

# Algo Assignment 7: Greedy 1

## General Instruction to submit the assignment

1. Open the link sent in email and enter your Student ID and ZEUS email address

2. You must change the programming language to Python3 before starting to code. There is a drop down button above the coding window.

- ▶ Read and solve your challenge.
- ▶ You can run code, and provide input in the text box
- ▶ You can run all the test cases, at once, it will indicate if any test case is failing
- ▶ Fix your code and run it as many times as you like before clicking submit button

3. If you submit and try to resubmit etc., the system will not let you do that. (once submitted you can not update your code)
4. Hence solve this question before hand and then enter your code in Coderbyte system. This way you will save a lot of hassle. The questions will also be released in the pdf format. Make sure your code is giving right results for all the given test cases before you submit your code.
5. You should not use **system libraries** while solving questions for at least first 3 assignments. Starting from DP assignments you can use **min max** and **sort** functions of python3.
6. Remember total points from the assignments are only **10%** of your grades. Hence these are mainly for practicing for mid-term and final-term coding test.
7. The time complexity of your submissions will be tested only after 4<sup>th</sup> Assignment.

## Bridging the Gap

A group of walkers arrives at a river in the night. They want to cross a bridge, which can hold a limited number of walkers at a time. The walkers have just one torch, which needs to be used when crossing the bridge. Each walker takes a certain time to cross; a group crossing together must walk at the slowest walker's pace. What is the shortest time it takes for all walkers to cross the bridge?

For example, Sample Input 1 assumes the bridge can hold 2 walkers at a time and there are 4 walkers with crossing times 1 minute, 2 minutes, 5 minutes and 10 minutes, respectively. The shortest time of 17 minutes can be achieved by the following sequence of crossings. First, the two fastest walkers cross in 2 minutes. Second, the fastest walker crosses back in 1 minute. Third, the two slowest walkers cross in 10 minutes. Fourth, the second-fastest walker crosses back in 2 minutes. Fifth, the two fastest walkers cross in 2 minutes.

Examples

Input: [4, 2, 1, 2, 10, 5]

Output: 17

Input: [6, 2, 1, 1, 1, 14, 14, 14]

Output: 35

1. For input [4, 2, 1, 2, 10, 5] the output was incorrect. The correct output is 17
  2. For input [6, 2, 1, 1, 1, 14, 14, 14] the output was incorrect. The correct output is 35
  3. For input [4, 6, 1, 2, 10, 5] the output was incorrect. The correct output is 10
  4. For input [5, 3, 1, 2, 3, 4, 5] the output was incorrect. The correct output is 9
  5. For input [5, 10, 1, 3, 6, 8, 12] the output was incorrect. The correct output is 12
  6. For input [2, 2, 6, 6] the output was incorrect. The correct output is 6
  7. For input [5, 4, 2, 3, 4, 6, 7] the output was incorrect. The correct output is 12
  8. For input [2, 4, 7, 8] the output was incorrect. The correct output is 8
  9. For input [4, 2, 7, 10, 13, 14] the output was incorrect. The correct output is 51
  10. For input [8, 3, 1, 1, 1, 1, 1, 1, 1, 2] the output was incorrect. The correct output is 8
-

## Bonusbudget

There are  $n$  workers in a factory. They stand in a row. Each worker has a performance evaluation value given in the integer array evaluation.

You are giving bonus to these worker in multiples of 1000\$ subjected to the following requirements:

Each worker must have at least one 1000\$ of bonus.

worker with a higher evaluation get more bonus than their neighbors.

Return the overall budget you need to have to give the decided bonus to each worker.

Time Complexity :  $O(n)$

Example 1:

Input: evaluation = [1, 0, 2]

Output: 5000

Explanation: You can allocate to the first, second and third worker with 2000\$, 1000\$, 2000\$ respectively. 1st worker has evaluation higher than his neighbor, so he will get higher bonus. 2nd worker has evaluation lower than his neighbors so he will get lower bonus. The 3rd worker has evaluation higher than his neighbor so he will have higher bonus.

Example 3:

Input: evaluation: [12, 4, 3, 11, 34, 34, 1, 67]

Output: 16000

Explanation : [3000, 2000, 1000, 2000, 3000, 2000, 1000, 2000]

Examples

Input: [12,4,3,11,34,34,1,67]

Output: 16000

Input: [1,2,3,4,6,4,3,1,2,7,6,4,3,3,2,1,1,2,2,2,4,3]

