ADA - Análisis y Diseño de Algoritmos, 2022-2
Tarea 5: Semanas 7, 8 y 9

Para entregar el domingo 1 de octubre de 2023

Problemas conceptuales a las 23:59 por BrightSpace

Problemas prácticos a las 23:59 en la arena de programación

Tanto los ejercicios como los problemas deben ser resueltos, pero únicamente las soluciones de los problemas deben ser entregadas. La intención de los ejercicios es entrenarlo para que domine el material del curso; a pesar de que no debe entregar soluciones a los ejercicios, usted es responsable del material cubierto en ellos.

Instrucciones para la entrega

Para esta tarea y todas las tareas futuras, la entrega de soluciones es *individual*. Por favor escriba claramente su nombre, código de estudiante y sección en cada hoja impresa entregada o en cada archivo de código (a modo de comentario). Adicionalmente, agregue la información de fecha y nombres de compañeros con los que colaboró; igualmente cite cualquier fuente de información que utilizó.

¿Cómo describir un algoritmo?

En algunos ejercicios y problemas se pide "dar un algoritmo" para resolver un problema. Una solución debe tomar la forma de un pequeño ensayo (es decir, un par de párrafos). En particular, una solución debe resumir en un párrafo el problema y cuáles son los resultados de la solución. Además, se deben incluir párrafos con la siguiente información:

- una descripción del algoritmo en castellano y, si es útil, pseudo-código;
- por lo menos un diagrama o ejemplo que muestre cómo funciona el algoritmo;
- una demostración de la corrección del algoritmo; y
- un análisis de la complejidad temporal del algoritmo.

Recuerde que su objetivo es comunicar claramente un algoritmo. Las soluciones algorítmicas correctas y descritas *claramente* recibirán alta calificación; soluciones complejas, obtusas o mal presentadas recibirán baja calificación.

Ejercicios

La siguiente colección de ejercicios, tomados del libro de Cormen et al. es para repasar y afianzar conceptos, pero no deben ser entregados como parte de la tarea.

15.1-1, 15.1-2, 15.1-3, 15.1-4 (página 425); 15.2-1, 15.2-2, 15.2-3 (página 430); 15.3-1, 15.3-2, 15.3-3, 15.3-6, 15.3-7 (página 439).

Problemas conceptuales

1. Problema 15.1: *Coin changing* (Cormen et al., página 446).
2. Ejercicio 23: *Climbing* (Erickson, página 184).

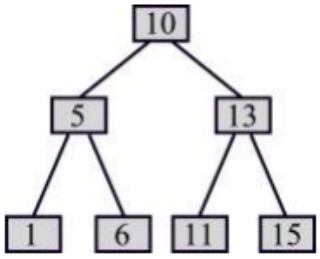
Problemas prácticos

Hay cuatro problemas prácticos cuyos enunciados aparecen a partir de la siguiente página.

