Динамическое программирование.

Вводные задачи

A. *Мячик на лесенке*

На вершине лесенки, содержащей N ступенек, находится мячик, который начинает прыгать по ним вниз, к основанию. Мячик может прыгнуть на следующую ступеньку, на ступеньку через одну или через 2. (То есть, если мячик лежит на 8-ой ступеньке, то он может переместиться на 5-ую, 6-ую или 7-ую.) Определить число всевозможных «маршрутов» мячика с вершины на землю.

Вводится одно число $0 < N < 31$.

Выведите одно число — количество маршрутов.

Input	Output
4	7

B. *Последовательность из 0 и 1*

Требуется подсчитать количество последовательностей длины N, состоящих из 0 и 1, в которых никакие две единицы не стоят рядом.

На вход программы поступает целое число N ($1 \leqslant N \leqslant 10^5$).

Выведите количество искомых последовательностей по модулю $10^9 + 7$.

Input	Output
1	2

C. *Без трёх единиц*

Определите количество последовательностей из нулей и единиц длины N (длина — это общее количество нулей и единиц), в которых никакие три единицы не стоят рядом.

Вводится натуральное число N, не превосходящее 40.

Выведите количество искомых последовательностей. Гарантируется, что ответ не превосходит $2^{31} - 1$.

Input	Output
3	7

D. *Платная лестница*

Мальчик подошел к платной лестнице. Чтобы наступить на любую ступеньку, нужно заплатить указанную на ней сумму. Мальчик умеет перешагивать на следующую ступеньку, либо перепрыгивать через ступеньку. Требуется узнать, какая наименьшая сумма понадобится мальчику, чтобы добраться до верхней ступеньки.

В первой строке входного файла вводится одно натуральное число $N \leqslant 100$ — количество ступенек. В следующей строке вводятся N натуральных чисел, не превосходящих 100 — стоимость каждой ступеньки (снизу вверх).

Выведите одно число — наименьшую возможную стоимость прохода по лесенке.

Input	Output
3 1 3 1	2

Одномерное динамическое программирование

E. *Радиоактивные материалы*

При переработке радиоактивных материалов образуются отходы трех видов — особо опасные (тип `A`), неопасные (тип `B`) и совсем не опасные (тип `C`). Для их хранения используются одинаковые контейнеры. После помещения отходов в контейнеры последние укладываются вертикальной стопкой. Стопка считается взрывоопасной, если в ней подряд идет более одного контейнера типа `A`. Стопка считается безопасной, если она не является взрывоопасной.

Для заданного количества контейнеров N определить число безопасных стопок.

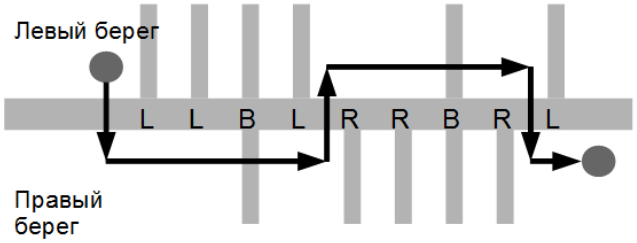
На вход программе подаётся одно число $1 \leqslant N \leqslant 10^5$.

Программа должна вывести одно число — количество безопасных вариантов формирования стопки, взятое по модулю $10^9 + 7$.

Input	Output
2	8

F. *Поход вдоль реки*

Дети пошли в поход вдоль реки. Поход начинается на левом берегу, заканчивается на правом. Дана последовательность букв 'L', 'R', 'B', означающая с какой стороны в реку впадает приток: 'L' – слева (left), 'R' – справа (right), 'B' – с обеих сторон (both). Определить минимальное количество переправ, которое придётся сделать школьникам.



На вход программа получает строку, содержащую только символы `R, L, B` в произвольном порядке. Длина строки не превосходит 10^5 символов.

Выведите одно целое число — минимальное количество переправ.

