Analysis of Algorithms
CS 375, Fall 2022
Problem Set 5
Due **by 11:59pm** on **Friday, December 9**

- For this assignment, standard file naming conventions apply: Please submit your typewritten answers in a PDF file named `CS375_PS5_<userid>.pdf` where `<userid>` is replaced by your full Colby userid, and submit it to your `SubmittedWork` folder. Please reach out to me right away with any questions or concerns about this!

- *A general note for CS375:* When writing up your homework, please present your answers 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 hard to understand solutions. 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!

- **Important Reminder:** As stated on our syllabus / first-day handout, all work for CS375 Problem Sets and Smaller Assignments must be submitted by the end of the last day of classes—that's **11:59pm on Friday, December 9**. Even late work must be turned in by then to have the grade dropped when final grades are calculated. Please turn in this PS, and all other work (other than our last Project) by that deadline!

  As always, extenuating circumstances will be considered. Please talk with me as soon as possible if there are extenuating circumstances or you believe you cannot meet that deadline!

Exercises

1. **A Rook-y Move!** In chess, a rook can move horizontally or vertically to any square in the same row or in the same column of a chessboard. Find the *number of shortest paths* by which a rook can move *from the bottom-left corner of a chessboard to the top-right corner*. (The length of a path is measured by the number of squares it passes through, including the first and the last squares.)

   Solve the problem by a dynamic programming method. That is, come up with a relevant recurrence—i.e., a recursive definition of the relevant values for a solution, as we've seen in the examples from class—and using dynamic programming techniques, calculate the solution. Please show your work by giving an 8×8 table (each element in the table represents the corresponding square on the chessboard), where each entry in the table should be the number of paths from the bottom-left corner to that square on the chessboard; therefore, the answer to this exercise will be the number in the top-right element of your table.

   Note that you do not need to provide a shortest path from one corner to the other, just the number of shortest such paths. Please be sure to include an explanation of the correctness of your recurrences (a well-worded paragraph could suffice); you do not need to explain how you used the generated the table from the recurrence.

2. **Bridge Crossing Revisited!** Consider the generalization of the bridge crossing exercise earlier in the semester (PS0, exercise 5) in which there are $n > 1$ people whose bridge crossing times are $t_1, t_2, \ldots t_n$. All the other conditions of the problem are the same as before: at most two people at a time can cross the bridge (and they move with the speed of the slower of the two) and they must carry with them the only flashlight the group has.

   (a) Design a greedy method for getting the entire group across the bridge, with the goal of minimizing the total time for the group to cross. Give a very brief explanation of what makes it a greedy method.

   (b) Show that the greedy method does not always yield an optimal solution (i.e., a minimal crossing time) for every instance of this problem by giving a concrete counterexample **with the smallest number of people** for which it is not optimal.