Комплект методических материалов заключительного этапа Всероссийской олимпиады школьников по информатике 2018 года

Тексты заданий

Общая информация по задачам первого тура

Задача	Тип задачи	Ограничения
1. Добыча радия	Стандартная	1 с, 512 МБ
2. Вирусы	Стандартная	1 с, 512 МБ
3. Иннофон	Стандартная	3 с, 512 МБ
4. Квантовая телепортация	Стандартная	4 с, 512 МБ

Необходимо считывать данные из стандартного потока ввода. Выходные данные необходимо выводить в стандартный поток вывода.

Во всех задачах, баллы за подзадачу начисляются только, если все тесты для подзадачи пройдены.

Задача 1. Добыча радия

Ограничение по времени: 1 секунда

Ограничение по памяти: 512 мегабайт

Для геологической разведки перед добычей радия на плато Меридиана на орбиту Марса выведен специальный спутник, позволяющий измерять уровень радиоактивности на поверхности.

Представим плато как прямоугольник, состоящий из $n \times m$ единичных квадратов, обозначим j-й квадрат в i-м ряду как (i,j).

В результате сканирования плато для каждого единичного квадрата был определён уровень радиоактивности. Уровень радиоактивности квадрата (i,j)

задаётся целым положительным числом a_{ij}. Точность измерений настолько велика, что все числа a_{ij} различны. Единичный квадрат (i,j) считается подходящим для добычи радия, если значение a_{ij} является максимальным в i-й строке, а также максимальным в j-м столбце.

В процессе наблюдений было проведено q последовательных уточнений уровня радиоактивности. А именно, k-е уточнение изменяло значение $a_{r_k c_k}$ на некоторое строгобольшее значение. При этом после каждого уточнения все значения a_{ij} оставались различными.

Требуется написать программу, которая по заданным исходным значениям a_{ij} и списку уточнений после каждого уточнения информации определяет количество подходящих для добычи радия единичных квадратов.

Формат входных данных

Первая строка входных данных содержит три положительных целых числа: n, m и q ($1 \leqslant n \times m \leqslant 200000$, $1 \leqslant q \leqslant 200000$). Обратите внимание, что ограничение сверху дано на площадь плато, а не на количество столбцов и строк по отдельности.

Следующие n строк содержат по m положительных целых чисел, j-е число в i-й из этих строк задаёт начальное значение a_{ij} ($1 \leqslant a_{ij} \leqslant 10^7$, все a_{ij} различны).

Следующие q строк описывают уточнения данных, k-я из них содержит три целых числа r_k, c_k и x_k и задаёт изменение информации об уровне радиоактивности единичного квадрата (r_k,c_k), новое значение равно x_k ($1 \leqslant r_k \leqslant n$, $1 \leqslant c_k \leqslant m$, $1 \leqslant x_k \leqslant 10^7$). Гарантируется, что x_k строго больше предыдущего уровня радиоактивности в этом квадрате, и что все уровни радиоактивности различны после каждого изменения.

Формат выходных данных

Выходные данные должны содержать q строк, в k-й из этих строк требуется вывести одно число — количество подходящих для добычи радия единичных квадратов после k-го обновления информации.

Пример

стандартный ввод	стандартный вывод
2 3 3 1 4 3 6 5 2 2 2 9 1 3 5 2 2 10	1 2 2

Система оценивания

Подзадача	Баллы	Ограничения		Необх. подзадачи	Результаты во время тура
		n, m	q		
1	25	$1 \leqslant n \times m \leqslant 100$	$1 \leqslant q \leqslant 100$	У	Потестовые
2	25	$1 \leqslant n \times m \leqslant 5000$	$1 \leqslant q \leqslant 5000$	У, 1	Потестовые
3	25	$1 \leqslant n \leqslant 400$, $1 \leqslant m \leqslant 400$	$1 \leqslant q \leqslant 200000$	У, 1	Потестовые
4	25	$1 \leqslant n \times m \leqslant 200000$	$1 \leqslant q \leqslant 200000$	У, 1 – 3	Потестовые

Задача 2. Вирусы

Ограничение по времени: 1 секунда

Ограничение по памяти: 512 мегабайт

Одной из важнейших задач современной информатики является моделирование биологических процессов. Недавно биологи обнаружили n вирусов, каждому из

которых был присвоен уникальный кодовый номер от 1 до n. Вирус обладает возможностью встраиваться в клетки других организмов. Изначально в распоряжении ученых находятся n клеток, пронумерованных от 1 до n, при этом клетка с номером i заражена вирусом i. Каждая клетка может быть заражена только одним вирусом.

Для каждой клетки был установлен уровень её восприимчивости к каждому из вирусов. А именно, для каждой клетки известен вирус, к которому она наиболее восприимчива, к какому из оставшихся вирусов она наиболее восприимчива, и так далее.

Зараженные вирусами клетки атакуют друг друга. Пусть клетка с номером i сейчас заражена вирусом с номером a и атакует клетку с номером j, которая заражена вирусом с номером b. Тогда, если клетка с номером j является более восприимчивой к вирусу a, чем к вирусу b, то клетка с номером j становится заражена вирусом a.