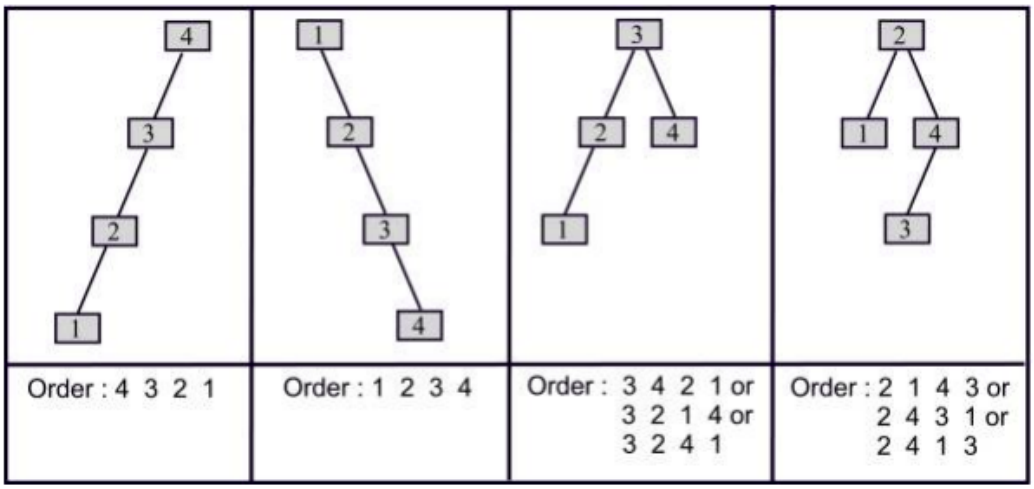
A - Constructing BST

Source file name: `bst.py`
Time limit: 1 second

BST (Binary Search Tree) is an efficient data structure for searching. In a BST all the elements of the left sub-tree are smaller and those of right sub-tree are greater than the root. A typical example of BST is



Normally, we construct BST by successively inserting an element. In that case, the ordering of elements has great impact on the structure of the tree. Look at the following cases:



In this problem, you have to find the order of 1 to N integers such that the BST constructed by them has height of at most H. The height of a BST is defined by the following relation

1. BST having no node has height 0.
2. Otherwise, it is equal to the maximum of the height of the left sub-tree and right sub-tree plus 1.

Again, several orderings can satisfy the criterion. In that case we prefer the sequence where smaller numbers come first. For example, for $N = 4$, $H = 3$ we want the sequence 1 3 2 4 rather than 2 1 4 3 or 3 2 1 4.

Input

Each test case starts with two positive integers N ($1 \leq N \leq 10000$) and H ($1 \leq H \leq 30$). Input is terminated by $N = 0$, $H = 0$. This case should not be processed. There can be at most 30 test cases.

The input must be read from standard input.

Output

Output of each test case should consist of a line starting with `'Case #:'` where `#` is the test case number. It should be followed by the sequence of *N* integers in the same line. There must not be any trailing space at the end of the line. If it is not possible to construct such tree then print `Impossible.` (without the quotes).

The output must be written to standard output.

Sample Input	**Sample Output**

`4 3`
`4 1`
`6 3`
`0 0`
	`Case 1: 1 3 2 4`
	`Case 2: Impossible.`

B - Keep the Customer Satisfied

Source file name: `customer.py`
Time limit: 1 second

Simon and Garfunkel Corporation (SG Corp.) is a large steel-making company with thousand of customers. Keeping the customer satisfied is one of the major objective of Paul and Art, the managers.

Customers issue orders that are characterized by two integer values q, the amount of steel required (in tons), and d, the due date (a calender date converted in seconds). The due date has to be met if SG Corp. accepts the order. Stated another way, when an order is accepted, the corresponding amount of steel has to be produced before its due date. Of course, the factory can process no more than one order at a time.

Although the manufacturing process is rather complex, it can be seen as a single production line with a constant throughput. In the following, we assume that producing q tons of steel takes exactly q seconds (i.e., throughput is 1). The factory runs on a monthly production plan. Before the beginning of the month, all customers' orders are collected and Paul and Art determine which of them are going to be accepted and which ones are to be rejected in the next production period. A production schedule is then designed. To keep customers satisfied, Paul and Art want to minimize the total number of orders that are rejected. In the following, we assume that the beginning of the next production plan (i.e., the first day of the next month) corresponds to date 0.

Consider the following data set made of 6 orders $J_1, \ldots, J_6$. For a given order, J_j, q_j denotes the amount of steel required and d_j is the associated due date.

Order	q_j	d_j
J_1	6	8
J_2	4	9
J_3	7	15
J_4	8	20
J_5	3	21
J_6	5	22

You can check by hand that all orders cannot be accepted and it's very unlikely you could find a solution with less than two rejected orders. Here is an optimal solution: Reject J_1 and J_4, accept all other orders, and process them as follows.

Accepted Order	Starting Time	Completion Time
J_2	0	4
J_3	4	11
J_5	11	14
J_6	14	19

Note that the production line is never idle.

