**Data Set:**

The dataset used in this project is the Goodreads dataset, which was uploaded to Kaggle by a user named Soumik. The Goodreads dataset was created by Soumik because he wanted to provide people with a clean dataset of books and their reviews. I choose this dataset because I liked how it provided important information about books in a simple format. The original dataset includes 10 fields, I have shortened the dataset to 6 fields by deleting columns from the CSV file. The entries are ordered starting at bookID number 1 and go in ascending order. Each field in my program will display an error message if the dataset is not being read in correctly.

**The fields:**

* bookID: Integer that lists all the books in the dataset. Lists all the rows in the file as well.
* title: A string that displays the books title.
* Authors: A string that displays the author or authors names.
* average\_rating: String that displays the average rating of a book.
* Isbn: String displaying a unique number to identify a book.
* isbn13: String displaying another unique number to identify a book.

**Analysis:**

* Graph the number of reads and writes for each sorting algorithm.
* Compare and contrast the different sorting algorithms and draw conclusions about which is more efficient.
* Discuss complexities and their effect.

**Which sorting algorithm:**

* If you need to sort a contacts list on a mobile app, which sorting algorithm would you use and why?
* If you need to sort a database of 20 million client files that are stored in a datacenter in the cloud, which sorting algorithm would you use and why?

**Extra Credit:**

* Performed the same experiment, except doubled the size of the data set each time.