Due 12th November 2022 at 11:59pm Sydney time

Your solutions must be typed, machine readable PDF files. *All submissions will be checked for plagiarism!*

For each question requiring you to design an algorithm, you *must* justify the correctness of your algorithm. If a time bound is specified in the question, you also *must* argue that your algorithm meets this time bound.

Partial credit will be awarded for progress towards a solution.

Please note that for all Dynamic Programming solutions we expect you to clearly state a subproblem, a recurrence relationship, base cases, and how the final answer can be obtained. These element should all be related and in sync with each other! To justify the correctness of a Dynamic Programming solution, you need to explain (in words) how your recursion handles all necessary cases in the problem and why each case is correct. Roughly 10% of the total marks will be specifically allocated towards the clarity and conciseness of your explanations.

Question 1 *Bridge Support*

[**20 marks**] The NSW government has decided to build a new bridge. The bridge is to be n meters long, where $A[i]$ is the maximum load on the bridge between $(i-1)$ meters and i meters.

The government have contracted you to complete this project, and you've decided to build the bridge in spans of integer length. The cost of building a span is given by the length of the span squared, plus the maximum load that the span needs to withstand. The total cost of building the bridge is the sum of the cost of all spans. More explicitly:

$$\texttt{total\_cost} = \sum_{s \in \text{spans}} (s_R - s_L)^2 + \max(A[(s_L+1)..s_R])$$

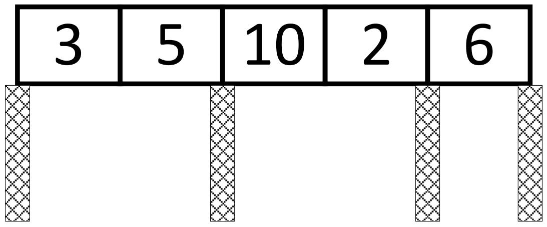
where s_L and s_R are the left and right endpoints respectively of the span s.

Your goal is to determine the **minimum** cost required to build a bridge that supports the load.

An example for the cost calculation: if $n = 5$ and $A = [3, 5, 10, 2, 6]$, and you decided to put pillars in at 2 meters and 4 meters, the total cost would be:

$\texttt{span\_[0,2]} = (2-0)^2 + \max(A[1..2]) = \$9$
$\texttt{span\_[2,4]} = (4-2)^2 + \max(A[3..4]) = \$14$
$\texttt{span\_[4,5]} = (5-4)^2 + \max(A[5..5]) = \$7$
$\texttt{total\_cost} = \$9 + \$14 + \$7 = \$30$

A depiction of this example is shown in image below.



1.1 [**14 marks**] Design an $O(n^3)$ algorithm which achieves your goal.

1.2 [**6 marks**] With an election looming, the NSW government has asked you to hurry up. Design an algorithm that achieves the goal in $O(n^2)$.

You may choose to skip Question 1.1, in which case your solution to Question 1.2 will also be submitted for Question 1.1.

Question 2 *Mountain Radios*

[**30 marks**] Alice and Bob are bored of the park, and have decided to go mountaineering! They will each be traversing the mountain and moving through a sequence of n camps. Both Alice and Bob will always spend the night at one of their camps. Alice's camps are given by a sequence of coordinates pairs $A_1, ..., A_n$, where $A_i = (A_{i_x}, A_{i_y})$, and Bob's camps are given as $B_1, ..., B_n$, where $B_j = (B_{j_x}, B_{j_y})$. On the first night, Alice will spend the night at camp A_1 and Bob will spend the night at camp B_1. Each day, both Alice and Bob have the choice of staying at their current camp or moving to the next camp in their sequence. They can never move back to a camp they have previously visited. At least one of them must move each day, meaning that both Alice and Bob will have reached their final camps no later than the $(2n-1)$th night.