Input	Output
`LLBLRRBRL`	5

G. *Гвоздики*

В дощечке в один ряд вбиты гвоздики. Любые два гвоздика можно соединить ниточкой. Требуется соединить некоторые пары гвоздиков ниточками так, чтобы к каждому гвоздику была привязана хотя бы одна ниточка, а суммарная длина всех ниточек была минимальна.

В первой строке входных данных записано число N — количество гвоздиков ($2 \leqslant N \leqslant 100$).

В следующей строке заданы N неотрицательных целых чисел x_i ($x_i \leqslant 10^4$) — координаты всех гвоздиков в порядке возрастаниия.

Выведите единственное число — минимальную суммарную длину всех ниточек.

Input	Output
5 0 2 4 10 12	6

H. *Покупка билетов*

За билетами на премьеру нового мюзикла выстроилась очередь из N человек, каждый из которых хочет купить 1 билет. На всю очередь работала только одна касса, поэтому продажа билетов шла очень медленно, приводя «постояльцев» очереди в отчаяние. Самые сообразительные быстро заметили, что, как правило, несколько билетов в одни руки кассир продаёт быстрее, чем когда эти же билеты продаются по одному. Поэтому они предложили нескольким подряд стоящим людям отдавать деньги первому из них, чтобы он купил билеты на всех.

Однако для борьбы со спекулянтами кассир продавала не более 3-х билетов в одни руки, поэтому договориться таким образом между собой могли лишь 2 или 3 подряд стоящих человека.

Известно, что на продажу i-му человеку из очереди одного билета кассир тратит A_i секунд,

на продажу двух билетов — B_i секунд, трех билетов — C_i секунд. Напишите программу, которая подсчитает минимальное время, за которое могли быть обслужены все покупатели. Обратите внимание, что билеты на группу объединившихся людей всегда покупает первый из них. Также никто в целях ускорения не покупает лишних билетов (то есть билетов, которые никому не нужны).

На вход программы поступает сначала число N — количество покупателей в очереди ($1 \leqslant N \leqslant 5000$). Далее идет N троек натуральных чисел A_i, B_i, C_i. Каждое из этих чисел не превышает 3600. Люди в очереди нумеруются, начиная от кассы.

Требуется вывести одно число — минимальное время в секундах, за которое могли быть обслужены все покупатели.

Input	Output
5 5 10 15 2 10 15 5 5 5 20 20 1 20 1 1	12
4 100 2 100 100 2 100 100 57 100 100 1 100	59

Комментарий ко второму тесту: после того, как первый покупатель взял два билета, заплатив 2, второй покупатель *не должен ничего покупать*, так как на него билет уже куплен.

I. *Калькулятор с восстановлением ответа*

Имеется калькулятор, который выполняет три операции:

1. прибавить к числу X единицу
2. умножить число X на 2
3. умножить число X на 3

Определите кратчайшую последовательность операций, необходимую для получения из числа 1 заданное число N ($1 \leqslant N \leqslant 10^6$).

Выведите строку, состоящую из цифр 1, 2 или 3, обозначающих одну из трех указанных операций, которая получает из числа 1 число N за минимальное число операций. Если возможных минимальных решений несколько, выведите любое из них.

Указание: не стоит в массиве сохранять сами последовательности.

Input	Output
1	
5	121
562340	3333312222122213312

J. *Наибольшая возрастающая подпоследовательность за $O(n^2)$*

Дана последовательность чисел длины n. Определить длину наибольшей возрастающей подпоследовательности этой последовательности.

В этой задаче предполагается решение со сложностью $O(n^2)$.

