**B. Технический долг**

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Реализация большого проекта — очень сложная задача, и при разработке программист Алексей руководствуется следующим принципом: сначала написать работающий прототип, а потом улучшать код. Чтобы не забыть, что именно отложено на потом, на каждый такой долг Алексей заводит на себя задачу в специальной системе Yagile.

Система устроена следующим образом: для каждой задачи задается дедлайн — день ti. Если задача не решена до этого момента времени, то в задачу приходит робот и пишет комментарий о том, что задачу надобно закрыть. Если через X дней задача не решена, то робот приходит снова. Так продолжается до тех пор, пока задача не будет решена.

Алексей каждый день заходит в Yagile и видит сообщения от робота. Если Алексей не хочет приступать к решению накопленных задач, то он прочитывает все сообщения с помощью одной кнопки и занимается другими делами. Однако Алексей понимает, что так долго делать нельзя, поэтому он разрешает себе нажимать на эту кнопку ровно K−1 раз, а на K-й раз садится и решает все задачи разом (даже те, у которых не настал дедлайн).

Определите день, когда Алексей закроет все задачи.

**Формат ввода**

Первая строка содержит три целых числа N (1≤N≤105) — количество накопленных задач, X (1≤X≤109) — количество дней, через которое приходит робот и число K из условия (1≤K≤109).

Вторая строка содержит N целых чисел t1, t2, …, tN (1≤ti≤109) — дедлайны соответствующих задач.

**Формат вывода**

Выведите одно число — день, когда Алексей закроет все задачи.

**Пример 1**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 6 5 10  1 2 3 4 5 6 | 10 |

**Пример 2**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 5 7 12  5 22 17 13 8 | 27 |

**Примечания**

Во втором примере Алексей завёл 5 задач, и каждые 7 дней, начиная с дня дедлайна, робот оставляет комментарий. Например, в первой задаче робот будет оставлять комментарии в дни 5, 12, 19, 26, 33, и т. д. Если рассмотреть все комментарии робота по задачам Алексея, то они будут написаны в следующие дни: 5, 8, 12, 13, 15, 17, 19, 20, 22 (во 2-й и 5-й задачах одновременно), 24, 26, 27, 29, … На 12-й по счёту день Алексей закроет все задачи, чему соответствует день 27.

**C. Сапёр**

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Программист Петя очень любит играть в игру «Сапёр». Но он не хочет тратить время впустую, поэтому согласен играть, только если для победы нужно сделать очень мало ходов. Он попросил вас написать программу, которая по положению мин определяет, сколько ходов нужно сделать Пете, чтобы определить местоположение всех мин на поле.

Игра устроена следующим образом: когда Петя открывает клетку на поле, автоматически проверяются клетки сверху, снизу, слева и справа от открытой. Если какая-либо из них не содержит мин, она тоже считается открытой и автоматически проверяются её соседи. Петя считается достаточно опытным игроком, поэтому он почти наверняка будет каждый раз открывать пустую клетку, которая не была открыта до этого.

**Формат ввода**

В первой строке входных данных записаны два целых положительных числа n и m (1≤n,m≤100) — размеры поля.

В следующей строке дано целое неотрицательное число k (0≤k≤n⋅m) — количество мин на поле. В следующих k строках даны целочисленные координаты каждой из мин: номер строки r и столбца (1≤r≤n, 1≤c≤m).

**Формат вывода**

В единственной строке выведите одно число, равное количеству необходимых Пете ходов, чтобы открыть все пустые клетки на поле.

**Пример 1**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 3 3  3  2 1  2 2  2 3 | 2 |

**Пример 2**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 3 3  2  2 1  2 2 |  |

**D. Матрица поворота**

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Преобразование Фурье больше не отвечает современным потребностям. Поэтому отдел разработки новых алгоритмов сжатия придумал новое линейное ортогональное преобразование исходных данных, которое улучшает показатель сжатия. К сожалению, последняя строка матрицы этого преобразования утеряна, нужно её восстановить.

Напомним, что линейное преобразование называется ортогональным, если оно сохраняет длину векторов. Кроме того, известно, что исходное преобразование имело определитель равный единице.

