Задача A. 1.5G — модем

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это задача с двойным запуском.

Алиса должна передать Бобу целое неотрицательное число X, меньшее 10^{10}. Единственным способом связи у Алисы является старый 1.5G модем, который при передаче произвольным образом переставляет цифры в числе.

Алиса может отправить по модему число из не более, чем m цифр (ведущие нули допустимы).

Боб получает от Алисы отправленное число с переставленными произвольным образом цифрами. Задача Боба — восстановить по полученному сообщению число X.

Формат входных данных

Ваше решение будет запущено два раза.

Первая строка входных данных содержит одно число «1» или «2» — номер запуска.

При первом запуске вторая строка содержит целое неотрицательное число X, меньшее 10^{10}, заданное без ведущих нулей. Третья строка содержит число m — максимальное количество цифр, которое может передать Алиса.

При втором запуске вторая строка содержит строку s из цифр — отправленное Алисой число с переставленными цифрами.

Формат выходных данных

При первом запуске программа должна вывести непустую строку, состоящую из не более, чем m цифр. Если ваше решение содержит более m цифр, вы получите вердикт «Wrong answer».

При втором запуске программа должна вывести одно число — предполагаемое значение X.

Система оценки

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Результаты
1	20	$m = 2023$	У	потестовые
2	30	$m = 100$	У, 1	потестовые
3	50	$m = 50$	У, 1, 2	потестовые

Замечание

Обратите внимание, что в примере приведён конкретный вариант ввода при втором запуске. В зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

Пример

Стандартный ввод	Стандартный вывод
Первый запуск	
1 2022 2023	668899
Второй запуск	
2 968896	2022

Задача B. Исправление одной ошибки

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это задача с двойным запуском.

Ваше решение будет запущено два раза.

При первом запуске ему на вход будет передана строка x из нулей и единиц длины n. Программа должна вывести строку y из нулей и единиц, длина которой не более чем m. Число m не известно вашей программе и зависит от подзадачи, соответствующее значение указано в таблице системы оценивания. Если ваша программа выведет строку длиннее, чем m, она получит вердикт «`Wrong answer`».

Между запусками решения программа жюри внесет в строку y не более одной модификации, заменив ноль на единицу или единицу на ноль, получив, таким образом, строку z.

При втором запуске программе на вход будет подана строка z. Она должна восстановить исходную строку x и выдать её на выход.

Формат входных данных

При первом запуске на первой строке ввода находится число 1. На второй строке ввода находится строка x из нулей и единиц длины n ($10 \leqslant n \leqslant 100\,000$).

При втором запуске на первой строке ввода находится число 2. На второй строке ввода находится строка z из нулей и единиц длины не больше m ($10 \leqslant m \leqslant 300\,000$). Гарантируется, что эта строка равна строке y, выведенной программой при первом запуске, или получена из неё изменением ровно одного символа на противоположный.

Формат выходных данных

При первом запуске необходимо вывести строку y, которая позволит восстановить x после внесения в нее изменения. Длина строки y не должна превышать m.

При втором запуске по заданной строке z необходимо восстановить исходную строку x и вывести её.

Система оценки

Подзадача	Баллы	Ограничения	Необходимые подзадачи	Результаты
1	40	$n = 10$, $m = 30$	У	потестовые
2	15	$n = 100\,000$, $m = 300\,000$	У, 1	потестовые
3	15	$n = 100\,000$, $m = 200\,000$	У, 1, 2	потестовые
4	15	$n = 100\,000$, $m = 101\,000$	У, 1–3	потестовые
5	15	$n = 100\,000$, $m = 100\,017$	У, 1–4	потестовые

Замечание

Обратите внимание, что в примерах приведены конкретные варианты ввода при втором запуске. В зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

Примеры

Пример 1

Стандартный ввод	Стандартный вывод
Первый запуск	
1 0000000000	00000000000000000000000000000000
Второй запуск	
2 00000000000000000000000000000000	0000000000

Пример 2

Стандартный ввод	Стандартный вывод
Первый запуск	
1 0000000000	00000000000000000000000000000000
Второй запуск	
2 00000000000010000000000000000000	0000000000

Задача C. ASCII-кодирование

Имя входного файла:	`input.dat`
Имя выходного файла:	`output.dat`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Это программа с двойным запуском.