Рассмотрим величину d_i — длину максимальной подпоследовательности, заканчивающейся элементом x_i. Тогда:

$$d[0] = 1; d_i = 1 + max\{d_j : 0 \leqslant j \leqslant i-1, x[j] < x[i]\}$$

Здесь используется следующий факт — если отбросить x_i из подпоследовательности длины $d[i]$, то подпоследовательность длины $d[i]-1$, заканчивающаяся каким-то x_j также будет оптимальной, т.е. иметь максимальную длину из всех возможных, заканчивающихся элементом x_j.

Результат: $max\{d[j] : 0 \leqslant j \leqslant n-1\}$

В первой строке входных данных задано число N — длина последовательности ($1 \leqslant N \leqslant 1000$). Во второй строке задается сама последовательность — целые числа, не превосходящие 10000 по модулю.

Требуется вывести длину наибольшей строго возрастающей подпоследовательности.

Input	Output
6 3 29 5 5 28 6	3

K. *Наибольшая возрастающая подпоследовательность с восстановлением ответа*

Дана последовательность чисел длины n. Определить наибольшую возрастающую подпоследовательность этой последовательности.

В этой задаче предполагается использовать решение предыдущей задачи со сложностью $O(n^2)$, но вместо длины вывести пример такой подпоследовательности. Если есть несколько подпоследовательностей максимальной длины — вывести любую.

Input	Output
6 3 29 5 5 28 6	3 5 28

L* *Наибольшая возрастающая подпоследовательность с восстановлением ответа*

Оказывается, предыдущую задачу можно решить гораздо быстрее (за $n \cdot \log n$), изменив схему динамического программирования.

Рассмотрим d_i — минимальное число, на которое заканчивается возрастающая подпоследовательность длины i (для программирования надо запоминать не значение, а его индекс, но думать проще про значение).

Поскольку $d_{i-1} \leqslant d_i$ для всех $i = 1 \ldots n$ и каждый элемент `x[i]` обновляет максимум одну ячейку массива `d`, то для вычисления `d` можно использовать бинарный поиск.

Числовая последовательность задана рекуррентной формулой: $a_{i+1} = (ka_i + b) \% m$. Найдите её наибольшую возрастающую подпоследовательность.

Программа получает на вход пять целых чисел: длину последовательности N ($1 \leqslant N \leqslant 10^5$), начальный элемент последовательности a_1, параметры k, b, m для вычисления последующих членов последовательности ($1 \leqslant m \leqslant 10^4, 0 \leqslant k < m, 0 \leqslant b < m, 0 \leqslant a_1 < m$).

Требуется вывести наибольшую возрастающую подпоследовательность данной последовательности, разделяя числа пробелами. Если таких последовательностей несколько, необходимо вывести одну (любую) из них.

Input	Output
5 41 2 1 100	41 67 71

Двумерное динамическое программирование

M. *Количество маршрутов*

В прямоугольной таблице $N \times M$ игрок находится в левой верхней клетке. За один ход ему разрешается перемещаться в соседнюю клетку либо вправо, либо вниз. При этом некоторые клетки таблицы посещать запрещено. Определите количество маршрутов, проходящих по разрешённым клеткам, которые ведут в правую нижнюю клетку. Гарантируется, что левая верхняя и правая нижняя клетки доступны.

Вводятся два числа N и M — размеры таблицы ($1 \leqslant N \leqslant 1000, 1 \leqslant M \leqslant 1000$). Затем в N строках вводится по M чисел: 1 означает, что в клетку можно попасть, 0 – что клетка недоступна.

Выведите искомое количество способов или слово `Impossible`, если это сделать невозможно. Гарантируется, что ответ не превосходит $2 \cdot 10^9$.

Input	Output
3 5 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1	3

N. *Самый дешёвый путь*

В некоторых клетках прямоугольного клетчатого поля с путника взимают плату. За один ход ему разрешается перемещаться в соседнюю клетку либо вправо, либо вниз (влево и вверх перемещаться запрещено). За какую минимально возможную плату можно добраться из левой верхней в правую нижнюю клетку?

