Департамент образования администрации города Нижнего Новгорода
Городской ресурсный центр физико-математического образования
Нижегородский государственный университет им. Н. И. Лобачевского

Четырнадцатая нижегородская городская олимпиада школьников по информатике имени В. Д. Лелюха

17 февраля 2018 г.

Нижний Новгород
2018

Результаты, архивы и другие материалы олимпиады можно найти на сайте `http://olympiads.nnov.ru`

Оригинал-макет подготовлен в системе LaTeX 2ε с использованием набора шрифтов LH.

Четырнадцатая городская олимпиада по информатике им. В. Д. Лелюха

17 февраля 2018 г. Городской ресурсный центр физико-математического образования в лице учредителей: Департамента образования администрации города Нижнего Новгорода, Нижегородского государственного университета им. Н.И. Лобачевского и МБОУ Лицей № 40 проводит тринадцатую городскую олимпиаду по информатике им. В.Д. Лелюха среди учащихся 8–11 классов образовательных учреждений города Нижнего Новгорода.

Целью проведения олимпиады является поиск талантливой молодёжи, привлечение её в науку, повышение уровня преподавания предметов физико-математического цикла.

Спонсорами городской олимпиады школьников по информатике являются НПП «Прима», компания «Яндекс», ННГУ им. Н.И. Лобачевского.

Четырнадцатая городская олимпиада проводится в ННГУ им. Лобачевского на базе Института информационных технологий, математики и механики (корпус 2).

Олимпиада проводится в соответствии с Положением о городской олимпиаде по информатике (Приказ № 633 от 06.09.2016 Департамента образования администрации г. Нижнего Новгорода).

Для участия в городской олимпиаде приглашаются талантливые школьники из Нижегородской области.

Регламент проведения олимпиады

9:30—10:00	Регистрация участников
10:00—10:30	Открытие олимпиады, информация от жюри
10:30—11:00	Ознакомление с рабочими местами, пробный тур
11:00—16:00	Решение задач олимпиады
16:00—17:00	Обед
16:00—17:00	Работа жюри
17:00—18:30	Приветствие участников олимпиады, выступления учредителей, спонсоров. Подведение итогов олимпиады. Поздравление победителей и призёров.

Состав оргкомитета олимпиады

Сидоркина С. Л., заместитель директора департамента образования администрации города Нижнего Новгорода;
Авралев Н. В., проректор по связям с общественностью ННГУ им. Н.И. Лобачевского;
Умнова Н. С., директор муниципального бюджетного образовательного учреждения лицей № 40;
Смирнов А. А., заместитель директора МБОУ лицей № 40;
Тинькова Е. В., консультант отдела общего среднего образования департамента образования администрации города Нижнего Новгорода;
Гашпар И. Л., куратор классов НОЦ ИПФ РАН.

Состав предметной комиссии (жюри)

Председатель — Калинин П. А., старший разработчик, ООО «Яндекс.Технологии», к.ф.-м.н.;

Члены комиссии:
Борисов Н. А., доцент кафедры программной инженерии ИТММ ННГУ, к.т.н;
Дроздова А. А., студентка 1 курса СПб ИТМО;
Жидков Н. В., студент 3 курса СПб АУ;
Калинин Н. А., студент 4 курса ВШОПФ ННГУ;
Кривоносов М. И., студент 2 курса магистратуры ИИТММ ННГУ;
Кузьмичёв Д. А., студент 5 курса МФТИ;
Матросов М. В., инженер-программист, НПП «АВИАКОМ»;
Семёнов Ю. Д., студент 1 курса ???;
Шмелёв А. С., инженер-программист, НПП «ПРИМА».

Ниже приведены примеры текстов программ, написанных на разрешённых языках программирования. Программы считывают данные (два числа в одной строке) с клавиатуры и выводят на экран их сумму:

Pascal

```
var a,b:integer;
begin
  read(a,b);
  writeln(a+b);
end.
```

C/C++

```
#include <iostream>
using namespace std;
int main() {
    int a, b;
    cin >> a >> b;
    cout << a + b << endl;
    return 0;
}
```

C#

```
using System;
class Program {
static void Main(string[] args) {
    string[] s = Console.ReadLine().
        Split();
    int n = Int32.Parse(s[0]);
    int m = Int32.Parse(s[1]);
    Console.Write(n + m);
}
}
```

Python 3

```
a, b = map(int, input().split())
print(a + b)
```


XIV Городская олимпиада школьников по информатике 17 февраля 2018 г.