Alice and Bob would like to be able to talk to each other at night via a radio, but all of the wireless radios available for purchase have a fixed range. Alice has noticed that if they carefully plan who moves to the next camp on each day, it will affect how far apart they get during their journey. The distance between two coordinates (x_1, y_1) and (x_2, y_2) is calculated as $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

For example, let $n = 5$ and $A_1 = (1,1)$, $A_2 = (3,1)$, $A_3 = (4,1)$, $A_4 = (5,1)$, $A_5 = (6,1)$, $B_1 = (1,2)$, $B_2 = (2,2)$, $B_3 = (3,4)$, $B_4 = (4,2)$, $B_5 = (6,2)$. In this example, the furthest apart Alice and Bob **must** get during their journey is 3 units. One way this could be achieved is with the sequence of stays $[(A_1, B_1), (A_2, B_2), (A_2, B_3), (A_2, B_4), (A_3, B_4), (A_4, B_5), (A_5, B_5)]$. With this sequence of moves, the furthest apart that Alice and Bob get is on night 3, when Alice is at A_2 and Bob is at B_3.

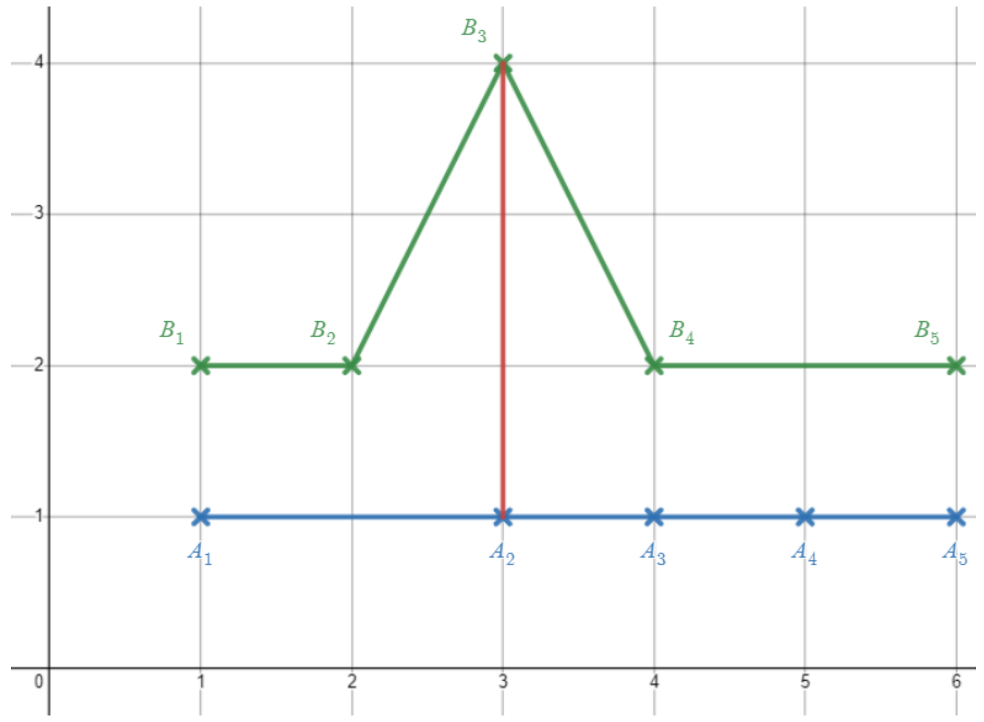


Figure 1: The simple example described above. The blue line represents the path Alice will walk and the green line represents the path Bob will walk. The red line shows the furthest apart Alice and Bob **must** get during their journey.

2.1 [**15 marks**] Design an $O(n^2)$ algorithm to determine whether a radio with a range of D will allow them to communicate every night.

In the provided example, your method should determine that it is impossible for any value of $D < 3$, and possible otherwise.

2.2 [**15 marks**] Design an $O(n^2)$ algorithm that calculates the minimum range required for Alice and Bob to be able to communicate every night.

In the provided example, your method should calculate a minimum required range of 3.

You can receive up to 7 marks for an $O(n^2 \log n)$ solution.

You may choose to skip Question 2.1, in which case your solution to Question 2.2 will also be submitted for Question 2.1.

