Использовать стандартную библиотеку (std::lower_bound, std::upper_bound, std::sort, std::stable_sort, std::nth_element, java.util.Arrays.sort, java.util.Arrays.binarySearch, std::priority_queue, java.util.PriorityQueue) не разрешается. Все задачи решаются с помощью изученных тем, никаких неизученных структур данных не требуется.

Задача A. Сортировка

Имя входного файла:	`sort.in`
Имя выходного файла:	`sort.out`
Ограничение по времени:	2 секунды

Дан массив целых чисел. Ваша задача — отсортировать его в порядке неубывания.

Формат входных данных

В первой строке входного файла содержится число n ($1 \leqslant n \leqslant 300\,000$) — количество элементов в массиве. Во второй строке находятся n целых чисел, по модулю не превосходящих 10^9.

Формат выходных данных

В выходной файл надо вывести этот же массив в порядке неубывания, между любыми двумя числами должен стоять ровно один пробел.

Пример

sort.in	sort.out
10 1 8 2 1 4 7 3 2 3 6	1 1 2 2 3 3 4 6 7 8

Задача B. Двоичный поиск

Имя входного файла:	`binsearch.in`
Имя выходного файла:	`binsearch.out`
Ограничение по времени:	2 секунды

Дан массив из n элементов, упорядоченный в порядке неубывания и m запросов: найти первое и последнее вхождение числа в массив.

Формат входных данных

В первую строке входного файла содержится одно число n — размер массива.($1 \leqslant n \leqslant 100000$). Во второй строке находится n чисел в порядке неубывания — элементы массива. В третьей строке находится число m — количество запросов. В следующей строке находится m чисел — запросы.

Формат выходных данных

Для каждого запроса выведите в отдельной строке номер первого и последнего вхождения этого числа в массив. Если числа в массиве нет выведите два раза -1.

Пример

binsearch.in	binsearch.out
5 1 1 2 2 2 3 1 2 3	1 2 3 5 -1 -1

Задача C. K-ая порядковая статистика

Имя входного файла:	`kth.in`
Имя выходного файла:	`kth.out`
Ограничение по времени:	2 секунды

Дан массив из n элементов. Какое число k-е в порядке возрастания в этом массиве.

Формат входных данных

В первую строке входного файла содержится два числа n — размер массива и k.($1 \leqslant k \leqslant n \leqslant 3 \cdot 10^7$). Во второй строке находятся числа A, B, C, a_1, a_2 по модулю не превосходящие 10^9. Вы должны получить элементы массива начиная с третьего по формуле: $a_i = A * a_{i-2} + B * a_{i-1} + C$. Все вычисления должны производится в 32 битном знаковом типе, переполнения должны игнорироваться.

Формат выходных данных

Выведите значение k-ое в порядке возрастания число в массиве a.

Пример

kth.in	kth.out
5 3 2 3 5 1 2	13
5 3 200000 300000 5 1 2	2

Во втором примере элементы массива a равны: (1, 2, 800005, −516268571, 1331571109).

Задача D. Гирлянда

Имя входного файла:	`garland.in`
Имя выходного файла:	`garland.out`
Ограничение по времени:	2 секунды

Гирлянда состоит из n лампочек на общем проводе. Один её конец закреплён на заданной высоте A мм ($h_1 = A$). Благодаря силе тяжести гирлянда прогибается: высота каждой неконцевой лампы на 1 мм меньше, чем средняя высота ближайших соседей ($h_i = \frac{(h_{i-1}+h_{i+1})}{2} - 1$ для $1 < i < N$). Требуется найти минимальную высоту второго конца B ($B = h_n$) при условии, что ни одна из лампочек не должна лежать на земле ($h_i > 0$ для $1 \leqslant i \leqslant N$).

Формат входных данных

В первую строке входного файла содержится два числа n и A ($3 \leqslant n \leqslant 1000$, n —целое, $10 \leqslant A \leqslant 1000$, A —вещественное).

Формат выходных данных

Вывести одно вещественное число B с двумя знаками после запятой.