Задача 1. ННОИ

Ввод	`nnoi.in` или стандартный ввод
Вывод	`nnoi.out` или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	200

В 2018 году проводится четырнадцатая Нижегородская городская олимпиада школьников по информатике имени В. Д. Лелюха. Считая, что далее каждый год будет проводиться одна олимпиада, определите, в каком году будет проводиться N-я?

Формат входных данных

Вводится одно целое число N — номер олимпиады ($14 \leqslant N \leqslant 1000$).

Формат выходных данных

Выведите одно число — год, в котором будет проводиться N-я олимпиада.

Пример

Ввод	*Вывод*
34	2038
14	2018

Решение

Данная задача не представляла участникам никаких сложностей, несложно видеть, что надо просто к данному числу N прибавить 2004.

Задача 2. Удобно для всех

Ввод	`contest.in` или стандартный ввод
Вывод	`contest.out` или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

В далеком будущем на Земле сутки длятся n часов, и, соответственно, есть n часовых поясов. Местное время в соседних часовых поясах различается на час. При записи местного времени для числа часов используются числа от 1 до n, т.е. времени «0 часов» не бывает, вместо этого бывает «n часов». Когда местное время в 1-м часовом поясе 1 час, местное время в i-м часовом поясе i часов.

Некоторая платформа по проведению онлайн-соревнований по программированию хочет провести соревнование длиной в один час, причем так, чтобы начало соревнования совпало бы с началом какого-то часа (во всех часовых зонах). Платформа знает, что из i-го часового пояса в соревновании хотят принять участие a_i человек. Каждый человек примет участие, если соревнование начнется не раньше s

часов 00 минут местного времени (по часовому поясу этого человека), а закончится — не позже f часов 00 минут местного времени: в это время человек на работе, а, как известно, на работе не жалко потратить час на самообразование. Начало и конец рабочего дня s и f одинаковы для всех часовых поясов. После f часов 00 минут люди уже не работают, т.е. если соревнование начинается в f часов 00 минут местного времени, то люди участвовать в нем не будут

Помогите онлайн-платформе выбрать такой час, когда наибольшее число людей примут участие в соревновании.

Формат входных данных

В первой строке записано одно целое число n — длина суток в часах ($2 \leqslant n \leqslant 100\,000$).

Во второй строке через пробел записаны n натуральных чисел a_1, a_2, …, a_n — количество людей в i-м часовом поясе, которые хотят принять участие в соревновании ($1 \leqslant a_i \leqslant 10\,000$).

В третьей строке записаны через пробел два натуральных числа s и f — времена начала рабочего дня и конца рабочего дня по местному времени, соответственно ($1 \leqslant s < f \leqslant n$).

Формат выходных данных

В единственной строке выведите одно целое число — время начала соревнований по местному времени первого часового пояса, такое, чтобы для максимального числа людей соревнование прошло в течение рабочего дня. Если ответов несколько, выведите минимальный.

Система оценивания

В тестах общей стоимостью 30 баллов N не будет превосходить 10 000.

Пример

Ввод	*Вывод*
3 1 2 3 1 3	3
5 1 2 3 4 1 1 3	4

Примечание: В первом примере оптимально начать соревнование в 3 часа по времени первого часового пояса. Тогда во втором часовом поясе будет 1 час, а в третьем — 2 часа. Только один человек из первого часового пояса не примет участия в соревновании, т. к. он заканчивает работать в момент начала соревнования.

Во втором примере в соревновании примут участие люди из третьего и четвертого часовых поясов.

Решение

Заранее посчитаем префиксные суммы — общее количество людей в часовых поясах с первого по i-й для каждого i. Теперь переберем время начала соревнования. В

зависимости от него, у нас не работают люди из некоторого отрезка (или двух отрезков) часовых поясов. Будем узнавать, сколько людей из тех, что должны прийти, у нас сейчас не работают (за $O(1)$ с помощью насчитанных заранее сумм), и выберем наилучший ответ. Итоговая асимптотика $O(N)$.