Входные данные содержат два числа N и M — размеры таблицы ($1 \leqslant N \leqslant 20, 1 \leqslant M \leqslant 20$), а в следующих N строках по M чисел в каждой — размеры штрафов за прохождение через соответствующие клетки (числа от 0 до 100).

Input	Output
5 5 1 1 1 1 1 3 100 100 100 100 1 1 1 1 1 2 2 2 2 1 1 1 1 1 1	11

O. *Вывести маршрут максимальной стоимости*

В левом верхнем углу прямоугольной таблицы размером $N \times M$ находится черепашка. В каждой клетке таблицы записано некоторое число. Черепашка может перемещаться вправо или вниз, при этом маршрут черепашки заканчивается в правом нижнем углу таблицы.

Подсчитаем сумму чисел, записанных в клетках, через которую проползла черепашка (включая начальную и конечную клетку). Найдите наибольшее возможное значение этой суммы и маршрут, на котором достигается эта сумма.

В первой строке входных данных записаны два натуральных числа N и M, не превосходящих 100 — размеры таблицы. Далее идет N строк, каждая из которых содержит M чисел, разделенных пробелами — описание таблицы. Все числа в клетках таблицы целые и могут принимать значения от 0 до 100.

Первая строка выходных данных содержит максимальную возможную сумму, вторая — маршрут, на котором достигается эта сумма. Маршрут выводится в виде последовательности, которая должна содержать $N-1$ букву `D`, означающую передвижение вниз и $M-1$ букву `R`, означающую передвижение направо. Если таких последовательностей несколько, необходимо вывести ровно одну (любую) из них.

Input	Output
5 5 9 9 9 9 9 3 0 0 0 0 9 9 9 9 9 6 6 6 6 8 9 9 9 9 9	74 D D R R R R D D

P. *Наибольшая общая подпоследовательность*

Даны две последовательности, требуется найти длину их наибольшей общей подпоследовательности.

В первой строке входных данных содержится число N — длина первой последовательности ($1 \leqslant N \leqslant 1000$). Во второй строке заданы члены первой последовательности (через пробел) — целые числа, не превосходящие 10000 по модулю.

В третьей строке записано число M — длина второй последовательности ($1 \leqslant M \leqslant 1000$). В четвертой строке задаются члены второй последовательности (через пробел) — целые числа, не превосходящие 10000 по модулю.

Требуется вывести одно число — длину наибольшей общей подпоследовательности двух данных последовательностей или 0, если такой подпоследовательности нет.

Input	Output
3 1 2 3 3 2 1 3	2

Q. *Наибольшая общая подпоследовательность с восстановлением ответа*

Даны две последовательности, требуется найти и вывести их наибольшую общую подпоследовательность.

В первой строке входных данных содержится число N — длина первой последовательности ($1 \leqslant N \leqslant 1000$). Во второй строке заданы члены первой последовательности (через пробел) — целые числа, не превосходящие 10000 по модулю.

В третьей строке записано число M — длина второй последовательности ($1 \leqslant M \leqslant 1000$). В четвертой строке задаются члены второй последовательности (через пробел) — целые числа, не превосходящие 10000 по модулю.

В первой строке требуется вывести длину наибольшей общей подпоследовательности. Во второй строке требуется вывести наибольшую общую подпоследовательность данных последовательностей, разделяя числа пробелами.

Если длина общей подпоследовательности равна нулю, во второй строке ничего выводить не надо.

Input	Output
3 1 2 3 3 2 3 1	2 2 3

R. *Расстояние по Левенштейну*

Дана текстовая строка. С ней можно выполнять следующие операции:

- Заменить один символ строки на другой символ.
- Удалить один произвольный символ.
- Вставить произвольный символ в произвольное место строки.

