# Programming Assignment 6: Dynamic Programming 2

Revision: November 11, 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 Maximum Value of an Arithmetic Expression** **4**

**4 Appendix** **4**

4.1 Compiler Flags . . . . . 4

4.2 Frequently Asked Questions . . . . . 5

# 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 reason: 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.

## Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

## Solution

A detailed solution (with Python code) for this challenge is covered in the companion MOOCBook. We strongly encourage you to do your best to solve the challenge yourself before looking into the book! There are at least three good reasons for this.

- By solving this challenge, you practice solving algorithmic problems similar to those given at technical interviews.
- The satisfaction and self confidence that you get when passing the grader is priceless =)
- Even if you fail to pass the grader yourself, the time will not be lost as you will better understand the solution from the book and better appreciate the beauty of the underlying ideas.

# 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
40
```

Output:

```
0
```

**Sample 3.**

Input:

```
11
17 59 34 57 17 23 67 1 18 2 59
```

Output:

```
1
```

$34 + 67 + 17 = 23 + 59 + 1 + 17 + 18 = 59 + 2 + 57$.

**Sample 4.**

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 Maximum 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.

$$\max(5-8+7\times4-8+9) = ?$$

## 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.** $1 \le n \le 14$ (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
```

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

## Need Help?

Ask a question or see the questions asked by other learners at this forum thread.

# 4 Appendix

## 4.1 Compiler Flags

**C** (`gcc 5.2.1`). File extensions: `.c`. Flags:

```
gcc -pipe -O2 -std=c11 <filename> -lm
```

**C++** (`g++ 5.2.1`). File extensions: `.cc`, `.cpp`. Flags:

```
g++ -pipe -O2 -std=c++14 <filename> -lm
```

If your `C/C++` compiler does not recognize `-std=c++14` flag, try replacing it with `-std=c++0x` flag or compiling without this flag at all (all starter solutions can be compiled without it). On Linux and MacOS, you most probably have the required compiler. On Windows, you may use your favorite compiler or install, e.g., `cygwin`.

**C#** (`mono 3.2.8`). File extensions: `.cs`. Flags:

```
mcs
```

**Go** (`golang 1.12`). File extensions: `.go`. Flags

```
go
```

**Haskell** (`ghc 7.8.4`). File extensions: `.hs`. Flags:

```
ghc -O2
```

**Java** (`Open JDK 8`). File extensions: `.java`. Flags:

```
javac -encoding UTF-8
java -Xmx1024m
```

**JavaScript** (`Node v10.15.3`). File extensions: `.js`. No flags:

```
nodejs
```

**Kotlin** (`Kotlin 1.2.21`). File extensions: `.kt`. Flags:

```
kotlinc
java -Xmx1024m
```

**Python 2** (`CPython 2.7`). File extensions: `.py2` or `.py` (a file ending in `.py` needs to have a first line which is a comment containing "python2"). No flags:

```
python2
```

**Python 3** (`CPython 3.4`). File extensions: `.py3` or `.py` (a file ending in `.py` needs to have a first line which is a comment containing "python3"). No flags:

```
python3
```

**Ruby** (`Ruby 2.1.5`). File extensions: `.rb`.

```
ruby
```

**Rust** (`Rust 1.28.0`). File extensions: `.rs`.

```
rustc
```

**Scala** (`Scala 2.11.6`). File extensions: `.scala`.

```
scalac
```

## 4.2 Frequently Asked Questions

### Why My Submission Is Not Graded?

You need to create a submission and upload the *source file* (rather than the executable file) of your solution. Make sure that after uploading the file with your solution you press the blue "Submit" button at the bottom. After that, the grading starts, and the submission being graded is enclosed in an orange rectangle. After the testing is finished, the rectangle disappears, and the results of the testing of all problems are shown.