Задача 3. Честный дележ

Ввод	`divide.in` или стандартный ввод
Вывод	`divide.out` или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

Трое друзей отправились на рыбалку. Они беззаботно ловили рыбу, наслаждаясь природой и хорошей компанией.

Но вот подошла к концу совместная поездка, и стало понятно, что кто-то поймал больше, а кто-то меньше. Один из друзей, поймавший меньше всех, неожиданно заявил, что если он приедет домой с такими результатами, то его жена, объявив его жалким неудачником, уйдет от него. Поэтому было решено разделить весь улов на троих. Вам надо придумать, как организовать обмен рыбы между друзьями, чтобы все в итоге вернулись домой с одинаковым уловом.

Более формально, вам дан список из n попыток друзей поймать рыбу. Для каждой попытки известно, кто пытался поймать рыбу, и сколько он рыб он вытащил (в штуках). Вам требуется вывести таблицу размера 3 на 3, где на j-м месте в i-й строке будет находиться число $a_{i,j}$, указывающее, сколько рыб i-й человек должен получить от j-го. Это число может быть как положительным, так и отрицательным (в случае, если i-й человек должен отдать рыбу j-му). Для вашей таблицы должно выполняться равенство $a_{i,j} = -a_{j,i}$ (в частности, из этого следует, что $a_{i,i} = 0$).

Суммарный итоговый улов каждого из друзей должен быть одинаковыми (улов i-го человека складываются из той рыбы, которую он поймал сам, и той рыбы, которую он получит в результате итогового обмена).

Гарантируется, что существует такой способ обменяться рыбой, чтобы каждый рыбак ушел домой с одинаковым целым количеством рыб.

Формат входных данных

В первой строке вводится одна строка, состоящая только из букв латинского алфавита — имя первого из друзей. Имя непустое и его длина не превосходит 10. Во второй и третей строках в таком же формате вводятся имена второго и третьего друга. Гарантируется, что все имена различны.

В четвертой строке вводится одно целое число n — количество попыток поймать рыбу ($1 \leqslant n \leqslant 100$).

В следующих n строках вводятся сами попытки — сначала имя человека, который удил рыбу (гарантируется, что это имя было указано в одной из первых трех строк входных данных), затем через пробел одно целое число c_i — сколько рыб он поймал ($1 \leqslant c_i \leqslant 10^4$).

Гарантируется, что суммарный улов можно разделить на троих поровну, оставляя рыбы целыми.

Формат выходных данных

Выведите 3 строки, в каждой по 3 целых числа. В i-й строке j-е число должно быть равно $a_{i,j}$ — сколько рыб (в штуках) должен получить i-й человек от j-го (нумерация друзей соответствует порядку, в котором они даны во входных данных).

Для выведенных чисел должны выполнятся следующие условия:

- $a_{i,j} = -a_{j,i}$;
- $a_{i,j}$ — целое число, удовлетворяющее ограничению $|a_{i,j}| \leqslant 10^7$;
- суммарный улов каждого из друзей (с учетом попыток, указанных во входных данных) одинаков.

Можно показать, что хотя бы один способ подобрать такие $a_{i,j}$ существует.

Пример

Ввод	*Вывод*
Vlad Kolya Misha 1 Kolya 3	0 0 1 0 0 -2 -1 2 0

Примечание: В первом примере ловил рыбу только Коля. Один из вариантов честного дележа такой: Коля отдает Мише 2 рыбы, потом Миша отдает Владу 1 рыбу.

