: Create a Java program that loops through the following methods:

- Read - reads a string from the console input
- Clean - converts all letters to upper case, discards any spaces
- Sort - sorts the letters in the string into alphabetic order
- Palindrome - creates a palindrome from the letters
- Print – prints a string to the console output

Use an instance of a class called Processor with three methods: read, clean, sort, palindrome to perform each operation. An example of input and corresponding output after all steps would be “Bruce” and “URECBCERU”. The program will end if a null string is submitted as input. Run the program using both “Bruce” and your full name as input and capture the output to a Results2.txt text file (cut and paste of output from the console into the file is fine).

Your Java should be well commented and easily readable. **If you take code from external sources, include a comment with the URL of the source.**

Your Java main functions should be very short and do very little – the work should be done in the objects instantiated from the classes. For Program 1, for instance, your main Java program should be limited to something like:

```
Reader reader = new Reader();           // instantiate the objects
Analyzer analyzer = new Analyzer();
ArrayList data = reader.getData();      // read data from the user
analyzer.analyze(data);                 // perform all calculations
```

Grading Rubric:

Project 1.1 is worth 10 points total and is due on Wednesday 8/31 at 8 PM.

The submission for 1.1 will be your PDF with each of the six terms, their definitions, and your examples.

Questions for 1.1 will be graded on the quality (not quantity) of the answer. Remember your answers require definitions, examples, and citations (in any format). A solid answer will get full points, missing elements or poor-quality answers will cost -1 point each, missing answers completely will be -2 points.

Project 1.2 is worth 15 points total and is due on Wednesday 9/7 at 8 PM

The submission for 1.2 will be a URL for a private GitHub repository. The repository should contain the Java code for both programs, a captured text file with results for each program (use the names Results1.txt and Results2.txt), and a README markdown file.

The README file should include (at a minimum) your name, the Java version you're using, and any other comments on your work – it should be easily located in the GitHub repo. Incomplete/missing READMEs will be penalized -1 to -2 points.

Code for Part 2 should be commented appropriately, including citations (URLs) of any code taken from external sources. A penalty of -1 to -2 will be applied per exercise for poor or missing comments.

Each program should include evidence of output from a run captured in the text file mentioned per exercise. A penalty of -1 to -2 will be applied per exercise for incomplete or missing output.

Please ensure all class staff are added as project collaborators to allow access to your private GitHub repository. Do not use public repositories.

GitHub Repo IDs:

Bruce	brmcu
Dwight	dbrownecu
Gayathri	gayathripadmanabhan1
Max	wmaxdonovan-cu
Roshan	roshannp
Vardhini	vaba6239

Overall Project Guidelines

Assignments will be accepted late for four days. There is no late penalty within 4 hours of the due date/time. In the next 48 hours, the penalty for a late submission is 5%. In the next 48 hours, the late penalty increases to 15% of the grade. After this point, assignments will not be accepted.

Use office hours, e-mail, or Piazza to reach the class staff regarding homework/project questions, or if you have issues in completing the assignment for any reason.