В эксперименте ученые помещают все n клеток в замкнутую среду, в результате чего клетки могут атаковать друг друга произвольным образом. Эксперимент завершается, когда в результате таких атак ни для какой клетки не может измениться вирус, которым она заражена.

Ученые называют вирус с номером i стабильным, если при любой последовательности атак клетками друг друга, приводящей к завершению эксперимента, останется хотя бы одна клетка, зараженная вирусом с номером i.

Ученые называют вирус с номером i жизнеспособным, если существует такая последовательность атак клетками друг друга, приводящая к завершению эксперимента, после которой останется хотя бы одна клетка, зараженная вирусом с номером i.

Например, пусть есть два вируса, при этом клетка номер 1 наиболее восприимчива к вирусу номер 1, а клетка с номером 2 — наиболее восприимчива к вирусу номер 2. Тогда эксперимент завершается сразу: любая

атака не приводит к изменению того, каким вирусом заражена клетка. Таким образом оба вируса являются стабильными и жизнеспособными.

Пусть теперь есть два вируса, но клетка с номером 1 наиболее восприимчива к вирусу с номером 2, а клетка с номером 2 — к вирусу с номером 1. Тогда эксперимент завершается после любой атаки одной клеткой другой. Возможны два сценария. В первом сценарии клетка 1 атакует клетку 2, обе клетки становятся заражены вирусом 1. Во втором сценарии клетка 2 атакует клетку 1, после этого обе клетки становятся заражены вирусом 2. Таким образом, стабильных вирусов нет, но оба вируса являются жизнеспособными.

Наконец, пусть есть два вируса, и обе клетки более восприимчивы к вирусу с номером 1. Тогда атака клеткой 2 клетки 1 не приводит к изменению вируса, которым она заражена, а если клетка 1 атакует клетку 2, то вторая клетка становится зараженной вирусом 1. Следовательно эксперимент завершится после атаки клетки 1 клеткой 2, вирус 1 является стабильным и жизнеспособным, а вирус 2 не обладает ни тем, ни другим свойством.

Ученым необходимо отвечать на два типа вопросов, какие вирусы являются стабильными и какие вирусы являются жизнеспособными.

Требуется написать программу, которая по описанию клеток и типу вопроса определяет все стабильные, либо все жизнеспособные вирусы.

Формат входных данных

В первой строке входных данных содержится целое число n — количество вирусов и, соответственно, клеток ($1 \leqslant n \leqslant 500$).

Далее в n строках содержатся описания клеток. Для каждой клетки указано n различных чисел от 1 до n: номера вирусов в порядке убывания восприимчивости к ним этой клетки.

Последняя строка содержит число p, которая задаёт свойство вирусов, которое интересует ученых. Значение $p = 1$ означает, что ученые хотят определить все

стабильные вирусы, а значение $p = 2$ означает, что ученые хотят определить все жизнеспособные вирусы.

Формат выходных данных

Первая строка выходных данных должна содержать целое k — количество вирусов, которые обладают интересующим ученых свойством ($0 \leqslant k \leqslant n$).

Вторая строка должна содержать k целых чисел — номера вирусов, обладающих этим свойством. Номера вирусов необходимо выводить в возрастающем порядке.

Примеры

стандартный ввод	стандартный вывод
2 1 2 2 1 1	2 1 2
2 1 2 2 1 2	2 1 2
2 2 1 1 2 1	0
2 2 1 1 2 2	2 1 2

2 1 2 1 2 1	1 1
2 1 2 1 2 2	1 1
4 3 2 4 1 1 4 2 3 3 1 2 4 1 4 2 3 1	1 3
4 3 2 4 1 1 4 2 3 3 1 2 4 1 4 2 3 2	3 1 3 4

Система оценивания

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	p		
1	11	$1 \leqslant n \leqslant 5$	$p = 1$		Потестовые
2	21	$1 \leqslant n \leqslant 500$	$p = 1$	1	Первая ошибка

3	22	$1 \leqslant n \leqslant 5$		У, 1	Потестовые
4	31	$1 \leqslant n \leqslant 50$		У, 1 – 3	Первая ошибка
5	15	$1 \leqslant n \leqslant 500$		У, 1 – 4	Первая ошибка

Задача 3. Иннофон

Ограничение по времени: 3 секунды

Ограничение по памяти: 512 мегабайт

Одна телекоммуникационная компания планирует в скором будущем выпустить на рынок сразу два инновационных смартфона. Эти смартфоны будут называться «иннофон» и «иннофон плюс». Устройства уже полностью готовы к производству, и последняя задача, которую необходимо решить руководству компании, — выбрать оптимальную цену для каждого из смартфонов.

Аналитики компании провели исследование, в результате которого построили следующую модель. Всего есть n потенциальных покупателей инновационных смартфонов. Для принятия решения i-й покупатель использует следующий алгоритм, характеризующийся двумя числами a_i и b_i ($a_i \geqslant b_i$):

- если цена на «иннофон плюс» не больше a_i, то он покупает «иннофон плюс»,
- иначе, если цена на «иннофон» не больше b_i, то он покупает «иннофон»,
- иначе он не покупает ничего.