Решение

Так как мы можем пользоваться только целыми числами, то итоговый улов должен быть не только одинаковым, но и целым. Значит, общий улов (сумма c_i) должен делиться на 3. Обозначим за x требуемый итоговый улов каждого друга (то есть $\left(\sum_{i=1}^{n} c_i\right)/3$). Пусть cnt_i - текущий улов i-ого человека (до обмена рыбой). Теперь легко вывести 3 уравнения, которые нам необходимо удовлетворить:

$cnt_0 + a_{0,1} + a_{0,2} = x$

$cnt_1 + a_{1,0} + a_{1,2} = x$

$cnt_2 + a_{2,0} + a_{2,1} = x$

Учитывая, что $cnt_0 + cnt_1 + cnt_2 = 3x$ и $a_{i,j} = -a_{j,i}$, можно понять, что мы имеем 3 неизвестных на 3 зависимых уравнения (уравнения зависимы, так как, если мы их сложим, то получим тождество $0 = 0$). Поэтому достаточно зафиксировать одну из неизвестных, остальные выражаются однозначно. Например, подойдет следующее решение:

$a_{0,1} = a_{1,0} = 0$

$-a_{2,0} = a_{0,2} = x - cnt_0$

$-a_{2,1} = a_{1,2} = x - cnt_1$

Задача 4. Бутявочная ферма

Ввод	butyavki.in или стандартный ввод
Вывод	butyavki.out или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

У мальчика Димы есть своя бутявочная ферма. Скоро на ней вырастут N бутявок, и Диме надо будет отвезти их на рынок продавать.

Бутявок надо перевозить в стеклянных банках. Если какая-то банка будет заполнена не полностью, то бутявки в этой банке закузявятся, поэтому каждую банку надо заполнять полностью.

Дима может покупать банки на стеклозаводе. Завод производит K типов банок, банки i-го типа вмещают a_i бутявок каждая. Дима может заказать на заводе сколько угодно банок, но, чтобы получить оптовую скидку, он должен заказывать банки только какого-то одного типа.

Естественно, Дима хочет заказать банки так, чтобы можно было каждую банку заполнить бутявками доверху и отвезти на рынок; если при этом какие-то бутявки не влезут, то Дима оставит этих бутявок на ферме.

Определите, сколько банок и какого типа должен заказать Дима, чтобы увезти на рынок как можно больше бутявок.

Формат входных данных

Первая строка входных данных содержит два целых числа N и K — количество бутявок, которые вырастут у Димы на ферме, и количество типов банок, которые производит стеклозавод ($0 \leqslant N \leqslant 10^{18}$, $1 \leqslant K \leqslant 10^5$).

Во второй строке находятся K целых чисел a_1, a_2, …, a_K — вместимости каждого типа банок ($1 \leqslant a_i \leqslant 10^{18}$ для всех i).

Формат выходных данных

Выведите два числа — номер типа банок, которые должен заказать Дима, и их количество. Типы банок нумеруются от 1 до K в том порядке, как они описаны во входных данных.

Если верных ответов несколько, вы можете вывести любой из них.

Система оценивания

В тестах общей стоимостью 25 баллов все числа во входных данных не будут превосходить 1000.

В тестах общей стоимостью 50 баллов все числа во входных данных не будут превосходить 10^5.

Пример

Ввод	*Вывод*
19 3 5 4 10	2 4
28 3 5 6 30	1 5

Решение

Самое простое решение этой задачи состояло в том, чтобы искать не максимальное количество бутявок, которых можно отвезти на рынок, а минимальное количество бутявок, которых можно оставить на ферме. Если Дима купит банки i-го типа, то очевидно, что на ферме останется $n \bmod a_i$ бутявок, здесь mod — операция взятия остатка от деления. Соответственно, надо выбрать такой тип банок x, чтобы величина $n \bmod a_x$ была минимальной среди всех x; необходимое количество таких банок будет, очевидно, $n \operatorname{div} a_x$, где div — операция взятия неполного частного.

Задача 5. Максимизируй!

Ввод	`maximum.in` или стандартный ввод
Вывод	`maximum.out` или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

Дано множество S из натуральных чисел, изначально пустое. С ним производятся операции двух типов:

1. Добавить в S натуральное число, не меньшее всех уже лежащих в множестве элементов.

2. Найти непустое подмножество s множества S такое, что значение $\max(s) - \mathrm{mean}(s)$ максимально. Здесь $\max(s)$ обозначает максимум среди всех элементов s, $\mathrm{mean}(s)$ — среднее арифметическое всех элементов s. Вывести величину $\max(s) - \mathrm{mean}(s)$.