### What Are the Possible Grading Outcomes?

There are only two outcomes: "pass" or "no pass." To pass, your program must return a correct answer on all the test cases we prepared for you, and do so under the time and memory constraints specified in the problem statement. If your solution passes, you get the corresponding feedback "Good job!" and get a point for the problem. Your solution fails if it either crashes, returns an incorrect answer, works for too long, or uses too much memory for some test case. The feedback will contain the index of the first test case on which your solution failed and the total number of test cases in the system. The tests for the problem are numbered from 1 to the total number of test cases for the problem, and the program is always tested on all the tests in the order from the first test to the test with the largest number.

Here are the possible outcomes:

- `Good job!` Hurrah! Your solution passed, and you get a point!
- `Wrong answer.` Your solution outputs incorrect answer for some test case. Check that you consider all the cases correctly, avoid integer overflow, output the required white spaces, output the floating point numbers with the required precision, don't output anything in addition to what you are asked to output in the output specification of the problem statement.
- `Time limit exceeded.` Your solution worked longer than the allowed time limit for some test case. Check again the running time of your implementation. Test your program locally on the test of maximum size specified in the problem statement and check how long it works. Check that your program doesn't wait for some input from the user which makes it to wait forever.
- `Memory limit exceeded.` Your solution used more than the allowed memory limit for some test case. Estimate the amount of memory that your program is going to use in the worst case and check that it does not exceed the memory limit. Check that your data structures fit into the memory limit. Check that you don't create large arrays or lists or vectors consisting of empty arrays or empty strings, since those in some cases still eat up memory. Test your program locally on the tests of maximum size specified in the problem statement and look at its memory consumption in the system.
- `Cannot check answer. Perhaps the output format is wrong.` This happens when you output something different than expected. For example, when you are required to output either "Yes" or "No", but instead output 1 or 0. Or your program has empty output. Or your program outputs not only the correct answer, but also some additional information (please follow the exact output format specified in the problem statement). Maybe your program doesn't output anything, because it crashes.
- `Unknown signal 6 (or 7, or 8, or 11, or some other).` This happens when your program crashes. It can be because of a division by zero, accessing memory outside of the array bounds, using uninitialized variables, overly deep recursion that triggers a stack overflow, sorting with a contradictory comparator, removing elements from an empty data structure, trying to allocate too much memory, and many other reasons. Look at your code and think about all those possibilities. Make sure that you use the same compiler and the same compiler flags as we do.
- `Internal error: exception...` Most probably, you submitted a compiled program instead of a source code.
- `Grading failed.` Something wrong happened with the system. Report this through Coursera or edX Help Center.

### May I Post My Solution at the Forum?

Please do not post any solutions at the forum or anywhere on the web, even if a solution does not pass the tests (as in this case you are still revealing parts of a correct solution). Our students follow the Honor Code: "I will not make solutions to homework, quizzes, exams, projects, and other assignments available to anyone else (except to the extent an assignment explicitly permits sharing solutions)."

### Do I Learn by Trying to Fix My Solution?

*My implementation always fails in the grader, though I already tested and stress tested it a lot. Would not it be better if you gave me a solution to this problem or at least the test cases that you use? I will then be able to fix my code and will learn how to avoid making mistakes. Otherwise, I do not feel that I learn anything from solving this problem. I am just stuck.*

First of all, learning from your mistakes is one of the best ways to learn.

The process of trying to invent new test cases that might fail your program is difficult but is often enlightening. Thinking about properties of your program makes you understand what happens inside your program and in the general algorithm you're studying much more.

Also, it is important to be able to find a bug in your implementation without knowing a test case and without having a reference solution, just like in real life. Assume that you designed an application and an annoyed user reports that it crashed. Most probably, the user will not tell you the exact sequence of operations that led to a crash. Moreover, there will be no reference application. Hence, it is important to learn how to find a bug in your implementation yourself, without a magic oracle giving you either a test case that your program fails or a reference solution. We encourage you to use programming assignments in this class as a way of practicing this important skill.

If you have already tested your program on all corner cases you can imagine, constructed a set of manual test cases, applied stress testing, etc, but your program still fails, try to ask for help on the forum. We encourage you to do this by first explaining what kind of corner cases you have already considered (it may happen that by writing such a post you will realize that you missed some corner cases!), and only afterwards asking other learners to give you more ideas for tests cases.