CS 333 - Homework #2

Due To

5 May 2023 Friday, by 11.59 pm.

Homework 2

There are 2 independent tasks in this homework.

Task1 (50 Points)

Problem Description

The input to our algorithm is going to be n total boxes. Each box has a length, width, and height. We will always call the largest dimension the box's length and the smallest dimension the box's height (i.e. for each box, $length \geq width \geq height$).

For two boxes, A and B, box A fits inside of box B provided A's length is strictly less than B's length, A's width is strictly less than B's width, and A's height is strictly less than B's height. For example, a box with dimensions $(20, 19, 18)$ fits inside a box with dimensions$(20.1, 19.5, 19)$, but it does not fit into a box with dimensions $(20, 20, 20)$.

Given a list of n boxes, each with a length, width, and height, you are to write a dynamic programming algorithm which prints the maximum nesting depth for that collection of boxes. In other words, your algorithm should print just a single integer representing the the largest possible number of boxes that can be nested. The filename for source code will be "box.py" or "Box.java".

Input:

A list of n box objects, each box object has a length, width, and heigh attribute.

Output:

Your algorithm should print just the integer representing the maximum nesting depth. You algorithm should have no other print statements. Input will be given as a file by providing the filename as a command line argument. To provide an input with n items, you will have $n + 1$ lines in your input as follows:

- The first line contains the value of n, i.e. the number of boxes in the list.

- The remaining n lines contain a space-separated list of 3 floats/doubles, each box's dimensions. The first number (which is the largest of the three) represents the length, the second represents the width, the third (which is the smallest) represents the height.

Python Sample run: python box.py test1.txt

Test Cases Summary

There are 5 test cases provided to you. The answers for these tests are listed below:

- test1.txt has answer 5
- test2.txt has answer 9
- test3.txt has answer 8
- test4.txt has answer 15
- test5.txt has answer 16

Task2 (50 Points)

Problem Introduction

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

Problem Description

Task. Find the maximum value of an arithmetic expression by specifying the order of applying its arithmetic operations using additional parentheses. The filename for source code will be "arithmetic.py" or "Arithmetic.java".

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/Print the maximum possible value of the given arithmetic expression among different orders of applying arithmetic operations. Your algorithm should print just the integer representing the maximum possible value.

Python Sample run: python arithmetic.py test1.txt

Test Cases Summary

There are 2 test cases provided to you. The answers for these tests are listed below:

- test1.txt has answer 6
- test2.txt has answer 200

Submission Rules

You will submit this homework via the LMS system. You should follow the file-naming conventions and guidelines below:

- You should submit your source files as a ZIP archive file (NOT RAR or other formats). The name of the file should be in format **"<USER-ID>_hw<HOMEWORK-NR>.zip"**. For example, if your username is vy1043, then the name of the submitted file should be "vy1043_hw2.zip". Pay attention that all the letters are in lower-case. ZIP archive is supposed to contain just the source files, under two folders corresponding to the two tasks ("Task1"and "Task2" folders).
- Late submissions and files that do not compile are not accepted.
- You can resubmit your homework (until the deadline) if you need to.
- You are not allowed to use any external libraries and/or functions. Everything must be implemented by you from scratch.
- Any type of plagiarism will not be tolerated. Your submitted codes will be compared with other submissions and also the codes available on internet and violations will have a penalty of -100 points. (In case of copying from another student both parties will get -100)