# Exercise: Dynamic Programming Advanced

This document defines the lab for the ["Algorithms – Advanced (C#)" course @ Software University](https://softuni.bg/trainings/3810/algorithms-advanced-with-c-sharp-september-2022).

Please submit your solutions (source code) to all below-described problems in [Judge](https://judge.softuni.org/Contests/2848/Dynamic-Programming-Advanced-Exercise).

## Cable Merchant

You are given different lengths of cable {1, 2, 3, …, n} each with a different price. For example, we are given the sequence **K = { 3, 8, 13, 15, 18, 20, 22 }**:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Length** | **1** | **2** | **3** | **4** | **5** | **6** | **7** |
| **Price** | 3 | 8 | 13 | 15 | 18 | 20 | 22 |

Instead of selling a 5m cable for 18$, you noticed you could cut that cable into parts of lengths **2m** (8$) and **3m** (13$). Then you could use 2 connectors (to connect the cables) for a price of 2 \* 1$ = 2$ and make a profit of **8 + 13 – 2 \* 1 = 19$**.

Your task is to calculate the best price for each length.

### Input

* On the first input line, you are given the sequence **K** – the prices for each length of cable.
  + The prices will be separated by a single space.
  + Each price will always refer to a length equal to its position in the sequence.
    - Ex. the first price will always be for a length of 1, the second for a length of 2, and so on, check the table above).
* On the second line, you are given the number **C** – the price for a single connector.

### Output

* Print a new sequence with the maximum prices for each length of the original sequence **K.**
* The prices should follow the original sequence order (i.e., first print the price for length 1, then the price for length 2, etc.).

### Constraints

* Each price in K will be an integer between **[1…100000]**.
* The number of elements in **K** will be between **[1…100]**.
* The price for a connector **C** will be an integer between **[0…10000]**.

### Examples

|  |  |  |
| --- | --- | --- |
| **Input** | **Output** | **Comments** |
| 3 8 13 15 18 20 22  1 | 3 8 13 15 19 24 26 | The prices of cables we have are:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | **Length** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | | **Price** | 3 | 8 | 13 | 15 | 18 | 20 | 22 |   The 4m cable which is sold for 15$ can be split into **2m (8$) + 2m (8$) = 16$**. But because of the 2 connectors \* 1$ = 2$, the total price is **16 – 2 = 14$**. That is worse than the current price 15$.  We can split 5m into **2m (8$) + 3m (13$) – 2 \* 1$ for connectors = 19$**. That is a better price than 18$.  Applying the same idea for all lengths will give us the best prices:   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | **Length** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | | **Price** | 3 | 8 | 13 | 15 | 19 | 24 | 26 | |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 391 705 1005 1493 1775 2229 2505 3010 3112 2334  38 | 391 706 1021 1493 1808 2229 2544 3010 3325 3646 |

## Battle Points

Getting battle points is easy, but you have limited energy.

You will be given **enemies** as a **sequence** of **integers** on **two** **separated** **lines**. The first line will hold the **energy** **required** to **defeat** **each** **enemy**, then on the **second** **line,** the **battle** **points** you get for **defeating each enemy**.

On the **third** line, you will get **your** **energy** points.

You have to **print** **the** **maximum** **battle** **points** you can get by **choosing** **which** **enemy** to defeat at what cost.

### Input

* The **first line** holds the energy **required** to **defeat** **each** enemy.
* On the **second line**, you will receive the **battle** **points** **gained** by **defeating** each enemy.
* At the third line, you will receive the **initial** **energy** **points** you have

### Output

* Print the **maximum** **battle** **points** you can obtain.

### Constraints

* The input will be only valid **positive** **integers,** as well as the number of enemies **[1…300]**

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 4 2 1 6 4 7 5  6 3 4 1 5 5 2  1 | 4 |
| 3 7 2 2 1 2 7  6 3 4 7 3 3 5  4 | 11 |

## Longest String Chain

Given a list of strings, write a program that returns the longest string chain that can be built from those strings.

A string chain is defined as follows: subsequence of a given sequence in which the subsequence's elements are in sorted order (string length), lowest to highest, and in which the subsequence is as long as possible.

If several sequences with equal length exist, find the left-most of them.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| a ab abcd abc | a ab abcd |
| a ab abcd abc abcd abcde | a ab abc abcd abcde |
| abde abc abd abcde ade ae 1abde abcdef | abde abcde abcdef |

## Longest Zigzag Subsequence

A zigzag sequence is one that alternately increases and decreases. More formally, such a sequence has to comply with one of the two rules below:

1. Every even element is smaller than its neighbors, and every odd element is larger than its neighbors, or
2. Every odd element is smaller than its neighbors, and every even element is larger than its neighbors

1 3 2 is a zigzag sequence, but 1 2 3 is not. Any sequence of one or two elements is zigzag.

Find the longest zigzag subsequence in a given sequence.

### **Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 8 3 5 7 0 8 9 10 20 20 20 12 19 11 | 8 3 5 0 20 12 19 11 |
| 1 2 3 | 1 2 |
| 1 3 2 | 1 3 2 |
| 24 5 31 3 3 342 51 114 52 55 56 58 | 24 5 31 3 342 51 114 52 55 |