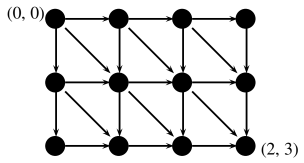
CSCD320 Homework6, Winter 2024, Eastern Washington University, Spokane, Washington.

Name: **EWU ID:**

Please follow these rules strictly:

1. Write your name and EWUID on **EVERY** page of your submission.
2. Verbal discussions with classmates are encouraged, but each student must independently write his/her own solutions, without referring to anybody else's solution.
3. The deadline is sharp. Late submissions will **NOT** be accepted (it is set on the Canvas system). Send in whatever you have by the deadline.
4. Submission must be computer typeset in the **PDF** format and sent to the Canvas system. I encourage you all to use the LaTeX system for the typesetting, as what I am doing for this homework as well as the class slides. LaTeX is a free software used by publishers for professional typesetting and are also used by nearly all the computer science and math professionals for paper writing.
5. Your submission PDF file must be named as: **firstname_lastname_EWUID_cscd320_hw5.pdf**
   (1) We use the underline '_' not the dash '-'.
   (2) All letters are in the lower case including your name and the filename's extend.
   (3) If you have middle name(s), you don't have to put them into the submission's filename.
6. Sharing any content of this homework and its keys in any way with anyone who is not in this class of this quarter is NOT permitted.

Problem 1 (70 points)**.** *Consider a directed graph arranged into rows and columns. Each vertex is labeled (j, k), where row j lies between 0 and M, and column k lies between 0 and N. The start vertex is $(0, 0)$ and the destination vertex is (M, N). Each vertex (j, k) has an associated weight $W(j, k)$. There are edges from each vertex (j, k) to at most three other vertices: $(j + 1, k)$, $(j, k + 1)$, and $(j + 1, k + 1)$, provided that these other vertices exist.*



An example graph with $M = 2$ and $N = 3$.

The input of your algorithm is a 2d array $W[0 \ldots M \times 0 \ldots N]$, where $W[j, k]$ stores the weight of the node at coordinates (j, k).

The output of your algorithm is the number of the lightest path from $(0, 0)$ to (M, N). The lightest path is one that has the minimum weight. The weight of a path is defined as the total weights of all the nodes on that path.

For example, say the given array W for the above example picture is:

Name: **EWU ID:**

5	*3*	*8*	*2*
7	*2*	*1*	*9*
8	*2*	*1*	*5*

The lightest path from $(0,0)$ *to* $(2,3)$ *is:* $(0,0) \rightarrow (1,1) \rightarrow (1,2) \rightarrow (2,3)$ *or* $(0,0) \rightarrow (1,1) \rightarrow (2,2) \rightarrow (2,3)$. *The weight of either one of the lightest paths is* $5+2+1+5=13$. *Therefore, the number of the lightest path from* $(0,0)$ *to* $(2,3)$ *is* 2.

1. *Provide a recursive formula that computes the exact value that is specified in the problem. Check the book chapter on dynamic programming for an example of such recursive formular, which essentially catches the understanding of the recursive structure in the solution.*

2. *Express the recursive formula as a bottom-up and iteration-based dynamic programming algorithm, and determine its running time in big-oh notation. Make your bound as tight as possible.*

Problem 2 (20 points)**.** *Search and learn three existing algorithms that use the dynamic programming strategy. You cannot use those that we have discussed the class ("rod-cutting" and "matrix-chain"). For each problem and its algorithm solution, in your own language, concisely and clearly describe:*

1. *The problem statement*

2. *The recursive structure in the solution.*

3. *Why does the recursion-based idea not work well ?*

4. *Why does dynamic programming work ?*

5. *The source of your finding. For example, the url of the webpages, the title and page of a book, the title/author/year of an article, etc.*

Note: If you copy and paste with minor/trivial change in the language without your own understanding, you will get zero for this problem.