Hogdson and Moore have been appointed as Chief Scientific Officers and you are requested to help them to compute an optimal solution and to build a schedule of all accepted orders (starting time and completion time).

Input

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs. In each test case, the first line contains the number n of orders (n can be as large as 800 000). It is followed by n lines. Each of which describes an order made of two integer values: the amount of steel (in tons) required for the order (lower than 1 000) and its due date (in seconds; lower than 2×10^6).

The input must be read from standard input.

Output

For each test case, you are required to compute an optimal solution and your program has to write the number of orders that are accepted. The outputs of two consecutive cases will be separated by a blank line.

The output must be written to standard output.

Sample Input	**Sample Output**
1 6 7 15 8 20 6 8 4 9 3 21 5 22	4

C - Dynamic Frog

Source file name: `frog.py`
Time limit: 1 second

With the increased use of pesticides, the local streams and rivers have become so contaminated that it has become almost impossible for the aquatic animals to survive.

Frog Fred is on the left bank of such a river. N rocks are arranged in a straight line from the left bank to the right bank. The distance between the left and the right bank is D meters. There are rocks of two sizes. The bigger ones can withstand any weight but the smaller ones start to drown as soon as any mass is placed on it. Fred has to go to the right bank where he has to collect a gift and return to the left bank where his home is situated.

He can land on every small rock at most one time, but can use the bigger ones as many times as he likes. He can never touch the polluted water as it is extremely contaminated.

Can you plan the itinerary so that the maximum distance of a single leap is minimized?

Input

The first line of input is an integer T ($T < 100$) that indicates the number of test cases. Each case starts with a line containing two integers N ($0 \leq N \leq 100$) and D ($1 \leq D \leq 1000000000$). The next line gives the description of the N stones. Each stone is defined by $S - M$. S indicates the type Big (B) or Small (S) and M ($0 < M < D$) determines the distance of that stone from the left bank. The stones will be given in increasing order of M.

The input must be read from standard input.

Output

For every case, output the case number followed by the minimized maximum leap.

The output must be written to standard output.

Sample Input	**Sample Output**

`3`
`1 10`
`B-5`
`1 10`
`S-5`
`2 10`
`B-3 S-6`
	`Case 1: 5`
	`Case 2: 10`
	`Case 3: 7`

D - Short Story Competition

Source file name: `story.py`
Time limit: x seconds

Machado wants to be a writer. He has written many short stories, book reviews, reports of trips he made, and a little romance. Now Machado wants to take part in a short story competition, which has very strict rules about the submission format.

The rules of the competition limit the number of total pages, and specify the maximum number of characters per line, and the maximum number of lines per page. Additionally, each word must be written integrally in one line (ie, a word cannot be separated in two lines). Machado wants to write a story with as many words as possible within the rules of the contest, and needs your help.

Given the maximum number of characters per line, the maximum number of lines per page, and the words of the short story that Machado is writing, he wants to know the minimum number of pages that his short story will ocuppy, considering the rules of the contest.

Input

The first line of a test case contains three integers N, L and C, which indicate, respectively, the number of words of the short story, the maximum number of lines per page and the maximum number of characters per line. Machado's short story is innovative and contains no characters besides upper and lower case letters and blanks. The second line contains Machado's short story, consisting of N word separated by exactly one blank space.

The input must be read from standard input.

Output

For each test case your program must output a single line containing a single integer indicating the minimum number of pages the short story will ocuppy, considering the contest rules.

Restrictions

- $2 \leq Nleq1000$
- $1 \leq L \leq 30$
- $1 \leq C \leq 70$
- $1 \leq \ length\ of\ each\ word\ \leq C$

The output must be written to standard output.

Sample Input

```
14 4 20
Se mana Piedade tem casado com Quincas Borba apenas me daria uma esperanca colateral
16 3 30
No dia seguinte entrou a dizer de mim nomes feios e acabou alcunhando me Dom Casmurro
5 2 2
a de i de o
5 2 2
a e i o u
```

Sample Output

```
2
1
3
3
```