Формат входных данных

В первой строке задано единственное натуральное число Q ($1 \leqslant Q \leqslant 5 \cdot 10^5$) — количество операций. В каждой из следующих Q строк следует описание очередной операции. Для операций типа 1 в строке будут записаны два числа вида 1 и x, где x ($1 \leqslant x \leqslant 10^9$) — натуральное число, которое нужно добавить в S. Гарантируется, что x будет не меньше любого элемента S на данный момент. Для операции типа 2 в соответствующей строке будет записано ровно одно число 2.

Гарантируется, что первая операция будет иметь тип 1, то есть множество S не будет пустым при поступлении операций типа 2.

Формат выходных данных

Выведите ответ на каждую операцию второго типа в том порядке, в котором эти операции шли во входных данных. Каждое число должно располагаться на отдельной строке и содержать минимум шесть верных значащих цифр.

Система оценивания

Решения, правильно работающие для $Q \leqslant 1400$, будут набирать не менее 30 баллов.

Решения, правильно работающие для $Q \leqslant 20\,000$, будут набирать не менее 44 баллов.

Пример

Ввод	*Вывод*
6 1 3 2 1 4 2 1 8 2	0.0000000000 0.5000000000 3.0000000000
4 1 1 1 4 1 5 2	2.0000000000

Решение

Сначала докажем необходимые леммы. Будем обозначать $a_0, \ldots, a_n$ — числа, на текущий момент лежащие в S, упорядоченные по возрастанию.

Лемма 1. Среди оптимальных множеств s всегда существует такое, в которое входит максимальный на данный момент элемент множества S, т.е. a_n.

Доказательство: рассмотрим произвольное оптимальное множество s. Пусть в него не входит элемент a_n, пусть максимальный элемент множества s — это a_i при $i < n$. Заменим этот элемент в множестве s на a_n. Обозначая $\Delta = a_n - a_i$, видим, что величина $\max(s)$ увеличилась на Δ, а величина $\mathrm{mean}(s)$ увеличилась на Δ/k, где k — количество элементов в множестве s. Поэтому величина $\max(s) - \mathrm{mean}(s)$ увеличилась на $\Delta(1 - 1/k)$, что неотрицательно, т.к. $\Delta \geqslant 0$ и $k \geqslant 1$. Следовательно, новое множество также оптимальное.

Таким образом, можно искать решение только среди множеств, содержащих a_n.

Лемма 2. Существует такое оптимальное множество s, что в нем наибольший элемент — это a_n, а все остальные элементы образуют префикс массива a (т.е. помимо a_n в множество s входят элементы a_0, a_1, …, a_k при некотором k, и только они).

Доказательство: предположим, это не так. Рассмотрим a_i, где $i < n$, — первый элемент из a, который не лежит в s. Из предположения известно, что в s лежит некоторый a_j такой, что $i < j < n$. Заменим в s элемент a_j на a_i, среднее $\mathrm{mean}(s)$ не увеличится, так как $a_i \leqslant a_j$, а $\max(s)$ не изменится. Таким образом, величина $\max(s) - \mathrm{mean}(s)$ не уменьшится, т.е. новое множество тоже оптимальное. Повторяя такие операции, получим оптимальное множество, удовлетворяющее указанному в лемме требованию.

Лемма 3. Обозначим $m_i = a_n - \left(a_n + \sum_{j=0}^{i-1} a_j\right)/(i+1)$ — это значение величины $\max(s) - \mathrm{mean}(s)$, которую мы хотим максимизировать, если s состоит из a_n, а также из префикса a длины i, то есть элементов $a_0, \ldots, a_{i-1}$. Утверждение:

$$\operatorname{sign}(m_{i+1} - m_i) = \operatorname{sign}\left(a_n + \sum_{j=0}^{i-1} a_j - a_i \cdot (i+1)\right),$$

где $sign(x)$ обозначает знак числа x.