Часто возникает необходимость кодировать двоичные данные при помощи обычных символов, используемых в текстовых файлах. Например, протокол пересылки электронной почты SMTP был разработан для передачи текстовых сообщений, а приложенные к письму файлы, которые могут быть двоичными, кодируются в виде текста. Или можно в прямо в HTML-страницу вставить изображение, закодировав его при помощи base64, как в примере `https://stackoverflow.com/questions/8499633/how-to-display-base64-images-in-html`.

Реализуйте задачу кодирования **двоичного** файла при помощи символов ASCII (однобайтовых) с кодами от 33 «`!`» до 126 «`~`», далее будем называть их *печатными символами*.

Ваша программа должна работать в одном из двух режимов: *режиме кодирования* двоичного файла в печатные символы или *режиме декодирования* ранее выведенной последовательности печатных символов и записи результата в двоичный файл.

Формат входных данных

В режиме декодирования программе подаются данные на стандартный ввод. В этом режиме программа должна учитывать только символы ASCII с кодами от 33 до 126 и игнорировать пробельные символы (пробелы, концы строк). Программа читает данные со стандартного ввода до окончания потока ввода. Если на стандартный ввод программе был подан хотя бы один печатный символ, программа осуществляет декодирование полученной последовательности печатных символов. Входные данные, которые подаются программе в режиме декодирования, совпадают с выводом вашей программы в режиме кодирования в части печатных символов, но могут отличаться добавлением и удалением пробелов и концов строк.

Если на стандартном вводе не было ни одного печатного символа, программа запускается в режиме кодирования и должна открыть на чтение файл `input.dat` в двоичном режиме и считать содержимое данного файла. Размер файла s в байтах не превышает 2^{16}, файл не пуст.

Формат выходных данных

В режиме кодирования программа выводит на стандартный вывод последовательность печатных ASCII-символов, также программа может выводить пробелы и концы строк. При этом максимальное количество печатных символов n, которое может использоваться для кодирования файлов, зависит от размера исходного файла s в байтах и группы тестов. Ограничение на n смотрите в описании групп тестов.

В режиме декодирования программа должна открыть на запись в двоичном режиме файл `output.dat` и вывести в него восстановленную последовательность байт.

Система оценки

Подзадача	Баллы	Ограничение на количество выводимых символов n	Необходимые подзадачи
1	20	$n \leqslant 8s$	У
2	20	$n \leqslant 2s$	У, 1
3	30	$n \leqslant \lceil 4s/3 \rceil$	У, 1, 2
4	30	$n \leqslant \lceil 5s/4 \rceil$	У, 1–3

Замечание

Обратите внимание, что в примере приведён конкретный вариант ввода при втором запуске. В зависимости от вашего вывода на первом запуске, ввод на втором запуске может быть другим.

Пример

В данном примере во входном файле записано три байта «sis», с ASCII-кодами 115, 105, 115.

Первый запуск	
`input.dat`	**Стандартный вывод**
`sis`	`01110011 01101001 01110011`
Второй запуск	
Стандартный ввод	`output.dat`

`01110011`
`01101001`
`01110011`
	`sis`

Указание

Для работы с двоичными файлами вы можете использовать функции языка C из заголовочного файла `cstdio` или функции языка C++ из заголовочного файла `fstream`.

В языке C необходимо открывать файл с параметром «b», например, `fopen("input.dat", "rb")`, `fopen("output.dat", "wb")`. Для чтения двоичных данных рекомендуется использовать функцию `fread`, для записи — функцию `fwrite`.

В языке C++ необходимо в конструктор передать флаг `std::ios::binary`, например, `ifstream fin("input.dat", std::ios::binary)`, `ofstream fout("output.dat", std::ios::binary)`. Для чтения двоичных данных рекомендуется использовать метод `read`, чтобы узнать количество считанных символов рекомендуется использовать метод `gcount`. Для записи двоичных данных рекомендуется метод `write`.

Задача D. Тайное послание

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1.5 секунд
Ограничение по памяти:	256 мегабайт

Это задача с двойным запуском. На каждом тесте ваше решение будет запущено два раза.

На уроке информатики Алеся и Борис изучают криптографию. Ребята решили изобрести свой способ шифрования сообщений.

