

Analysis of Algorithms
CS 375, Spring 2020
Project 1
Due **BY 11:59PM** Monday, March 16

Project 1: Determining Algorithmic Time Complexity

In this assignment, you'll work in teams (picked randomly by your Prof.—see the course website) to determine the asymptotic time complexities of methods in two 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 estimate 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 experimental methodology and the formal definitions of asymptotic complexity to empirically assess the worst-case time complexity of algorithms for which you don't have the source code;
- to give you practice presenting the results of your methods and analysis in a written report; 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. You do not need to analyze the `main()` itself method, but for every other 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 analyzed, 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!)

You need not explain how you came up with your complexity formula; in particular, you do not need to say what you used as the primitive operation (*barometer operation*) that you counted. Be sure to give the complexity of all methods (other than `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 3 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 3 sorting algorithms was run on array A . (To be clear: All 3 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 5 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. Please be sure that your explanations discuss your reasons for supporting your conjectures (or being skeptical of them, if that's relevant), and present the important factors that might affect the validity of your conjectures; 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.

Because you are reporting on conjectures based on experimental results in this part of the project, similar to a report for a science project (e.g., a physics or chemistry report), part of evaluating your work will be the consideration of whether or not data are sufficiently analyzed, conclusions are logical, and methods and analysis are effectively presented in writing. Please be sure to say more than just something like "Here's our conjecture: ..."; please also include a thorough analysis of the data and your process, such as describing how you came up with your conjecture instead of a different conjecture.

Generally good ideas for presenting reports are applicable here! Please keep the following notes in mind:

- An effective scientific report enables readers to fully examine not just the data but also the methods used and the conclusions drawn from them. Please include all data and methods used for your analyses!
- As in other Colby CS courses—and throughout the sciences—your report should have a brief introductory section, giving an overview of the findings presented in the full report.
- Please use section headings to separate sections of the report. In this case, you'll be analyzing multiple algorithms; in your report, present the data and analysis for each different algorithm in its own separate section.

- Please make sure all figures presented are accompanied by descriptive text, such as text on the figure (a *legend*), descriptive labels on axes, and some additional English description of the data presented, either in a caption or in the report’s main text (or both).

As usual in CS375 assignments, clear communication *is* the point: Your work on this project will be partly evaluated on the effectiveness of the presentation in your report, so 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 of all the methods. The time complexity 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_<LETTER>_ArrayIntegerSet.java`

where <LETTER> is replaced by the letter corresponding to the group in the team assignments. E.g., the file from *team a* would be called `team_a_ArrayIntegerSet.java`.

- For Part 2, turn in a `project1_report` PDF file that has your report, but as above, name the file to lead with your team name, e.g., `team_a_project1_report.pdf`

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.

In addition, each student (not just the designated submitter) is required to individually email me within a day after the project was turned in, telling me:

- how well your group worked together;
- how much (percentage) each member of your group contributed to the project; and
- how many hours you (individually) spent on this project.

If you would prefer, you are welcome to see me (again, within a day after the project was turned in) to give this information and discuss your group’s dynamics and your work as part of the group. And as always, please feel free to talk with me at any time about questions regarding any aspect of this project—I’ll be very happy to talk with you!