Руководство компании хочет установить цены на «иннофон» и «иннофон плюс» таким образом, чтобы обе цены были целым числом, цена «иннофона» была не больше цены «иннофона плюс», и при этом суммарная стоимость проданных смартфонов была максимальна.

Требуется написать программу, которая находит максимально возможную суммарную стоимость проданных смартфонов.

Формат входных данных

В первой строке содержится целое число n ($1 \leqslant n \leqslant 150000$) — число потенциальных покупателей.

В следующих n строках содержатся по два целых числа a_i, b_i ($0 \leqslant b_i \leqslant a_i \leqslant 10^9$) — характеристики алгоритма выбора телефона покупателем с номером i.

Формат выходных данных

Выведите одно целое число — максимальную возможную суммарную стоимость проданных смартфонов.

Примеры

стандартный ввод	стандартный вывод
5 80 20 60 50 40 40 15 10 70 30	220
1 50 0	50

Пояснение к примеру

В первом примере для достижения максимальной суммы следует назначить цены на «иннофон» и «иннофон плюс» равными 40 и 70 соответственно. Тогда первый и пятый покупатель купят «иннофон плюс», второй и третий покупатель купят «иннофон», четвертый покупатель не купит ничего. Суммарная стоимость проданных смартфонов будет 70+40+40+0+70 = 220.

Во втором примере нужно сделать цену «иннофона плюс» равной 50. Цена на «иннофон» при этом не важна.

Система оценивания

Подзадача	Баллы	Ограничения		Необх. подзадачи	Результаты во время тура
		n	Дополнительно		
1	9	$n \leqslant 100$	$b_i \leqslant a_i \leqslant 100$	У	Потестовые
2	10	$n \leqslant 300$		У, 1	Потестовые
3	16	$n \leqslant 3000$		У, 1, 2	Потестовые
4	11	$n \leqslant 10^5$	$b_i = 0$		Потестовые
5	16	$n \leqslant 10^5$	$ai = bi$		Потестовые
6	7	$n \leqslant 50000$		У, 1 – 3	Баллы
7	7	$n \leqslant 75000$		У, 1 – 3, 6	Баллы
8	8	$n \leqslant 100000$		У, 1 – 7	Баллы
9	8	$n \leqslant 125000$		У, 1 – 8	Баллы
10	8	$n \leqslant 150000$		У, 1 – 9	Баллы

Задача 4. Квантовая телепортация

Ограничение по времени: 4 секунды

Ограничение по памяти: 512 мегабайт

Ученые в IT-компании разработали квантовый суперкомпьютер. Опытный образец, разработанный учеными, содержит $n \times m$ квантовых процессоров,

организованных в виде сетки из n строк и m столбцов. Обозначим процессор в j-й ячейке i-й строки как (i,j).

Ученые запустили квантовый суперкомпьютер, однако после окончания вычислений произошел сбой в электропитании, из-за чего часть процессоров оказалась повреждена. В распоряжении исследователей осталось всего лишь k уцелевших процессоров.

Результат вычислений находится в памяти процессора (1,1), а устройство вывода подключено к процессору (n,m). Для передачи информации от одного процессора к другому используется квантовая телепортация. Особенность квантовой телепортации заключается в том, что с увеличением расстояния возникает нестабильность, требующая дополнительной энергии. Поэтому чтобы обеспечить перенос информации от процессора (x_i,y_i) к процессору (x_j,y_j) требуется $2^{\max(|xi-xj|,|yi-yj|)}$ единиц энергии. Ученые хотят перенести информацию с процессора (1,1) на процессор (n,m), затратив минимальное количество энергии. При этом можно использовать в качестве промежуточных другие уцелевшие процессоры. Использовать поврежденные процессоры нельзя.

Требуется написать программу, которая по описанию уцелевших процессоров определяет, каким образом необходимо передавать данные между процессорами, чтобы перенести информацию из процессора (1,1) в процессор (n,m), потратив минимальное суммарное количество энергии.

Формат входных данных

В первой строке входных данных находятся три целых числа n, m и k — количество строк и столбцов в сетке и количество оставшихся невредимыми после отключения электричества процессоров ($2 \leqslant n,m,k \leqslant 10000$).

Далее следуют k строк, в i-й из которых содержатся два целых числа x_i и y_i — номер строки и столбца i-го уцелевшего процессора ($1 \leqslant x_i \leqslant n$, $1 \leqslant y_i \leqslant m$).

Гарантируется, что $(x_1,y_1) = (1,1)$, $(x_k,y_k) = (n,m)$. Все процессоры находятся в разных ячейках сетки.

Формат выходных данных

Первая строка выходных данных должна содержать число L — количество процессоров, которые будут использованы при передаче информации.