Алеся выбирает k различных целых чисел от 1 до n и обозначает получившееся множество как T. Алеся хочет передать Борису в качестве сообщения множество T в зашифрованном виде. Для этого по множеству T Алеся построит и передаст Борису другое множество R, также состоящее из целых чисел от 1 до n.

Ребята не хотят, чтобы после шифрования размер сообщения изменялся, поэтому R также должно содержать ровно k чисел. А ещё они считают, что если T и R будут содержать хотя бы один общий элемент, то их шифрование будет недостаточно надежным. Поэтому не должно существовать числа, которое входит и в T, и в R, то есть множества T и R не должны пересекаться. Гарантируется, что $k \leqslant n/2$, поэтому по множеству T всегда возможно построить хотя бы одно множество R.

Когда Борис получит зашифрованное сообщение R, он должен будет его расшифровать и получить исходное сообщение T.

Помогите Алесе и Борису придумать и реализовать алгоритмы шифрования и дешифрования. При первом запуске ваша программа будет выступать в роли Алеси, а при втором запуске — в роли Бориса.

Формат входных данных

В первой строке входных данных дано одно число a, равное 1 или 2 — номер запуска вашей программы.

Во второй строке дано одно число m — количество сообщений ($1 \leqslant m \leqslant 300\,000$), которое ваша программа должна зашифровать (в первом запуске) или расшифровать (во втором запуске).

Следующие $2m$ строк содержат описания m сообщений, по две строки на сообщение.

В первой строке сообщения записаны два целых числа n_i и k_i ($2 \leqslant n_i \leqslant 10^9$, $1 \leqslant k_i \leqslant 300\,000$, $k_i \leqslant \frac{n_i}{2}$). Во второй строке сообщения записаны k_i различных целых чисел от 1 до n_i в возрастающем порядке.

Гарантируется, что сумма всех значений k_i в одном тесте не превосходит 300 000.

Если $a = 1$, то данные числа являются исходным сообщением. Если $a = 2$, то данные числа являются результатом запуска вашей программы для шифрования какого-либо сообщения при первом запуске вашей программы.

Формат выходных данных

Программа должна вывести m строк, i-я строка должна содержать k_i различных целых чисел от 1 до n_i в возрастающем порядке.

При первом запуске для каждого исходного сообщения T_i программа должна вывести множество R_i, которое не должно пересекаться с T_i.

При втором запуске программа для каждого зашифрованного сообщения R_i должна восстановить исходное сообщение T_i.

Система оценки

Чтобы указать дополнительные ограничения на входные данные, обозначим последовательность чисел, которые задают множество T_i, как t_1, t_2, …, t_{k_i}. Они расположены в порядке возрастания.

Обозначим сумму n_i в одном тесте как N.

Обозначим сумму k_i в одном тесте как K.

Подз.	Баллы	Дополнительные ограничения			Необх. подзадачи	Информация о проверке
		n_i, N	k_i, K	t_j		
1	9	$N \leqslant 5\,000$	$k_i = 1$			первая ошибка
2	11	$N \leqslant 5\,000$	$k_i = 2$			первая ошибка
3	9	$N \leqslant 300\,000$	$k_i = \frac{n_i}{2}$			первая ошибка
4	7	$n_i \leqslant 7$ $N \leqslant 5\,000$			У	первая ошибка
5	9			$t_{j+1} - t_j \geqslant 2$ $t_{k_i} \leqslant n_i - 1$		первая ошибка
6	9			$t_{k_i} - t_1 \leqslant \frac{n_i}{2} - 1$		первая ошибка
7	10	$N \leqslant 5\,000$		$t_{k_i} \leqslant n_i - k_i$		первая ошибка
8	12	$N \leqslant 100$			У	первая ошибка
9	3	$N \leqslant 5\,000$			У, 1–2, 4, 7–8	первая ошибка
10	7	$N \leqslant 300\,000$			У, 1–4, 7–9	первая ошибка
11	7		$K \leqslant 5\,000$		У, 1–2, 4, 7–9	первая ошибка
12	7	Без дополнительных ограничений			У, 1–11	первая ошибка

Пример

стандартный ввод	стандартный вывод
1 2 2 1 1 5 2 1 4	1 1 4