Due: Tuesday, February 21, 2023, 11:59pm

Submit online at (clickable link): `https://forms.gle/RA9RVB7hEX8CSzNs5`

1   0-1 Knapsack Table

Fill in the following dynamic programming table for the 0-1 Knapsack Problem. Use the subproblem definition and recursion show during lecture:

OPT_KS(i, C) = the highest possible total value obtained by choosing items from item $\#i$ through item $\#n$ such that the weight of the chosen items does not exceed C.

OPT_KS$(i, C) = \max\{$ OPT_KS$(\ i+1,\ C-w_i\ ) + v_i\ ,$ OPT_KS$(\ i+1,\ C\ )\ \}$

$$C = \text{capacity}$$

	w_i	v_i
1	1	2
2	4	6
3	3	5
4	1	3
5	2	4

Item $\#i$

	0	1	2	3	4	5	6	7	8
1									
2									
3									
4									
5									

2   History Final

You are taking a bring-your-own-notes final exam for a history class where the professor has allowed you to bring r index cards, each of which holds k lines of text. You have prepared n short summaries of events you have studied in this class. For example,

> July 11, 1804: Weehawken, dawn. Guns drawn. A.Ham + A.Burr. (Everything was legal in New Jersey.)

Some of your summaries are longer than others. Let ℓ_i denote the number of lines taken up by the i-th summary when written on an index card. Unfortunately you cannot fit all n summaries on the r index cards, so you have to pick and choose. There are some constraints:

- Each index card must have k or fewer lines of text.

- Every summary written on an index card must fit entirely on that index card. (I.e., summaries cannot start on one card and end on another card.)

- The summaries must be in chronological order. The summaries on each index card are sorted by date, and all of the summaries on card i predate the summaries on card $i + 1$.

Your objective is to include as many summaries as possible on the r index cards allotted to you.

Questions:

1. Define a function, MAXSUMM, that can be used to solve this dynamic programming problem. To do this, you must describe the input parameters to MAXSUMM and its "return value" using *English sentences*. (Note: you are specifying the input-output relation of the function. You should *not* describe how to compute the function.)

2. Give a mathematical formula that shows how MAXSUMM can be computed recursively. Then, explain all the major parts of the formula using English sentences. Remember to include the base cases.

3. Describe how MAXSUMM can be computed bottom up using a dynamic programming table. Be sure to include a description of the dimensions of the table and the order that the entries of the table are to be filled. Draw a diagram. Which entry has the solution to the original problem?

4. State and briefly justify the running time of your dynamic programming algorithm. Report your running time in terms of n, k and r.

3   Return to Oz

Upon returning to Kansas, Dorothy complains that she didn't get to see much of the Land of Oz. She just followed the Yellow Brick Road and missed all the lovely sights off that well-worn path. She makes a list of n destinations she would like to visit and asks you to take her. Why? Because you have a blimp (which is much better than a hot-air balloon).

So, here is the plan. You approximate Oz as a two-dimensional plane. Each of Dorothy's n destinations can be specified by its two-dimensional coordinates (x_i, y_i). Although, the blimp is powered and steerable, it is best to drift along with the prevailing winds. From the Munchkins, you find out that in the next month or so, the winds will blow east to west. After that the winds will switch directions and blow west to east. So, you can start in Munchkin Land on the eastern end of Oz and travel west visiting some of the n destinations. You'll end the first portion of your journey at the Wicked Witch's castle and hang out with the flying monkeys for a bit. After the winds start blowing eastward, you will travel east and visit the remaining destinations (the ones you didn't visit during your westward flights). You end your trip at Dorothy's house which had landed on the eastern edge of the map.