Вторая строка должна содержать L чисел — номера уцелевших процессоров в том порядке, в котором они будут получать информацию. Первым должен быть выведен процессор номер 1, а последним — процессор номер k.

Если вариантов передачи информации, минимизирующих затраченную энергию, несколько, то можно вывести любой из них.

Примеры

стандартный ввод	стандартный вывод
4 5 3 1 1 2 3 4 5	3 1 2 3
5 6 9 1 1 4 3 4 6 2 5 3 1 3 3 3 6 5 4 5 6	5 1 6 2 8 9

Система оценивания

Подзадача	Баллы	Ограничения		Необх. подзадачи	Результаты во время тура
		n, m, k	Дополнительно		
1	21	$2 \leqslant n,m,k \leqslant 20$		У	Потестовые
2	13	$2 \leqslant n,m,k \leqslant 500$		У, 1	Потестовые
3	33	$2 \leqslant n,m,k \leqslant 10000$	В каждой строке таблицы и в каждом столбце таблицы находится не более одного уцелевшего процессора		Первая ошибка
4	33	$2 \leqslant n,m,k \leqslant 10000$		У, 1 – 3	Первая ошибка

Общая информация по задачам второго тура

Задача	Тип задачи	Ограничения
5. Расшифровка	Стандартная	1 с, 512 МБ
6. Быстрая сортировка	Стандартная	1 с, 512 МБ
7. Робомарафон	Стандартная	1 с, 512 МБ
8. Сложение без переносов	Стандартная	2 с, 512 МБ

Необходимо считывать данные из стандартного потока ввода. Выходные данные необходимо выводить в стандартный поток вывода.

Во всех задачах, кроме задачи 7, баллы за подзадачу начисляются только, если все тесты для подзадачи пройдены.

Задача 5. Расшифровка

Ограничение по времени: 1 секунда

Ограничение по памяти: 512 мегабайт

Известно, что если сохранить в каждом слове текста первую и последнюю букву, а остальные переставить произвольным образом, получившийся текст по-прежнему можно достаточно свободно прочитать. В лаборатории информатики исследуют аналогичный феномен для числовых последовательностей.

Будем называть последовательность, состоящую из целых положительных чисел, корректной, если первое число в этой последовательности является минимальным, а последнее — максимальным. Например, последовательности [1,3,2,4] и [1,2,1,2] являются корректными, а последовательность [1,3,2] — нет.
Задана последовательность $[a_1,a_2,...,a_n]$. Будем называть отрезок элементов заданной последовательности $[a_l,a_{l+1},...,a_r]$ корректным, если он представляет собой корректную последовательность: a_l является минимальным числом на этом отрезке, а a_r — максимальным.
В рамках исследования необходимо разбить заданную последовательность на минимальное количество непересекающихся корректных отрезков. Например, последовательность [2,3,1,1,5,1] можно разбить на три корректных отрезка: [2,3] и [1,1,5] и [1].

Требуется написать программу, которая по заданной последовательности определяет, на какое минимальное количество корректных отрезков её можно разбить.

Формат входных данных

Первая строка входных данных содержит целое число n ($1 \leqslant n \leqslant 300000$) — количество элементов в заданной последовательности.

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ — заданную последовательность ($1 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

Выведите одно число — минимальное количество корректных отрезков, на которое можно разбить заданную последовательность.

Примеры

стандартный ввод	стандартный вывод
5 5 4 3 2 1	5
4 1 3 2 4	1
6 2 3 1 1 5 1	3

Система оценивания

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Результаты во время тура
		n		
1	30	$n \leqslant 500$	У	Потестовые
2	30	$n \leqslant 5000$	У, 1	Потестовые
3	40	$n \leqslant 300000$	У, 1, 2	Первая ошибка

Задача 6. Быстрая сортировка

Ограничение по времени: 1 секунда

Ограничение по памяти: 512 мегабайт

Новый процессор UL-2018, разработанный в научной лаборатории, предназначен для быстрой обработки массивов. Ключевой особенностью архитектуры нового процессора является операция расслоения отрезка массива. Рассмотрим массив $[a_1,a_2,...,a_n]$. Операция расслоения характеризуется двумя целыми числами l и r — номером первого и последнего элемента отрезка массива, к которому она применяется. Обозначим операцию расслоения отрезка массива $[a_l,a_{l+1},...,a_r]$ как $S(l,r)$. После выполнения операции $S(l,r)$ элементы массива на этом отрезке переупорядочиваются следующим образом. Сначала располагаются элементы отрезка с позиций $a_{l+1},a_{l+3},...$, то есть элементы с позиций i, для которых значение $i-l$ нечетно, их относительный порядок остаётся неизменным. Затем идут элементы отрезка $a_l,a_{l+2},...$, то есть элементы с позиций i, для которых значение $i-l$ четно, они также сохраняют свой относительный порядок.

Например, рассмотрим массив [2,4,1,5,3,6,7,8]. После выполнения операции расслоения $S(2,6)$ изменится порядок элементов на отрезке массива [4,1,5,3,6]. Новый порядок элементов на этом отрезке следующий: [1,3,4,5,6], а весь массив после выполнения этой операции равен [2,1,3,4,5,6,7,8].

