Programming Assignment 6: Dynamic Programming 2

Revision: April 21, 2019

Introduction

In this programming assignment, you will continue practicing implementing dynamic programming solutions.

Passing Criteria: 2 out of 3

Passing this programming assignment requires passing at least 2 out of 3 programming challenges from this assignment. In turn, passing a programming challenge requires implementing a solution that passes all the tests for this problem in the grader and does so under the time and memory limits specified in the problem statement.

Contents

1 Maximum Amount of Gold 2

2 Partitioning Souvenirs 3

3 Maximizing the Value of an Arithmetic Expression 4

1 Maximum Amount of Gold

Problem Introduction

You are given a set of bars of gold and your goal is to take as much gold as possible into your bag. There is just one copy of each bar and for each bar you can either take it or not (hence you cannot take a fraction of a bar).

Problem Description

Task. Given n gold bars, find the maximum weight of gold that fits into a bag of capacity W.

Input Format. The first line of the input contains the capacity W of a knapsack and the number n of bars of gold. The next line contains n integers $w_0, w_1, \ldots, w_{n-1}$ defining the weights of the bars of gold.

Constraints. $1 \leq W \leq 10^4$; $1 \leq n \leq 300$; $0 \leq w_0, \ldots, w_{n-1} \leq 10^5$.

Output Format. Output the maximum weight of gold that fits into a knapsack of capacity W.

Sample 1.

Input:

```
10 3
1 4 8
```

Output:

```
9
```

Here, the sum of the weights of the first and the last bar is equal to 9.

Starter Files

Starter files contain an implementation of the following greedy strategy: scan the list of given bars of gold and add the current bar if it fits into the current capacity (note that, in this problem, all the items have the same value per unit of weight, for a simple reasons: they are all made of gold). As you already know from the lectures, such a greedy move is not safe. You may want to additionally submit a starter file as a solution to the grading system to ensure that this greedy algorithm indeed might produce a non-optimal result.

2 Partitioning Souvenirs

You and two of your friends have just returned back home after visiting various countries. Now you would like to evenly split all the souvenirs that all three of you bought.

Problem Description

Input Format. The first line contains an integer n. The second line contains integers $v_1, v_2, \ldots, v_n$ separated by spaces.

Constraints. $1 \leq n \leq 20$, $1 \leq v_i \leq 30$ for all i.

Output Format. Output 1, if it possible to partition $v_1, v_2, \ldots, v_n$ into three subsets with equal sums, and 0 otherwise.

Sample 1.

Input:

```
4
3 3 3 3
```

Output:

```
0
```

Sample 2.

Input:

```
1
30
```

Output:

```
0
```

Sample 3.

Input:

```
13
1 2 3 4 5 5 7 7 8 10 12 19 25
```

Output:

```
1
```

$1 + 3 + 7 + 25 = 2 + 4 + 5 + 7 + 8 + 10 = 5 + 12 + 19.$

3 Maximizing the Value of an Arithmetic Expression

Problem Introduction

In this problem, your goal is to add parentheses to a given arithmetic expression to maximize its value.

$$\max(5 - 8 + 7 \times 4 - 8 + 9) = ?$$

Problem Description

Task. Find the maximum value of an arithmetic expression by specifying the order of applying its arithmetic operations using additional parentheses.

Input Format. The only line of the input contains a string s of length $2n + 1$ for some n, with symbols $s_0, s_1, \ldots, s_{2n}$. Each symbol at an even position of s is a digit (that is, an integer from 0 to 9) while each symbol at an odd position is one of three operations from `{+,-,*}`.

Constraints. $1 \leq n \leq 14$ (hence the string contains at most 29 symbols).

Output Format. Output the maximum possible value of the given arithmetic expression among different orders of applying arithmetic operations.

Sample 1.

Input:

```
1+5
```

Output:

```
6
```

Sample 2.

Input:

```
5-8+7*4-8+9
```

Output:

```
200
```

$200 = (5 - ((8 + 7) \times (4 - (8 + 9))))$