Доказательство:

$$\begin{aligned} m_{i+1} - m_i &= \frac{(-a_n - \sum_{j=0}^{i} a_j)(i+1) + (a_n + \sum_{j=0}^{i-1} a_j)(i+2)}{(i+1)(i+2)} \\ &= \frac{a_n + \sum_{j=0}^{i-1} a_j - a_i(i+1)}{(i+1)(i+2)}. \end{aligned}$$

Так как знаменатель этой дроби всегда положителен, то утверждение леммы верно.

Лемма 4. Обозначим $f(i) = a_n + \sum_{j=0}^{i-1} a_j - a_i(i+1)$. Тогда $f(i)$ монотонно не возрастает при увеличении i.

Доказательство:

$$\begin{aligned} f(i+1) &= f(i) + a_i - a_{i+1} \cdot (i+2) + a_i \cdot (i+1) \\ &= f(i) - (i+2) \cdot (a_{i+1} - a_i). \end{aligned}$$

Имеем $a_{i+1} - a_i \geqslant 0$, потому что a упорядочено по возрастанию. Тогда $f(i+1) - f(i) = -(i+2)(a_{i+1} - a_i) \leqslant 0$, то есть $f(i+1) - f(i) \leqslant 0$, что и означает нестрогое убывание $f(i)$ при увеличении i.

Таким образом, если мы рассматриваем множество s, состоящее из элементов a_0, a_1, …, a_i и еще a_n, то целевая величина $\max(s) - \operatorname{mean}(s)$ при увеличении i сначала возрастает, а потом убывает. Поэтому, чтобы найти оптимальное i, мы можем использовать бинарный поиск и искать наименьшее i такое, что $f(i) \leqslant 0$. Из лемм следует, что именно префикс длины i будет оптимальным для фиксированного $\max(s) = a_n$. Чтобы быстро считать суммы вида $\sum_{j=0}^{i} a_j$, будем поддерживать в отдельном массиве суммы на всех префиксах a —к этому массиву будет добавляться один элемент каждый раз при поступлении запроса типа 1.

Задача 6. Котлета

Ввод	`cutlet.in` или стандартный ввод
Вывод	`cutlet.out` или стандартный вывод
Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

Аркадий хочет обедать. Он только что вернулся из магазина, где приобрел полуфабрикат — котлету, которую осталось только поджарить. Надпись на упаковке гласит, что котлету нужно жарить на сковороде на умеренном огне ровно $2n$ секунд, причем сначала нужно ровно n секунд жарить котлету на одной стороне, а

затем ровно n секунд — на другой. Аркадий уже нашел сковороду и зажег умеренный огонь, но тут осознал, что, возможно, у него не получится перевернуть котлету ровно через n секунд после начала готовки.

Аркадий слишком занят расстановкой по порядку наборов стикеров в своем любимом мессенджере, и может отвлечься на переворот котлеты только в определенные моменты времени. А именно, есть k промежутков времени, в которые он может это сделать, i-й из них — это отрезок времени с l_i секунд от начала готовки до r_i секунд от начала готовки, включительно. Аркадий решил, что не обязательно переворачивать котлету ровно в середине готовки, вместо этого, он перевернет ее несколько раз таким образом, чтобы суммарно котлета провела n секунд на одной стороне и n секунд на другой.

Помогите Аркадию — узнайте, может ли он выполнить свой план, если он может переворачивать котлету только в указанные отрезки времени, и если да, то какое минимальное число раз ему придется перевернуть котлету.

Формат входных данных

Первая строка содержит два целых числа n и k ($1 \leqslant n \leqslant 500\,000$, $1 \leqslant k \leqslant 100$) — число секунд, которое котлета должна жариться с каждой из сторон, и число промежутков времени, когда Аркадий может ее переворачивать.

Следующие k строк содержат описания этих промежутков. Каждая строка содержит два целых числа l_i и r_i ($0 \leqslant l_i \leqslant r_i \leqslant 2 \cdot n$), что означает, что Аркадий может перевернуть котлету в любой момент времени, начиная с l_i секунд от начала готовки и заканчивая r_i секундами от начала готовки, включительно. В частности, если $l_i = r_i$, то Аркадий может перевернуть котлету в момент времени $l_i = r_i$. Гарантируется, что $l_i > r_{i-1}$ для всех $2 \leqslant i \leqslant k$.