Для демонстрации возможностей нового процессора решено было использовать операцию расслоения для реализации алгоритма сортировки массива различных чисел. Задан массив, содержащий n элементов, $1 \leqslant n \leqslant 3000$. Элементы массива — различные целые числа от 1 до n. Необходимо отсортировать заданный массив по возрастанию, используя не более 15000 операций расслоения отрезка массива.

Например, приведенный выше массив [2,4,1,5,3,6,7,8] можно отсортировать, используя две операции расслоения. Сначала выполним

продемонстрированную выше операцию $S(2,6)$, после которой массив принимает вид [2,1,3,4,5,6,7,8], а затем операцию $S(1,2)$, массив примет вид [1,2,3,4,5,6,7,8].

Требуется написать программу, которая по заданному массиву определит последовательность не более, чем из 15000 операций расслоения, после выполнения которых заданный массив окажется отсортирован по возрастанию. Минимизировать количество выполненных операций расслоения не требуется, достаточно использовать не более 15000 операций.

Формат входных данных

Первая строка входных данных содержит целое число n — количество элементов в заданном массиве ($1 \leqslant n \leqslant 3000$).

Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ — элементы заданного массива ($1 \leqslant a_i \leqslant n$, все a_i различны).

Формат выходных данных

В первой строке выходных данных необходимо вывести целое число k — количество выполненных операций расслоения ($0 \leqslant k \leqslant 15000$).

В последующих k строках необходимо вывести описание самих операций, в порядке их выполнения. Каждая операция описывается двумя целыми числами l и r — номером первого и последнего элемента отрезка массива, к которому необходимо применить очередную операцию расслоения ($1 \leqslant l < r \leqslant n$).

Следует обратить внимание, что минимизировать число k не требуется. Из возможных последовательностей операций расслоения, содержащих не более 15000 операций, после выполнения которых заданный массив будет отсортирован по возрастанию, можно вывести любую. Гарантируется, что хотя бы одна такая последовательность существует.

Примеры

стандартный ввод	стандартный вывод
5 3 1 4 2 5	1 1 5
8 2 4 1 5 3 6 7 8	2 2 6 1 2
2 2 1	3 1 1 2 2 1 2

Пояснение к примеру

Второй тест из примера подробно разобран в условии задачи.

Для третьего теста из примера существует решение из одной операции расслоения, но поскольку минимизировать количество операций не требуется, приведенный в примере ответ также является правильным.

Система оценивания

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	Доп. ограничения		
1	20	$1 \leqslant n \leqslant 100$	Существует ответ с $k = 1$		Потестовые
2	30	$1 \leqslant n \leqslant 100$		У, 1	Потестовые
3	30	$1 \leqslant n \leqslant 1000$		У, 1, 2	Первая ошибка

4	20	$1 \leqslant n \leqslant 3000$		У, 1 – 3	Первая ошибка

Задача 7. Робомарафон

Ограничение по времени: 1 секунда

Ограничение по памяти: 512 мегабайт

В робомарафоне принимают участие n роботов. Роботы должны преодолеть одинаковую дистанцию, передвигаясь по расположенным рядом друг с другом дорожкам шириной один метр каждая. Известно, что расположенный на i-й дорожке робот преодолевает дистанцию за a_i секунд.

В точке старта каждого робота установлено специальное сигнальное устройство, которое должно сработать в момент старта. Чтобы сделать соревнования менее предсказуемыми, судьи перед стартом могут отключить некоторые сигнальные устройства, остальные устройства останутся активными. Только активные устройства срабатывают в тот момент, когда главный судья начинает робомарафон. В начале робомарафона хотя бы одно сигнальное устройство должно являться активным.

Каждый робот начинает движение в тот момент, когда до него доходит стартовый сигнал от активного устройства. Сигнал распространяется со скоростью 1 метр в секунду. Расстояние между дорожками i и j равно $|i - j|$ метров. Обозначим как x_i расстояние от i-й дорожки до ближайшей дорожки, содержащей активное устройство. Робот на i-й дорожке начнёт движение через x_i секунд после старта, преодолеет дистанцию за a_i секунд, и финиширует через $f_i = a_i + x_i$ секунд после старта робомарафона.

Пусть k_i — количество роботов, которые финишировали строго раньше i-го робота. Место i-го робота по итогам робомарафона равно $k_i + 1$. Если несколько роботов финишируют одновременно, а перед ними финишировали k роботов, то считается, что все они заняли $(k + 1)$-е место.

Рассмотрим пример. Пусть $n = 3$, роботы преодолевают дистанцию за $a_1 = 2$, $a_2 = 3$ и $a_3 = 5$ секунд, а активным являлось только сигнальное устройство у третьего робота. Тогда первый робот начнёт движение через 2 секунды после начала забега, $f_1 = 4$. Второй робот начнёт движение через 1 секунду, $f_2 = 4$. Третий робот начнёт движение в момент старта, $f_3 = 5$. По итогам забега первый и второй робот делят первое место, третий робот занимает третье место. Если же, например, сработают все три сигнальных устройства, роботы финишируют через $f_1 = 2$, $f_2 = 3$, $f_3 = 5$, секунд, соответственно. Первый робот займёт первое место, второй робот займёт второе место, а третий робот — третье место.