Например, при помощи первой операции из строки `СОК` можно получить строку `СУК`, при помощи второй операции — строку `ОК`, при помощи третьей операции — строку `СТОК`.
Минимальное количество таких операций, при помощи которых можно из одной строки получить другую, называется *стоимостью редактирования* или *расстоянием Левенштейна*.
Определите расстояние Левенштейна для двух данных строк.
Программа получает на вход две строки, длина каждой из которых не превосходит 1000 символов, строки состоят только из заглавных латинских букв.
Требуется вывести одно число — расстояние Левенштейна для данных строк.

Input	Output
ABCDEFGH ACDEXGIH	3

Динамическое программирование на подотрезках

S. *Распил брусьев*

Вам нужно распилить деревянный брус на несколько кусков в заданных местах. Распилочная компания берет k рублей за распил одного бруска длиной k метров на две части.
Понятно, что различные способы распила приводят к различной суммарной стоимости заказа.
Например, рассмотрим брус длиной 10 метров, который нужно распилить на расстоянии 2, 4 и 7м, считая от одного конца. Это можно сделать несколькими способами.
Можно распилить сначала на отметке 2, потом 4 и, наконец, 7 м. Это приведет к стоимости $10 + 8 + 6 = 24$, потому что сначала длина бруса, который пилили, была 10 м, затем она стала 8м, и, наконец, 6м.
А можно распилить иначе: сначала на отметке 4, затем 2, затем 7м. Это приведет к стоимости $10 + 4 + 6 = 20$, что выгоднее.
Определите минимальную стоимость распила бруса на заданные части.
Первая строка входных данных содержит целое число L ($2 \leqslant L \leqslant 10^6$) — длину бруса и целое число N ($1 \leqslant N \leqslant 100$) — количество распилов. Во второй строке записано N целых чисел C_i ($0 < C_i < L$) в строго возрастающем порядке — места, в которых нужно сделать распилы.
Выведите одно натуральное число — минимальную стоимость распила.

Input	Output
10 3 2 4 7	20

T. *Предприниматели и золото*

Два предпринимателя А и Б делят добытое золото, разложенное по кошелькам, выложенным в ряд на столе. Процесс дележа заключается в том, что они по очереди берут себе один из крайних кошельков. Первым кошелёк забирает А. На сколько больше золота окажется у А, если оба будут придерживаться оптимальной стратегии?
На вход подаётся число $1 \leqslant N \leqslant 50$ — количество кошельков, а в следующей строке N целых положительных чисел, не превосходящих 100000, описывающих количество золота в кошельках. Требуется вычислить и вывести одно число — насколько сумма А окажется больше суммы Б (если у А окажется меньше денег, напечатайте соответствующее отрицательное число).

Input	Output
3 3 7 6	2

U. *Одномерная змейка*

Цель игрока в компьютерной игре `Snake1D` — привести ползающую по прямой змею из координаты x_1 в координату x_2. Первоначальная скорость змеи равна 1, а в некоторых местах на прямой расположены бонусы, поедание которых ускоряет змею в два раза.

На вход подаются два целых числа $0 \leqslant x_1, x_2 \leqslant 1000000$, целое число $0 \leqslant N \leqslant 50$ и N целых чисел в диапазоне [0..1000000] координаты бонусов. Бонусы можно считать точечными и поедаемыми мгновенно, никакие два бонуса не находятся в одной точке, в точках x_1 и x_2 бонусов нет.

Напечатайте минимальное время, за которое змея сможет добраться из точки x_1 в точку x_2. Ответ должен отличаться от правильного не более чем на 10^{-7}.

Input	Output
10 20 3 2 18 8	7.5

Комментарии к тесту:

Оптимальной стратегией в этом случае будет вернуться из точки 10 в точку 8, увеличить там скорость до 2, направиться в 18, где увеличить скорость до 4 и дойти наконец до точки 20. Всего будет затрачено время

$$|10 - 8| \times 1 + |18 - 8| \times 0.5 + |20 - 18| \times 0.25 = 7.5$$

Это быстрее, чем идти из точки 10 сразу в сторону точки 20.