Пример

garland.in	garland.out
8 15	9.75
692 532.81	446113.34

Задача E. Цифровая сортировка

Имя входного файла:	`radixsort.in`
Имя выходного файла:	`radixsort.out`
Ограничение по времени:	2 секунды

Дано n строк, выведите их порядок после k фаз цифровой сортировки.

Формат входных данных

В первой строке входного файла содержится число n — количество строк, m — их длина и k – число фаз цифровой сортировки ($1 \leqslant n \leqslant 1000$, $1 \leqslant k \leqslant m \leqslant 1000$). В следующих n строках находятся сами строки.

Формат выходных данных

Выведите строки в порядке в котором они будут после k фаз цифровой сортировки.

Пример

radixsort.in	radixsort.out
3 3 1 bbb aba baa	aba baa bbb
3 3 2 bbb aba baa	baa aba bbb
3 3 3 bbb aba baa	aba baa bbb

Задача F. Анти-QuickSort

Имя входного файла:	`antiqs.in`
Имя выходного файла:	`antiqs.out`
Ограничение по времени:	2 секунды

Для сортировки последовательности чисел широко используется быстрая сортировка - QuickSort. Далее приведена программа, которая сортирует массив a, используя этот алгоритм.

```
var
   a : array [1..N] of integer;

   procedure QSort(left, right : integer);
   var
      i, j : integer;
      key : integer;
      buf : integer;
   begin
      key := a[(left + right) div 2];
      i := left;
      j := right;
      repeat
         while a[i] < key do     {first while}
            inc(i);
         while key < a[j] do     {second while}
            dec(j);
         if i <= j then begin
            buf := a[i];
            a[i] := a[j];
            a[j] := buf;
            inc(i);
            dec(j);
         end;
      until i > j;

      if left < j then
         QSort(left, j);
      if i < right then
         QSort(i, right);
   end;

begin
   ...
   QSort(1, N);
end.
```

Хотя QuickSort является самой быстрой сортировкой в среднем, существуют тесты, на которых она работает очень долго. Оценивать время работы алгоритма будем количеством сравнений с элементами массива (то есть суммарным количеством сравнений в первом и втором while). Требуется написать программу, генерирующую тест, на котором быстрая сортировка сделает наибольшее число таких сравнений.

Формат входных данных

В первой строке находится единственное число n ($1 \leqslant n \leqslant 70000$).

Формат выходных данных

Вывести перестановку чисел от 1 до n, на которой быстрая сортировка выполнит максимальное

число сравнений. Если таких перестановок несколько, вывести любую из них.

Пример

antiqs.in	antiqs.out
3	1 3 2

Задача G. Сортировка за линейное время

Имя входного файла:	`buckets.in`
Имя выходного файла:	`buckets.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан массив случайных целых чисел, нужно отсортировать его.

Формат входных данных

На первой строке количество тестов t ($1 \leqslant t \leqslant 200$) и число n ($1 \leqslant n \leqslant 100\,000$) — размер массива в каждом из тестов. На второй строке пара целых чисел a, b от 1 до 10^9, используемая в генераторе случайных чисел.

```
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до 2^24 - 1.
5. }
6. unsigned int nextRand32() {
7.     unsigned int a = nextRand24(), b = nextRand24();
8.     return (a << 8) ^ b; // число от 0 до 2^32 - 1.
9. }
```

Тесты генерируются последовательно.

Элементы массива генерируются последовательно. `x[i] = nextRand32();`

Формат выходных данных

Для каждого теста выведите на отдельной строке $\left(\sum_{i=1}^{n} x_i \cdot i\right) \bmod 2^{64}$.

Примеры

buckets.in	buckets.out
1 6 239 13	46062181379

Замечание

Сгенерированный массив: $12, 130926, 3941054950, 2013898548, 197852696, 2753287507$.

В этой задаче очень небольшой запас по времени. Если она не сдается, это нормально.

Задача H. Количество инверсий

Имя входного файла:	`invcnt.in`
Имя выходного файла:	`invcnt.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан массив случайных целых чисел, нужно найти количество инверсий.
Инверсия — пара $i < j$ такая, что $x_i > x_j$.