Как видно из примера, место, которое займёт робот, зависит от того, какие сигнальные устройства являются активными. Необходимо обрабатывать два типа запросов:

1) для каждого робота определить минимальное место, которое он может занять;
2) для каждого робота определить максимальное место, которое он может занять.

Требуется написать программу, которая по типу запроса и информации о времени прохождения дистанции каждым роботом определяет для каждого робота минимальное или максимальное место, которое он может занять в робомарафоне.

Формат входных данных

В первой строке входных данных находятся два целых числа: n — количество роботов ($1 \leqslant n \leqslant 400000$), и p — тип запроса. Значение $p = 1$ означает, что для каждого робота необходимо определить минимальное место, которое он может занять, значение $p = 2$ означает, что для каждого робота необходимо определить максимальное место, которое он может занять.

Во второй строке находятся n целых чисел $a_1, a_2, \ldots, a_n$ — время, за которое роботы преодолевают дистанцию ($0 < a_i \leqslant 10^9$).

Формат выходных данных

Требуется вывести n целых чисел, i-е из которых, в зависимости от типа запроса, должно задавать минимальное или максимальное место, которое может занять i-й робот.

Примеры

стандартный ввод	стандартный вывод
5 1 8 5 5 7 7	3 1 1 2 1
5 2 8 5 5 7 7	5 3 2 4 5

Система оценивания

Группа тестов для подзадачи 3 включает 30 тестов. Каждый из этих тестов оценивается независимо в 1 балл.

Группа тестов для подзадачи 4 включает 50 тестов. Каждый из этих тестов оценивается независимо в 1 балл.

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	p		
1	10	$n \leqslant 20$	$p = 1$		Первая ошибка

2	10	$n \leqslant 20$	$p = 2$		Первая ошибка
3	до 30	$n \leqslant 400000$	$p = 1$	1	Баллы
4	до 50	$n \leqslant 400000$	$p = 2$	2	Баллы

Значения n для всех тестов в подзадаче 3 приведены в следующей таблице.

Тест	n	Тест	n	Тест	n	Тест	n	Тест	n
1	$n = 100$	7	$n = 15000$	13	$n = 70000$	19	$n = 130000$	25	$n = 200000$
2	$n = 500$	8	$n = 20000$	14	$n = 80000$	20	$n = 140000$	26	$n = 240000$
3	$n = 1000$	9	$n = 30000$	15	$n = 90000$	21	$n = 150000$	27	$n = 280000$
4	$n = 2500$	10	$n = 40000$	16	$n = 99999$	22	$n = 160000$	28	$n = 320000$
5	$n = 4999$	11	$n = 50000$	17	$n = 110000$	23	$n = 170000$	29	$n = 360000$
6	$n = 10000$	12	$n = 60000$	18	$n = 120000$	24	$n = 180000$	30	$n = 400000$

Значения n для всех тестов в подзадаче 4 приведены в следующей таблице.

Тест	n	Тест	n	Тест	n	Тест	n	Тест	n
1	$n = 100$	11	$n = 2500$	21	$n = 30000$	31	$n = 109999$	41	$n = 220000$
2	$n = 200$	12	$n = 3000$	22	$n = 35000$	32	$n = 120000$	42	$n = 240000$

3	n = 300	13	n = 4000	23	n = 40000	33	n = 130000	43	n = 260000
4	n = 400	14	n = 4999	24	n = 45000	34	n = 140000	44	n = 280000
5	n = 500	15	n = 7500	25	n = 50000	35	n = 150000	45	n = 300000
6	n = 750	16	n = 10000	26	n = 60000	36	n = 160000	46	n = 320000
7	n = 1000	17	n = 12500	27	n = 70000	37	n = 170000	47	n = 340000
8	n = 1250	18	n = 15000	28	n = 80000	38	n = 180000	48	n = 360000
9	n = 1500	19	n = 20000	29	n = 90000	39	n = 190000	49	n = 380000
10	n = 1999	20	n = 25000	30	n = 99999	40	n = 200000	50	n = 400000

Задача 8. Сложение без переносов

Ограничение по времени: 2 секунды

Ограничение по памяти: 512 мегабайт

Операция побитового «или» для набора целых положительных чисел, записанных в двоичной системе счисления, устроена следующим образом. Результатом её применения является число, в двоичной записи которого единица устанавливается в тех разрядах, в которых содержится единица хотя бы у одного числа из набора.

В редких случаях побитовое «или» можно использовать для сложения целых положительных чисел, записанных в двоичной системе счисления. Сумма набора чисел равна их побитовому «или», если для каждого разряда имеется не

более одного числа из этого набора, у которого в этом разряде находится единица. Такие наборы чисел назовём красивыми.

