Problem 1. A company is shipping inventory with the help of a shipping container. The company has reserved two slots in the container with heights $H_1$ and $H_2$, respectively, and they can only package their $n$ pieces of inventory vertically.

You are given the heights $h_i$ and values $v_i$ for each item, and you are tasked with calculating the maximum value of inventory the company can ship.

(a) Design an algorithm that determines the maximum value in $O(H_1H_2n)$ time and space.

(b) Provide a proof of correctness of your algorithm.

(c) Justify your algorithm’s runtime and memory utilization.

Solution.
Problem 2. You are given a roll (array) of $n$ magical coins. Each coin has an initial value $c_i$ which increases at each time step linearly. Therefore, at time step $t_j$ the coin has a value of $c_{i,j}$.

You want to sell the coins to maximize your total profit, however, you are only able to sell the coins one at a time. Furthermore, you are only able to sell the coins from either end of the roll of coins.

(a) You initially spring for a greedy approach to sell your magical coins, i.e. at each time step you compare both ends of the array and sell the coin with least value. Is this strategy flawed? Explain your reasoning.

(b) Design an algorithm which determines the optimal order to sell the magical coins to maximize profit, given that you know the starting value of each coin. Your algorithm should run in $O(n^2)$ time. Provide a proof of correctness for your algorithm, and justify its runtime and memory utilization.

Solution.
Problem 3. Given an unsorted array of \( n \) positive integers, your task is to remove the least number of elements from the start or end of the array until twice the minimum of the array is larger than the maximum.

(a) Design a dynamic programming algorithm which runs in \( O(n^2) \) time. Provide a proof of correctness for your algorithm, and justify its runtime and memory utilization.

(b) Suppose the integers are sorted in non-decreasing order. Design an algorithm that runs in \( O(n) \) time. Provide a proof of correctness for your algorithm, and justify its runtime and memory utilization.

Solution.