Формат входных данных

На первой строке числа n ($1 \leqslant n \leqslant 1\,000\,000$) — размер массива и m ($1 \leqslant m \leqslant 2^{24}$ числа в массиве от 0 до $m - 1$). На второй строке пара целых чисел a, b от 1 до 10^9, используемая в генераторе случайных чисел.

```
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до 2^24 - 1.
5. }
```

Элементы массива генерируются последовательно. x_i `= nextRand24() % m;`

Формат выходных данных

Выведите количество инверсий.

Примеры

invcnt.in	invcnt.out
20 5 19 18	63

Замечание

Сгенерированный массив: {0, 1, 1, 4, 2, 2, 1, 0, 4, 2, 4, 0, 3, 1, 3, 4, 3, 3, 3, 0}.

Задача I. Супермаркет

Имя входного файла:	`supermarket.in`
Имя выходного файла:	`supermarket.out`
Ограничение по времени:	2 секунда
Ограничение по памяти:	256 мегабайт

Антон и Адам решили устроить чаепитие и заразили своей идеей еще $n-2$ своих друзей.

Они собрались и выбрали в одном довольно большом супермаркете p тортиков. Настал черед расплатиться за них. В магазине есть m касс, занумерованных числами от 1 до m. Про i-ю кассу известно, что кассиру требуется a_i единиц времени на обработку одного товара и b_i единиц времени для того, чтобы рассчитаться с покупателем. Обойдя все кассы, студенты посчитали, что на обслуживание покупателей, уже стоящих в i-ю кассу, уйдет t_i единиц времени.

Теперь Антон и Адам задались вопросом, в какие кассы надо встать им и их друзьям (в каждую из выбранных касс должен стоять хотя бы один из них, и каждый из них может стоять не более, чем в одну кассу, поэтому суммарно они могут стоять не более чем в n касс) и сколько тортиков каждый должен взять, чтобы последний из них вышел из магазина как можно раньше. Некоторые из ребят могут в кассу не стоять, а, отдав все тортики другим, выйти через специальный выход для тех, кто ничего не купил.

Напишите программу, которая определит это минимальное время.

Формат входных данных

В первой строке записано одно число m — количество касс в супермаркете ($1 \leqslant m \leqslant 10^5$). В следующих m строках записано по три числа a_i, b_i, t_i ($0 \leqslant a_i, b_i, t_i \leqslant 10^5$). В последней строке записаны два числа — n и p — число студентов и покупок у них соответственно ($0 \leqslant p \leqslant 10^5$; $2 \leqslant n \leqslant 10^5$).

Все числа во входном файле целые.

Формат выходных данных

Выведите минимальное время выхода последнего студента из магазина.

Примеры

supermarket.in	supermarket.out
2 100 10 40 10 100 50 2 2	160
3 1 2 0 5 2 1 2 10 1 3 5	7

Задача J. Отрезки с большой суммой

Имя входного файла:	`bigseg.in`
Имя выходного файла:	`bigseg.out`
Ограничение по времени:	2.5 секунды
Ограничение по памяти:	256 мегабайт

Вам задан массив $a_1, a_2, \ldots, a_n$. Найдите число отрезков с суммой не меньше k: число пар (l, r), что $l \leqslant r$ и $\left(\sum_{i=l}^{r} a[i]\right) \geqslant k$.

Формат входных данных

В первой строке заданы два целых числа n и k ($1 \leqslant n \leqslant 3 \cdot 10^6$; $-10^{15} \leqslant k \leqslant 10^{15}$).

На второй строке пара целых чисел a, b от 1 до 10^9, используемая в генераторе случайных чисел.

```
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до 2^24 - 1.
5. }
6. unsigned int nextRand32() {
7.     unsigned int a = nextRand24(), b = nextRand24();
8.     return (a << 8) ^ b; // число от 0 до 2^32 - 1.
9. }
```

Элементы массива генерируются последовательно, приводя беззнаковый тип к знаковому с дополнительным кодом. a_i `= (int) nextRand32();`

Формат выходных данных

Выведите число искомых отрезков.