Задан набор целых положительных чисел $a_1,a_2,...,a_n$. Необходимо построить красивый набор целых положительных чисел $b_1,b_2,...,b_n$, чтобы для всех i от 1 до n выполнялось условие $b_i \geqslant a_i$, а сумма $b_1 + b_2 + ... + b_n$ была минимальна.

Требуется написать программу, которая по двоичной записи чисел $a_1,a_2,...,a_n$ определяет двоичную запись минимального значения суммы искомого красивого набора $b_1,b_2,...,b_n$.

Формат входных данных

В первой строке записано целое число n — количество чисел в наборе ($2 \leqslant n \leqslant 300000$).

Следующие n строк содержат двоичную запись целых положительных чисел a_i, по одному в строке. Числа не содержат ведущих нулей, и суммарная длина их двоичных записей не превосходит 300000.

Формат выходных данных

Требуется вывести двоичную запись минимальной суммы искомого красивого набора $b_1,b_2,...,b_n$. Ответ необходимо вывести без ведущих нулей.

Примеры

стандартный ввод	стандартный вывод
2 10 10	110
2 10100 1001	11101
3 1	1011

1 110	

Система оценивания

Обозначим максимальную длину двоичной записи числа во входных данных как max$L$.

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	max$L$		
1	4	$n = 2$	max$L \leqslant 10$		Первая ошибка
2	2	$n = 2$	max$L \leqslant 20$	1	Первая ошибка
3	2	$n = 2$	max$L \leqslant 100$	1, 2	Первая ошибка
4	2	$n = 2$	max$L \leqslant 1000$	1 – 3	Первая ошибка
5	2	$n = 2$	max$L \leqslant 300000$	1 – 4	Первая ошибка
В подзадачах 6 – 8 дополнительно $a_i = 2^{ki}$ для некоторого k_i					
6	4	$n \leqslant 100$	max$L \leqslant 100$		Первая ошибка
7	4	$n \leqslant 1000$	max$L \leqslant 1000$	6	Первая ошибка
8	4	$n \leqslant 300000$	max$L \leqslant 300000$	6, 7	Первая ошибка
9	4	$n \leqslant 5$	max$L \leqslant 5$	У	Первая ошибка

10	4	$n \leqslant 5$	$\max L \leqslant 1000$	У, 1 – 4, 9	Первая ошибка
11	4	$n \leqslant 1000$	$\max L \leqslant 5$	У, 9	Первая ошибка
12	4	$n \leqslant 10$	$\max L \leqslant 10$	У, 1, 9	Первая ошибка
13	4	$n \leqslant 50$	$\max L \leqslant 50$	У, 1, 2, 9, 12	Первая ошибка
14	7	$n \leqslant 100$	$\max L \leqslant 100$	У, 1 – 3, 6, 9, 12, 13	Первая ошибка
15	7	$n \leqslant 300$	$\max L \leqslant 300$	У, 1 – 3, 6, 9, 12 – 14	Первая ошибка
16	8	$n \leqslant 1000$	$\max L \leqslant 1000$	У, 1 – 4, 6, 7, 9 – 15	Первая ошибка
17	8	$n \leqslant 3000$	$\max L \leqslant 3000$	У, 1 – 4, 6, 7, 9 – 16	Первая ошибка
18	6	$n \leqslant 10000$	$\max L \leqslant 10000$	У, 1 – 4, 6, 7, 9 – 17	Первая ошибка
19	7	$n \leqslant 30000$	$\max L \leqslant 30000$	У, 1 – 4, 6, 7, 9 – 18	Первая ошибка
20	7	$n \leqslant 100000$	$\max L \leqslant 100000$	У, 1 – 4, 6, 7, 9 – 19	Первая ошибка
21	6	$n \leqslant 300000$	$\max L \leqslant 300000$	У, 1 – 20	Первая ошибка

Методика оценивания выполнения олимпиадных заданий участниками

На олимпиаде предложены задачи стандартного типа, решением которых является программа, формирующая по заданным входным данным соответствующие условию задачи выходные данные. Разные задачи можно решать с использованием разных языков и сред программирования.

В тексте условия задач указаны максимальное время работы программы и размер доступной программе памяти. Временем работы программы считается суммарное время работы процесса на всех ядрах процессора. Память, используемая программным приложением, включает всю память, которая выделена процессу операционной системой, включая память кода и стек.

После завершения исполнения программа участника должна всегда завершаться с кодом 0, другие коды завершения интерпретируются проверяющей системой как ошибки.

Размер файла с исходным текстом программы не должен превышать 256 Кбайт, а время компиляции программы должно быть не больше одной минуты.

Участникам заключительного этапа Олимпиады разрешается использование в решениях задач любых внешних модулей и заголовочных файлов, установленных на компьютерах участников в составе соответствующего компилятора или среды программирования.

В решениях задач участникам запрещается:

- создавать каталоги и файлы при работе программы;
- использовать любые сетевые средства;
- создавать элементы графического интерфейса;
- совершать любые другие действия, нарушающие работу проверяющей системы.

