

CS 630, Fall 2024, Homework 3

Due Wednesday, October 2, 2024, 11:59 pm EST, via Gradescope

Homework Guidelines

Collaboration policy Collaboration on homework problems, with the exception of programming assignments and reading quizzes, is permitted, but not encouraged. If you choose to collaborate on some problems, you are allowed to discuss each problem with at most 5 other students currently enrolled in the class. Before working with others on a problem, you should think about it yourself for at least 45 minutes. Finding answers to problems on the Web or from other outside sources (including generative AI tools or anyone not enrolled in the class) is strictly forbidden.

You must write up each problem solution by yourself without assistance, even if you collaborate with others to solve the problem. You must also identify your collaborators. If you did not work with anyone, you should write "Collaborators: none." It is a violation of this policy to submit a problem solution that

Typesetting Solutions should be typeset using any program you like. We recommend using LaTeX writing (overleaf.com) since it handles math very well. Word, Google Docs, etc. are not recommended.

Solution guidelines

1. pseudocode and, if applicable, pseudocode should be clear and concise.

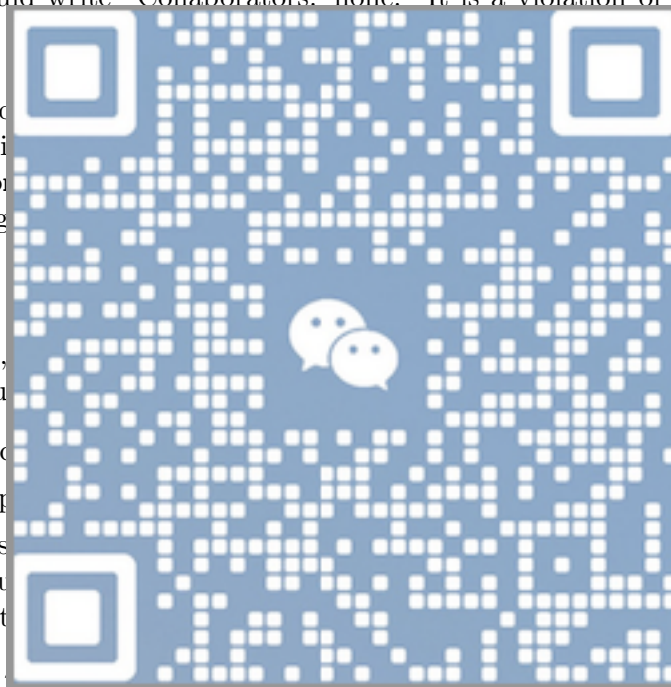
- A clear description of the algorithm.
- Any assumptions made.
- Instructions for how to implement the algorithm yet without any subroutines.

If the algorithm is not clear, it may not be graded.

2. a proof of correctness
3. an analysis of running time and space

You may use algorithms from class as subroutines. You may also use facts that we proved in class.

You should be as clear and concise as possible in your write-up of solutions. A simple, direct analysis is worth more points than a convoluted one, both because it is simpler and less prone to error and because it is easier to read and understand.



Gradescope. You may only use Gradescope for technical writing since it handles math very well.

When you submit a solution, you must provide: a clear description of the algorithm in English. As always,

(for example, you should not write "I thought about the problem and then I wrote the solution." Inputs and outputs of the algorithm should be specified, etc.)

Problem 1 *Polling (10 points)*

As the elections are coming near new polls are published every day. As part of a research group at BU you are conducting experiments on how belonging to a social group influences ones' vote. For this you will select some known groups and survey every one of its members. To get unbiased results you decide that none of the groups you select can share any members. From the Registrar at BU you get a list of all sanctioned groups within BU along with their member lists. Design an approximation algorithm to select the largest number of groups.

1. Design an approximation algorithm for this problem. (*For proof you have to show that the output of the algorithm indeed yields a valid solution for the problem. Don't forget to analyze the running time.*)
2. Clearly state and prove the approximation ratio of your algorithm. (*Any approximation ratio correlated with 0 will not get full credit.*)

Problem 2 *Catering*

You are a caterer and have a list of entrees and drinks (fancy steaks and wine). You also have a sommelier. You also have a list of acceptable pairings from your sommelier. You also have a list of acceptable pairings (list of acceptable choices), but the preferences of the guests (list of acceptable pairs). (For some reason you

1. Design an approximation algorithm for this problem. (*For proof you have to show that the output of the algorithm indeed yields a valid solution for the problem. Don't forget to analyze the running time.*)
2. Clearly state and prove the approximation ratio of your algorithm.
3. Now suppose that the guests can mix-and-match; they may take an entree from one pair with a drink from another pair as long as both are on their preference lists. Design an approximation algorithm to cater this scenario with the fewest possible number of pairs selected. (*Prove that your algorithm indeed yields a valid solution, but you can omit the running time analysis.*)
4. Clearly state and prove the approximation ratio of your second algorithm.