Analysis of Algorithms
CS 375, Fall 2022
Project 1
Due **BY 11:59pm** on Wednesday, September 28

Project 1: Determining Algorithmic Time Complexity

In this assignment, you'll work in teams of 2 or 3 to determine the asymptotic time complexities of methods in two Java Classes. In Part 1, you will be given source code for the class, and you will figure out asymptotic complexities by analyzing that code. In Part 2, you will *not* be given the source code for the Class—you will instead conjecture about time complexity by using definitions of asymptotic complexity and analyzing the number of comparisons the methods make during execution.

The goals of this project are:

- to give you practice determining the worst-case time complexity of an algorithm given its source code;
- to give you practice using the scientific method and the formal definitions of asymptotic complexity to guess at the worst-case time complexity of algorithms for which you don't have the source code; and
- to give you practice working with other students in a team.

Exercises

1. For **Part 1**, download the source code for the `ArrayIntegerSet` class from the course Projects and Presentations website.

   For each constructor and method in the `ArrayIntegerSet` class, determine its worst-case time complexity (using Θ notation) in terms of the number of elements n in the set and, if appropriate, any other data, such as the size m of any parameters. For each method, please add something like the following in its comment header:

   `// Time complexity:` $\Theta(n)$`, where` n `is the size of this ArrayIntegerSet`

   For every method, be sure to unambiguously state what each variable in your time complexity expression refers to. For example, if two relevant variables refer to arrays, be very clear which variable refers to which array. (For example, please be more specific than saying "the input array," as from some perspective, any relevant parameter could be viewed as input!)

   Along with each asymptotic complexity bound, include a short (1–2 sentences at most!) explanation of how you came up with your complexity bound. If your answer depends upon the complexity of other methods in the Class, note that as well.

   Be sure to give the complexity of all methods (except for `main()`), whether public or private.

2. For **Part 2**, download the sortData.csv file from the course Projects and Presentations website. This file contains the data obtained by running 4 sorting algorithms and counting how many comparisons they make during execution. (The sorting algorithms themselves will not be made available to you.) The data in the file was gathered as follows: For each desired array size, a random array A of that size was created; then, each of the sorting algorithms was run on array A. (To be clear: All algorithms were run on the same array A.)

   Based on this data, make a *useful* conjecture on the *worst-case* asymptotic time complexity of each of the 4 algorithms, to show a practical understanding of each algorithm's performance. Then write a report containing these conjectures and explanations supporting them. For each, be sure to explicitly employ the relevant **formal definition(s) of asymptotic complexity** used in your conjectures, and include the mathematical details about how you apply them to the data. For example, if using big-O notation, give the particular values of threshold input size n_0 and leading constant c used to support your conjecture, and show how they are applied to the data to lead to your conclusion; if using Θ notation, give the n_0 value and both leading constants, c_1 and c_2, and similarly show how they are applied. Full credit may not be given for conjectures that are not consistent with the known asymptotic time complexity performance of these sorting algorithms, so please be careful and detailed in your analysis!

   Please be sure that your explanations discuss your reasons for supporting your conjectures (or being skeptical of them, if that's relevant); because the formal definitions of asymptotic complexity are essential for analyzing your conjectures, please include all details necessary to show command of the relevant asymptotic complexity definitions and how they are employed. For example, simply employing a statistical curve-fitting method would not be sufficient for these exercises; additional interrogation of the data, directly relating to definitions of asymptotic complexity, is required.

   Please keep the following notes in mind:

   - Please treat this part of the project like a physics or chemistry project in which you are creating a report, making conjectures based on the results of an experiment. Therefore, a significant part of evaluating your work will be the consideration of whether or not the data are sufficiently analyzed or if the conclusions are logical. In other words, don't just say "Here's our conjecture: ..."; please also include your analysis of the data, such as describing how you came up with your conjecture instead of a different conjecture.
   - As usual in CS375 assignments, clear communication *is* the point: There will be deductions if it is difficult for your grader to understand your process and conclusion. Please treat your written report as being as important as your conclusions.
   - Also as usual for CS375 assignments, please feel free to talk with me if you have questions, especially if you are unsure what to include in your report!

What to Submit

Each group should submit just one set of answers. In particular, each team should submit two files:

- For Part 1, turn in a text file that includes only the source code that you analyzed and the time complexity information about all the methods, as described in Exercise 1 above. The time complexity information should be included in the comment header for each function.

  To indicate the team submitting that file, please be sure to name your file

  `team_<INITIALS>_ArrayIntegerSet.java`

  where `<INITIALS>` is replaced by the initials of the team members in the group in the team assignments. E.g., if Eric Aaron and Stephanie Taylor were the teammates, the file from that team would be called `team_EA_ST_ArrayIntegerSet.java`.

- For Part 2, turn in a `project1_report` PDF file that has your report, but in the filename, as above, please add the prefix indicating your team members' initials, e.g., `team_EA_ST_project1_report.pdf`

Lateness policy: To keep pace with the project assignments in CS375, it is important that this assignment be turned in promptly. For this project, there will be a deduction of 1.5% for each day late—i.e., 1.5% deduction for submitting up to 24 hours late; 3.0% deduction for submitting more than 24 hours late, up to 48 hours; etc—up to a 10% deduction for submitting up to 7 days (168 hours) late. After 7 days, late submissions will receive a one-third ($33.3\overline{3}\%$) deduction. Please submit your work promptly!

As always, extenuating circumstances will be considered—please contact me as soon as possible if any extenuating circumstances are impeding your work on this project!

Submission instructions: For this project, each group should have one "designated submitter" to handle submission of their materials. The designated submitter should **email me** (eaaron@colby.edu) the two files.