Примеры

bigseg.in	bigseg.out
4 777 3138 3139	6

Замечание

Массив из первого примера выглядит так: `{39513, 844135560, 447482473, -1948332165}`

Приведение беззнакового числа к знаковому с дополнительным кодом происходит так: число u от 0 до $2^{32} - 1$ приводится к числу v от -2^{31} до $2^{31} - 1$, если $u \equiv v \pmod{2^{32}}$.

Если у вас не получается решить эту задачу, подумайте, как решить задачу в случае $k = 0$.

Задача K. Поиск в большом массиве

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Это интерактивная задача.

Рассмотрим массив $a[1..n]$ длины $n = 10^{18}$ такой, что $a[i] = i$. Из массива a удалили m элементов и сделали циклический сдвиг, получился массив b.

Вам задано число x, найдите такой i, что $b[i] = x$. У вас есть не более 10 запросов.

Протокол взаимодействия

Сначала вашей программе подаётся на вход в отдельной строке два числа: x ($1 \leqslant x \leqslant 10^{18}$) и m — число удаленных элементов ($0 \leqslant m \leqslant 500$).

После этого ваша программа может делать запросы: «посмотреть, чему равен $b[v]$». Для этого нужно вывести в выходной поток на отдельной строке «`? v`» ($1 \leqslant v \leqslant 10^{18} - m$). В ответ на запрос во входном потоке на отдельной строке будет записано одно число — $b[v]$. Вы можете сделать не более десяти таких запросов, иначе ваша программа получит вердикт «`Wrong answer`».

В конце вы должны вывести в выходной поток «`! i`», где i такая позиция, что $b[i] = x$. Если такого i не существует, то следует вместо номера ячейки вывести -1, таким образом последнее сообщение должно быть «`! -1`».

Замечание

После каждого действия вашей программы выводите символ перевода строки. Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java или «`print`» в Python, сброс потока вывода у вас происходит автоматически, дополнительно делать «`flush`» не обязательно. Если вы используете другой способ вывода, рекомендуется делать «`flush`», но все равно обязательно требуется выводить символ перевода строки.

Ниже приведены наиболее типичные причины получения тех или иных сообщений об ошибке.

Если ваша программа соблюдает протокол, но неверно определяет искомый номер ячейки либо выполняет слишком много запросов, вы получите результат «Wrong Answer».

Если ваша программа выводит некорректно отформатированные сообщения интерактору, то вы получите результат «Presentation Error» либо «Wrong Answer».

Если ваша программа нарушила протокол и ждет ввода в то же время, когда его ждет и интерактор, то вы получите результат «Idleness Limit Exceeded». Обратите внимание, что к такому же результату может привести и то, что вы не переводите строку после каждого выведенного сообщения или выводите не тем способом, который описан в начале раздела, и не делаете «`flush`».

Пример

стандартный ввод	стандартный вывод
16 3 10 12 13 16	 ? 1 ? 2 ? 3 ? 4 ! 4

Замечание

В примере из массива были удалены элементы 11, 14 и 15, и массив был циклически сдвинут так, что первый элемент стал 10.

Задача L. K-Best

Имя входного файла:	`kbest.in`
Имя выходного файла:	`kbest.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

У Демьяны есть n драгоценностей. Каждая из драгоценностей имеет ценность v_i и вес w_i. С тех пор, как её мужа Джонни уволили в связи с последним финансовым кризисом, Демьяна решила продать несколько драгоценностей. Для себя она решила оставить лишь k лучших. Лучших в смысле максимизации достаточно специфического выражения: пусть она оставила для себя драгоценности номер $i_1, i_2, \ldots, i_k$, тогда максимальной должна быть величина

$$\frac{\sum\limits_{j=1}^{k} v_{i_j}}{\sum\limits_{j=1}^{k} w_{i_j}}$$

Помогите Демьяне выбрать k драгоценностей требуемым образом.