Проверка решений участников осуществляется с использованием программной проверяющей системы в автоматическом режиме. Для осуществления проверки участники должны во время тура отправлять свои решения задач на сервер проверяющей системы с использованием соответствующих сервисов этой системы.

В процессе проверки программ-решений проверяющая система сначала проверяет, компилируется ли программа и не нарушаются ли установленные ограничения на размер исходного файла с исходным текстом программы и время ее компиляции. Затем проверяющая система запускает программу-решение с тестовыми входными данными, проверяет выполнение условий, накладываемых на время исполнения программы и объем занимаемой памяти, и, если все требования соблюдаются, то проверяет полученный ответ.

Проверка решений задач участников осуществляется во время тура или после его окончания, если она не успела завершиться во время тура. По завершению такой проверки во время тура участнику сообщается информация о результатах такой проверки, о чем сообщается в условии задачи.

В условии задач с выдачей частичной информации о результатах окончательной проверки во время тура подробно описывается, какая часть информации о результатах проверки сообщается участнику во время тура.

Тесты для каждой задачи разделены на группы для каждой подзадачи. Для каждой подзадачи могут быть указаны необходимые для нее подзадачи. В условии задачи для каждой подзадачи указывается правило начисления баллов за нее.

Общее количество баллов за задачу будет равно сумме баллов, полученных за решения каждой подзадачи.

По истечении времени тура прием решений участников на проверку проверяющей системой автоматически прекращается.

Максимальное количество баллов, которое может набрать участник по результатам проверки каждой задачи, составляет 100 баллов. Окончательные баллы участника за задачу равны максимуму из оценок за решения, отправленные на проверку.

Итоговая оценка участника формируется по результатам проверки решений всех задач и определяется как сумма баллов, полученных участником за решение каждой задачи обоих туров.

Для проверки решений используются тесты, общая характеристика которых приведена в следующей таблице.

Задача 1			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1	43	9
Подзадача 1	2 − 20	26612	4530
Подзадача 2	21 − 36	1545964	275478
Подзадача 3	37 − 46	40946744	8767816
Подзадача 4	47 − 54	38236866	4800000
Задача 2			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 8	180	64
Подзадача 1	9 − 25	815	107
Подзадача 2	26 − 33	7487322	56
Подзадача 3	34 − 48	738	177
Подзадача 4	49 − 66	127348	1059
Подзадача 5	67 − 75	8514683	12772
Задача 3			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 2	47	9
Подзадача 1	3 − 11	4024	48

Подзадача 2	12 − 23	50935	135
Подзадача 3	24 − 35	450141	151
Подзадача 4	36 − 41	4958344	84
Подзадача 5	42 − 49	11044806	112
Подзадача 6	50 − 65	13705000	218
Подзадача 7	66 − 82	21579072	238
Подзадача 8	83 − 98	26820954	225
Подзадача 9	99 − 115	35923767	241
Подзадача 10	116 − 132	43162053	241
Задача 4			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 2	74	22
Подзадача 1	3 − 35	2743	797
Подзадача 2	36 − 80	122534	15146
Подзадача 3	81 − 125	1735004	174246
Подзадача 4	126 − 183	4970058	630600
Задача 5			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 3	42	9
Подзадача 1	4 − 21	53954	80
Подзадача 2	22 − 44	771923	112
Подзадача 3	45 − 67	54946632	134
Задача 6			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 3	42	103
Подзадача 1	4 − 10	1218	7486
Подзадача 2	11 − 30	5935	38650
Подзадача 3	31 − 53	89670	597572
Подзадача 4	54 − 78	347471	2229263

Задача 7			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 2	32	30
Подзадача 1	3 − 34	5761	1849
Подзадача 2	35 − 82	9033	3449
Подзадача 3	83 − 112	34382168	24766271
Подзадача 4	113 − 162	53043458	39888986
Задача 8			
Подзадача	Тесты	Размер ввода	Размер вывода
Тесты из примера	1 − 3	41	18
Подзадача 1	4 − 13	196	109
Подзадача 2	14 − 22	346	202
Подзадача 3	23 − 32	1706	1025
Подзадача 4	33 − 44	20057	12029
Подзадача 5	45 − 61	4440644	3236579
Подзадача 6	62 − 76	75345	1861
Подзадача 7	77 − 89	3275092	13979
Подзадача 8	90 − 101	4959237	1457201
Подзадача 9	102 − 126	628	196
Подзадача 10	127 − 145	64259	16534
Подзадача 11	146 − 160	66656	10221
Подзадача 12	161 − 179	1811	305
Подзадача 13	180 − 193	24409	1054
Подзадача 14	194 − 208	90792	2104
Подзадача 15	209 − 222	785161	5998
Подзадача 16	223 − 243	5505561	21655
Подзадача 17	244 − 260	4670393	51412
Подзадача 18	261 − 277	4969861	166099
Подзадача 19	278 − 293	4781089	456253

Подзадача 20	294 − 308	4713003	1287386
Подзадача 21	309 − 319	3303530	2560022