Формат выходных данных

Выведите единственное слово «`Hungry`», если Аркадий не может пожарить котлету ровно n секунд на одной стороне и ровно n секунд на другой.

В противном случае, выведите в первую строку одно слово «`Full`», а во вторую — минимальное количество раз, которое ему необходимо перевернуть котлету.

Система оценивания

Тесты, в которых в правильном ответе котлету нужно переворачивать не более двух раз, или Аркадий не может пожарить котлету требуемым образом, имеют суммарную стоимость не менее 30 баллов.

Тесты, в которых $n \leqslant 1000$, имеют суммарную стоимость не менее 40 баллов.

Пример

Ввод	*Вывод*
10 2 3 5 11 13	Full 2
10 3 3 5 9 10 11 13	Full 1
20 1 3 19	Hungry

Примечание: В первом примере котлету нужно перевернуть в моменты времени 3 секунды после начала и 13 секунд после начала.

Во втором примере котлету можно перевернуть, как и написано на упаковке, через 10 секунд после начала.

Решение

Для получения 30 баллов достаточно заметить, что, если возможно пожарить котлету за один переворот, то ее нужно перевернуть в момент времени n секунд с начала готовки. Легко проверить, возможен ли такой переворот, проверив каждый отрезок отдельно. Чтобы проверить, можно ли приготовить котлету за два переворота, заметим, что для этого надо сделать перевороты в момент времени x и $x + n$ секунд с начала, где $0 < x < n$. Также заметим, что эти перевороты должны соответствовать различным отрезкам времени, когда Аркадий свободен, так как в обратном случае можно было бы пожарить котлету за один переворот. Таким образом, достаточно перебрать все пары отрезков, когда Андрей свободен, перенести более поздний отрезок на n секунд раньше и проверить, что пересечение с более ранним отрезком не пусто. Если же котлету нельзя пожарить ни за один переворот, ни за два, то нужно вывести «`Hungry`».

Для получения 40 баллов необходимо воспользоваться методом динамического программирования. Решим следующую подзадачу: пусть прошло t секунд с момента начала готовки, при этом из них t_0 секунд котлета жарилась на той стороне, на которой она лежит сейчас; какое минимальное число переворотов необходимо, чтобы достичь такой ситуации? Это легко определить, используя ответы для подзадач $(t-1, t_0)$, $(t-1, t_0-1)$, в которых котлета лежит на той же стороне, и $(t-1, t-1-t_0)$ и $(t-1, t-t_0)$, в которых котлета лежит на другой стороне. Необходимо аккуратно учесть, какие переходы возможны с учетом занятости Аркадия, а также сколько переворотов котлеты необходимо для каждого перехода. Итоговая асимптотика $O(n^2)$.

Для полного решения необходимо рассматривать только моменты времени, соответствующие началу очередного отрезка, когда Аркадий свободен, т. к. между такими моментами времени можно делать максимум два переворота котлеты, в противном случае результат точно не будет оптимальным. Тогда окажется, что для того, чтобы посчитать ответ для подзадачи $(t = l_i, t_0)$, достаточно вычислить

минимальный из ответов среди подзадач $(t' = l_{i-1}, t'_0)$, где t' — время начала предыдущего отрезка — фиксировано, а t'_0 изменяется на нескольких отрезках времени, границы которых зависят от числа переворотов котлеты в промежутке от l_{i-1} до l_i секунд, а также параметров l_{i-1}, r_{i-1}, l_i, t_0. Для эффективного вычисления такого минимума можно использовать очередь минимумов, так как границы отрезка увеличиваются с увеличением t_0. Итоговая асимптотика $O(nk)$. Стоит отметить, что существуют решения, в которых очередь минимумов заменена на какую-нибудь другую структуру данных, например, дерево Фенвика, дерево отрезков или разреженные таблицы, однако, такие решения имеют асимптотику $O(nk \log(n))$, и с ними достаточно сложно получить полный балл.