Question 3 *KFC Now!*

[**20 marks**] There are $n \geq 0$ people queuing for KFC. The i^{th} person has an hunger of h_i, and everyone in line has a distinct hunger. A person's 'annoyance' is defined as the number of people in front of them who are less hungry, and the annoyance of the entire queue is the sum of every person's annoyance.

Given the hunger of every person, your goal is to determine how many arrangements of the queue will result in a total annoyance of exactly k.

As always, you can assume that all arithmetic operations ($+ - \times /$) are done in constant time.

3.1 [**12 marks**] Design an $O(n^2k)$ algorithm that achieves the goal.

3.2 [**8 marks**] Design an $O(nk)$ algorithm which achieves the goal.

You may choose to skip Question 3.1, in which case your solution to Question 3.2 will also be submitted for Question 3.1.

Question 4 *Antoni the Merchant*

[30 marks] Antoni wants to earn as much money as possible over the summer holidays as a travelling merchant in the DynaProg continent. He will begin his journey with \$0 wealth, and he only has D days before term starts and he is pulled back into answering forum questions, so he needs to plan very carefully. After thorough research, he has found n countries where he can turn an easy profit; in fact, he knows that if he is in country $i \in [1..n]$ on day $d \in [1..D]$, he is guaranteed to earn $P[i][d] > 0$ dollars.

All n countries have airports, and a direct flight to every other country. Each day Antoni can either travel to another country, or stay where he is. There is no restriction on how many times he can visit a certain country, nor on the number of days he can stay per visit.

However, the tax laws in DynaProg are vicious - every day he is taxed a *percentage of his current wealth*. Specifically, you are given a Tax Table $T[1..n][1..n]$ where $0 \leq T[j][i] \leq 100$ is the percentage Antoni will be taxed if he is currently in country j and was in country i the previous day. He is not taxed anything on the first day as he has \$0 wealth.

Help Antoni prepare a full itinerary (which country to be in on every day) that maximises his wealth by the end of D days.

Tax is deducted before the earnings are added for the day.

On Day 1, Antoni will not be taxed since his wealth is \$0 and there is nothing to deduct. He will then earn $\$P[i][1]$ for whichever country i he has chosen to start at.

On Day 2, he can either stay in country i, or travel to another country j. If he chooses to stay, he will be taxed $T[i][i]\%$ of his current wealth, and then earn $\$P[i][2]$. If he chooses to travel, he will be taxed $T[j][i]\%$ of his current wealth, and then earn $\$P[j][2]$.

4.1 [12 marks, 4 marks per part] Antoni has prepared a few Greedy approaches to solving this problem, but is not convinced they will always earn him the most money.

Provide a counter example for each of the following approaches. Your counter example must state the values for n, D, P, T, the wealth Greedy obtains and its itinerary, and finally, a more optimal wealth and the itinerary used to attain it. The more optimal solution does not have to be the best, it just has to beat Greedy.

- [A] On each day d, travel to the country where Antoni will earn the most money (i.e. travel to the country i such that $P[i][d]$ is maximised, for all days $1 \leq d \leq D$). If there is a tie, choose the country with the lower tax rate.
- [B] First, travel to the country with the highest Day 1 profit (i.e. travel to the country i such that $P[i][1]$ is maximised). Then, on each subsequent day d, travel to the country that minimises the amount of tax (in dollars) that Antoni will be charged (i.e. travel to the country j such that $T[j][i]\% \times$ `current_net_wealth` is minimised, for all days $2 \leq d \leq D$). If there is a tie, choose the country with the higher profit.
- [C] First, travel to the country with the highest Day 1 profit. Then, on each subsequent day d, travel to the country that maximises Antoni's wealth at the end of the day (i.e. travel to the country j such that `current_net_wealth` $\times (100 - T[j][i])\% + P[j][d]$ is maximised, for all days $2 \leq d \leq D$). Break ties arbitrarily.

4.2 [18 marks] Design an $O(n^2 D)$ algorithm that finds the maximum possible wealth Antoni can attain, and also provides the itinerary he needs to follow.