Формат входных данных

На первой строке n и k ($1 \leqslant k \leqslant n \leqslant 100\,000$).

Следующие n строк содержат пары целых чисел v_i, w_i ($0 \leqslant v_i \leqslant 10^6, 1 \leqslant w_i \leqslant 10^6$, сумма всех v_i не превосходит 10^7, сумма всех w_i также не превосходит 10^7).

Формат выходных данных

Выведите k различных чисел от 1 до n — номера драгоценностей. Драгоценности нумеруются в том порядке, в котором перечислены во входных данных. Если есть несколько оптимальных ответов, выведите любой.

Пример

kbest.in	kbest.out
3 2 1 1 1 2 1 3	1 2

Задача M. Проверьте сортирующую сеть

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	

Проверьте является ли сеть из n проводов сортирующей.

Формат входных данных

В первой строке входного файла содержится три числа n —количество проводов, m —количество компараторов в сети и k —количество слоев в сети ($1 \leqslant n \leqslant 15$, $0 \leqslant m, k \leqslant 150$). В каждой из следующих строк содержится описание слоя из компараторов: число r — количество компараторов в слое и далее r пар чисел, номера проводов, которые сравнивает компаратор. Внутри слоя все номера проводов различны.

Формат выходных данных

Выведите «Yes», если сеть является сортирующей и «No», если нет.

Пример

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

Задача N. Постройте сортирующую сеть

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	

Постройте сортирующую сеть для n проводов.

Формат входных данных

В первой строке входного файла находится одно число n ($1 \leqslant n \leqslant 16$) — требуемый размер сортирующей сети.

Формат выходных данных

В первую строку выходного файла выведите три числа n —количество проводов, m —количество компараторов в сети и k —количество слоев в сети. В каждой из следующих строк выведите описание слоя из компараторов, число r — количество компараторов в слое и далее r пар чисел, номера проводов, которые сравнивает компаратор. Внутри слоя все номера проводов должны быть различны. Число слоев не должно превышать 12.

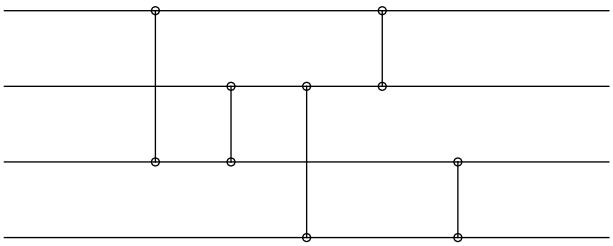
Пример

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

Задача O. Почти сортирующая сеть

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам задано несколько 0-1 последовательностей. Для каждой последовательности постройте сеть компараторов, которая сортирует все 0-1 последовательности, кроме заданной, или скажите, что такой не существует.



Сеть компараторов на картинке не является сортирующей, она не сортирует последовательность $[1, 0, 1, 0]$. Оказывается, $[1, 0, 1, 0]$ — единственная 0-1 последовательность, которая не сортируется этой сетью.

Формат входных данных

Входные данные состоят из нескольких тестов.

Каждый тест начинается с целого числа n ($2 \leqslant n \leqslant 10$) — число проводов. Далее следует n целых чисел $a_1, a_2, \ldots, a_n$ ($a_i \in \{0, 1\}$).

После последнего теста вводится $n = 0$, означающий конец входных данных.

Формат выходных данных

Для каждого теста выведите "-1", если не существует сети компараторов, сортирующей все 0-1 последовательности, кроме $[a_1, a_2, \ldots, a_n]$. В противном случае выведите описано такой сети. Оно должно состоять из целого числа m — количество компараторов ($0 \leqslant m \leqslant 1000$). Далее описываются компараторы. Каждый из них задается номерами ниток, которые он соединяет.

Гарантируется, что если такая сеть существует, то существует и сеть, в которой не более 100 компараторов (однако вы можете использовать до 1000).

Примеры

стандартный ввод	стандартный вывод
4 1 0 1 0 4 1 1 1 1 0	5 1 3 2 3 2 4 1 2 3 4 -1