Задача 7. Спасение любви

Ввод	`tshirt.in` или стандартный ввод
Вывод	`tshirt.out` или стандартный вывод
Ограничение по времени	1 секунда
Ограничение по памяти	256 мегабайт
Максимальный балл за задачу	100

Валя и Толя — идеальная пара, но даже у них бывают ссоры. Недавно Валя обиделась на своего кавалера, так как он пришел к ней в футболке, надпись на которой отличается от надписи на ее свитере. Теперь она не хочет с ним видеться, а Толя сидит целыми днями в своей комнате и плачет над ее фотографиями.

Эта история так и осталась бы такой печальной, если бы в нее не вмешалась добрая швея-волшебница (бабушка Толи). Ее сердце разрывается, когда она видит молодых людей в ссоре, и поэтому она срочно хочет все исправить. Бабушка уже тайно забрала Валин свитер и Толину футболку, осталось только сделать надписи на них одинаковыми. Для этого она может за одну единицу маны купить заклинание, которое позволяет менять некоторые буквы на одежде. Ваша задача — рассчитать, какое минимальное количество маны придется потратить бабушке Толи для спасения любви молодых людей.

Более формально, надписи на Валином свитере и Толиной футболке — это две строчки одинаковой длины n, состоящие только из маленьких букв латинского алфавита. За одну единицу маны бабушка может купить заклинание вида (c_1, c_2) (где c_1 и c_2 — произвольные строчные буквы латинского алфавита), с помощью которого она может сколько угодно раз менять букву c_1 на c_2 (и наоборот) как на Валином свитере, так и на Толиной футболке. Вам требуется найти минимальное количество маны, позволяющее приобрести набор заклинаний, с помощью которого можно сделать надписи одинаковыми. Также вам необходимо вывести соответствующий набор заклинаний.

Формат входных данных

В первой строке дано единственное натуральное число n — длина надписей $(1 \leqslant n \leqslant 10^5)$.

Во второй строке дана строка длины n, состоящая только из маленьких букв латинского алфавита — надпись на Валином свитере.

В третьей строке в таком же формате задана надпись на Толиной футболке.

Формат выходных данных

В первой строке выведите одно целое число — минимальное количество маны t для спасения любви молодых.

В следующих t строках выведите по две строчные буквы латинского алфавита через пробел — заклинания, которые нужно приобрести бабушке Толи. Заклинания и буквы в заклинаниях можно выводить в любом порядке.

Если оптимальных ответов несколько, выведите любой.

Пример

Ввод	*Вывод*
3 abb dad	2 a d b a
8 drpepper cocacola	7 l e e d d c c p p o o r r a

Примечание: В первом примере достаточно купить два заклинания: («a»,«d») и («b»,«a»). Тогда первые буквы совпадут, когда мы поменяем букву «a» на «d». Вторые совпадут, когда поменяем «b» на «a». Третьи совпадут, когда поменяем сначала «b» на «a», потом «a» на «d».

Решение

Рассмотрим граф на 26 буквах английского алфавита. Когда мы покупаем заклинание вида (c_1, c_2), проведем неориентированное ребро между вершинами c_1 и c_2. Теперь нетрудно понять, что мы можем поменять символ a на символ b тогда и только тогда, когда в этом графе существует путь между этими символами. Поэтому наша задача — построить минимальное количество ребер, чтобы символы $s_1[i]$ и $s_2[i]$ были в одной компоненте связности для любого i (здесь s_1 и s_2 — данные строки).

Теперь возьмем пустой граф на 26 буквах английского алфавита и проведем ребра между $s_1[i]$ и $s_2[i]$ для всех i. Эти ребра, как мы уже выяснили, задают ограничения на символы (то есть они говорят, что соединенные ребром символы обязаны лежать в одной компоненте связности в искомом графе заклинаний). Посчитаем в этом графе количество компонент связности — пусть это k. Рассмотрим одну компоненту связности, пусть ее размер x_i. Заметим, что купленные нами заклинания должны соединять все эти вершины в одну компоненту. Известно, что для этого нужно хотя бы $x_i - 1$ ребер, подходящим набором ребер является любое остовное дерево этой компоненты связности. Тогда суммарно нам нужно не меньше, чем $\sum_{i=1}^{k}(x_i - 1) = \sum_{i=1}^{k} x_i - k = n - k$ ребер. Это и есть ответ на задачу.

Содержание