1. For input `[12, 4, 3, 11, 34, 34, 1, 67]` the output was incorrect. The correct output is `16000`
  2. For input `[1, 2, 3, 4, 6, 4, 3, 1, 2, 7, 6, 4, 3, 3, 2, 1, 1, 2, 2, 2, 4, 3]` the output was incorrect. The correct output is `47000`
  3. For input `[1, 0, 2]` the output was incorrect. The correct output is `5000`
  4. For input `[4, 4, 3, 1, 34, 0, 1, 7, 6, 6]` the output was incorrect. The correct output is `17000`
  5. For input `[1, 3, 4, 6, 3, 1, 2, 6, 33, 32, 31, 1, 2, 2, 2, 4, 3]` the output was incorrect. The correct output is `35000`
  6. For input `[1, 2, 5, 6, 7, 6, 5, 5]` the output was incorrect. The correct output is `19000`
  7. For input `[2, 3, 3, 11, 4, 16]` the output was incorrect. The correct output is `9000`
  8. For input `[1, 2, 3, 4, 6, 4, 3, 1, 2, 7, 6, 4, 3, 3, 2, 1, 1, 2, 2, 2, 4, 3, 3, 3, 13, 14, 15, 14, 13]` the output was incorrect. The correct output is `61000`
  9. For input `[10, 9]` the output was incorrect. The correct output is `3000`
- 
10. For input `[10]` the output was incorrect. The correct output is `1000`

## Predict the Maximum Profit of Bitcoin

You are a Bitcoin investor. You have got a future bitcoin price prediction from a data analysis agency in a form of an array `cost` where `cost[i]` is the cost of a bitcoin on the  $i$ -th day. On each day, you may decide to buy and/or sell the coin. You can only hold at most one coin at any time. However, you can buy it then immediately sell it on the same day. Find and return the maximum profit you can achieve.

Return type: integer

Target time complexity:  $O(n^2)$

### Examples

Input: `cost = [2,4,6,20]`, Output: 18

-- Explanation: purchase on 1st and sell it on 4th day.

Input: `cost = [14,2,10,6,12,8]`, Output: 14

-- Explanation: purchase coin on 2nd day and market it on 3rd day, Then purchase coin on day 4th day and sell on 5th day, Total profit is  $8 + 6 = 14$ .

1. For input `[2, 4, 6, 20]` the output was incorrect. The correct output is 18
  2. For input `[14, 2, 10, 6, 12, 8]` the output was incorrect. The correct output is 14
  3. For input `[18, 18, 2, 1, 12]` the output was incorrect. The correct output is 11
  4. For input `[17, 1, 14, 12, 11, 1, 16]` the output was incorrect. The correct output is 28
  5. For input `[19, 20, 11, 3, 16, 7, 18]` the output was incorrect. The correct output is 25
  6. For input `[19, 3, 3, 14, 14, 8, 9, 16, 15, 5]` the output was incorrect. The correct output is 19
  7. For input `[1, 11]` the output was incorrect. The correct output is 10
  8. For input `[1]` the output was incorrect. The correct output is 0
  9. For input `[0, 0, 0, 0, 0]` the output was incorrect. The correct output is 0
  10. For input `[5, 4, 3, 2, 1]` the output was incorrect. The correct output is 0
-

## Spice Merchant

You are a legendary spice merchant planning a long journey across the desert. You have a knapsack that can hold a maximum weight of **cap**. You arrive at a warehouse containing **N** different types of spices. Each spice is represented as an array [weight, value], indicating the total weight of the spice available and its total market value.

Since the spices are in powder form, **you can take any fraction of a spice batch**. For example, if a spice weighs 10kg and is worth 100 gold coins, you can choose to take 5kg for 50 gold coins.

Your goal is to fill your knapsack such that the **total value** of the spices carried is maximized. Find and return this maximum total value.

Return type: integer (If the calculated value contains decimals, round it **down** to the nearest integer).

### Example

Input: cost = [50, 10, 60, 20, 100, 30, 120], Output: 240

-- Explanation: You have a knapsack that can hold a maximum weight of 50. There are 10kg of spice1 with 60 gold coins for total value, 20kg of spice2 with 100 gold coins for total value, and 30kg of spice3 with 120 gold coins for total value.

To maximize the total value, you should:

1) take all of spice1 (10kg, value: 60). Remaining cap: 40

2) take all of spice2 (20kg, value: 100). Remaining cap: 20

3) take 20kg of spice3 (20kg, value: 80). Remaining cap: 0

Total: 60 + 100 + 80 = 240

1. For input [50, 10, 60, 20, 100, 30, 120] the output was incorrect. The correct output is 240
  2. For input [100, 10, 20, 20, 30, 30, 60] the output was incorrect. The correct output is 110
  3. For input [0, 100, 100, 10, 20] the output was incorrect. The correct output is 0
  4. For input [10, 100, 100, 10, 20] the output was incorrect. The correct output is 20
  5. For input [10, 5, 50, 10, 60] the output was incorrect. The correct output is 80
  6. For input [15, 10, 100, 10, 100, 10, 100] the output was incorrect. The correct output is 150
  7. For input [3, 10, 50, 20, 60] the output was incorrect. The correct output is 15
  8. For input [20, 20, 60, 10, 200, 5, 150] the output was incorrect. The correct output is 365
  9. For input [1000, 500, 5000, 200, 4000, 1000, 5000] the output was incorrect. The correct output is 10500
  10. For input [7, 3, 10] the output was incorrect. The correct output is 10
-