That's a good plan, but you are worried about the available fuel for your blimp and also your carbon foot print. (Let's keep Oz green!) Thus, you want to plan this trip so the total distance traveled is minimized. Here are some constraints:
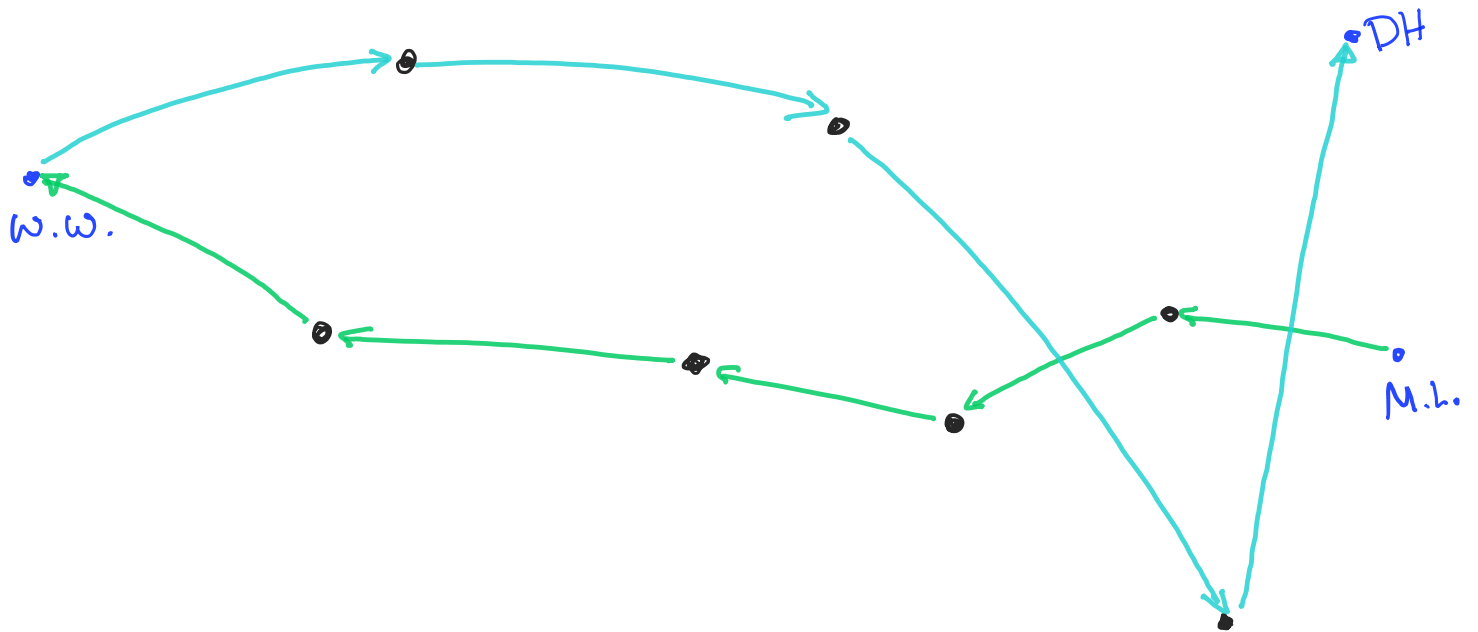
- Both Munchkin Land and Dorothy's house are on the eastern edge of Oz. There are no destinations east of these points. Let's say Munchkin Land is at (x_1, y_1) and Dorothy's house is at (x_2, y_2) and x_1 is a teeny bit larger than x_2.

- The distance between two destinations is just the Euclidean distance between their coordinates.

- During the first part of your journey, you must fly westward. The destinations that you visit must have decreasing x coordinate. (Let's just assume that none of the destinations have the same x coordinate.)

- Similarly, during the second part of your journey, you must fly eastward and visit destinations with increasing x coordinate.

- The Wicked Witch's castle is the most western destination and has the smallest x coordinate. For convenience, let's just assume that the destinations are sorted by decreasing x coordinate. That means the Wicked Witch's castle is located at (x_n, y_n).

This all seems rather complicated, but when you meet up with the Scarecrow and he says, "You know, ever since the Wizard gave me that diploma, I've found dynamic programming to be much easier — even though I don't really have a brain!"

Questions:

1. Define a function, MinDist, that can be used to solve this dynamic programming problem. To do this, you must describe the input parameters to MinDist and its "return value" using English sentences. (Consider figures on the next page.) *Note:* describe the value that MinDist must return in relation to its parameters, *not* how to compute it, that's the next question.

2. Give a mathematical formula that shows how MinDist can be computed recursively. Then, explain the main parts of the formula in English. Remember to include the base cases.

3. Describe how MinDist can be computed bottom up using a dynamic programming table. Be sure to include a description of the dimensions of the table and the order that the entries of the table are to be filled. Draw a diagram. Which entry has the solution to the original problem?

4. State and briefly justify the running time of your dynamic programming algorithm.

OPTION 1:

OPTION 2: