

CS375 Project 1

Due Feb 25th 10PM ET

A general note for CS375: When writing things up, please write neatly and explain your answers clearly, giving all details needed to make your answers easy to understand. Graders may not award full credit to incomplete or illegible solutions.

As a reminder, projects can be submitted in small groups, optimally with 3 people. If you feel exceptionally strongly about having more than 4 people come see me.

Clear communication **is the point**, on every assignment. In general in CS375, unless explicitly specified otherwise, answers should be accompanied by explanations. Answers without explanations may not receive full credit. Please feel free to ask me any questions about explanations that might come up!

Necessary acknowledgements: many of these questions are reproductions or adaptations of questions of Eric Aaron's devising.

1. Download the source code for the `ArrayIntegerSet` class from this [page](#). For each constructor and method in the `ArrayIntegerSet` class, determine its worstcase 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. (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 (12 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. On this [page](#) you will find 8 different problems: pick 1 from each section (for a total of 3 problems) and
 - i) write the pseudocode for a brute force algorithm to solve the problem. While detailed correctness arguments are unnecessary, I'm asking for a short English description of what the algorithm does along with a detailed analysis of the worst case runtime.
 - ii) describe a simple **reasonable** greedy algorithm for the problem and show that it doesn't work. Note that this is asking for two things: a simple algorithm and a (preferably simple) instance upon which the algorithm fails to produce the correct answer.
3. Lastly, pick one of your problems that you chose in the previous question and record a video demo of at most 15 minutes in length. In later projects this demo will have tighter time

restrictions, but I'm largely thinking of this as a training run for your later demos - I'm **expecting** to give a lot of feedback for this demo regarding what I would expect to see improve for future projects. This demo should include:

- A description of the problem, along with an example input and output.
- A presentation of your algorithm from 2.ii that fails, along with a trace-through of the algorithm on that input.
- A presentation of your brute-force algorithm from 2.i, along with a thorough description of its worst case runtime.

The point of this demo is to exercise your ability to orally communicate about both problem statements and algorithms. I expect that your demo will contain thoughtful diagrams / figures as necessary (particularly for the example input/output and your trace-through). While I would probably recommend presenting the material in the order I wrote above, it's not a requirement and if you wanted to talk about the brute force first I can imagine plenty of presentations that could do that successfully. I would suggest that your target audience for this demo be any fellow student in CS375, which means in particular that you can assume that they have seen the pseudocode and understand the runtime of `GENERATEALLSUBSETS` and `GENERATEALLPERMUTATIONS`. **Please note that all group members are required to split the amount of time speaking reasonably evenly.**