**Формат ввода**

В первой строке входных данных записано число n (2≤n≤16) — размер матрицы. В следующих n−1 строках записаны по n чисел в каждой — первые N−1 строк матрицы преобразования. Элементы матрицы записаны с точностью 10−12.

Гарантируется, что решение всегда существует.

**Формат вывода**

Выведите N чисел — последнюю строку матрицы. Ответ будет засчитан, если относительная или абсолютная погрешность каждого из чисел не превосходит 10−6.

Если линейных ортогональных преобразований, удовлетворяющих условию задачи, несколько, то выведите любое из них.

**Пример 1**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 2  0 1 | -1.000000000000 0.000000000000 |

**Пример 2**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 3  0.6666666666666 0.6666666666666 0.3333333333333  -0.3333333333333 0.6666666666666 -0.6666666666666 | -0.666666666667 0.333333333333 0.66666 |

**E. Игра на дереве**

|  |  |
| --- | --- |
| Ограничение времени | 2 секунды |
| Ограничение памяти | 256Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Есть дерево из n вершин, корнем которого является вершина 1. Два игрока играют в следующую игру. На корне дерева стоит фишка. Игроки ходят по очереди, и на своём ходу игрок должен передвинуть фишку на одного из потомков текущей вершины. Каждый лист дерева является выигрышным, проигрышным или нейтральным. Если фишка находится на листе, тот игрок, который её только что туда передвинул, заканчивает игру с результатом, который указан на этом листе.

Если фишка находится на расстоянии k от корня и не на листе, игрок может сказать «реверс» перед тем, как её двигать, — тогда все результаты на листьях изменятся на противоположные. Обратите внимание, что «реверс» может быть сказан не более одного раза за игру.

Как закончится игра, если оба игрока действуют оптимально?

**Формат ввода**

В первой строке записаны два числа n и k (2≤n≤5000,0≤k<n) — количество вершин и глубина, на которой можно сделать «реверс».

Далее записана строка из n символов. i-й символ этой строки равен «+», если i-я вершина является листом и выигрышная, «-», если i-я вершина является листом и проигрышная, «0», если i-я вершина является листом и нейтральная, и «.» в противном случае (вершина не является листом).

Далее записаны n−1 строк, в каждой по два числа ui, vi (1≤ui,vi≤n) — ребра дерева.

**Формат вывода**

Выведите First, Second или Draw, в зависимости от того, кто выиграет.

**Пример 1**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 8 1  ..-0..-+  1 2  2 3  2 4  1 5  5 6  6 7  6 8 | First |

**Пример 2**

| **Ввод** |
| --- |

Вывод

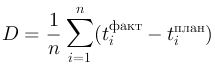
|  |  |
| --- | --- |
| 8 1  ..-0..0+  1 2  2 3  2 4  1 5  5 6  6 7  6 8 | Draw |

**F. Распределение курьеров по зонам доставки**

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Сервис Яндекс.Еда доставляет пользователям заказы из ресторанов. Доставку выполняют курьеры, которые работают сменами по несколько часов. Для распределения курьеров по городу, весь город разбивается на квадратные геозоны размером 500 × 500 метров. Из этих «кирпичиков» строятся прямоугольные геозоны доставки, а уже потом курьеры распределяются по этим прямоугольникам и выполняют доставку в рамках геозоны.

Яндекс.Еде важно доставить заказ в срок, обещанный пользователю при заказе. Чтобы оценить насколько хорошо работает доставка, аналитики каждый день считают среднее отклонение фактического времени доставки от планируемого по всем заказам за этот день:



Здесь: n — число заказов в день,  — фактическое время доставки i-ого заказа,  — планируемое время доставки i-ого заказа. Если этот показатель больше нуля, то в среднем курьеры опаздывают, если меньше нуля  — доставляют вовремя или раньше.

С ростом сервиса растёт количество новых курьеров. Для улучшения качества доставки, при выводе нового курьера используется следующий алгоритм назначения новому курьеру зоны доставки:   
1. Для каждой квадратной геозоны 500 × 500 метров рассчитывается показатель отклонения времени доставки D в этой зоне за день   
2. В качестве зоны доставки назначается прямоугольная зона, состоящая не более, чем из K квадратных геозон 500 × 500, сумма значений D которых максимальна.

Реализуйте алгоритм поиска зоны доставки для нового курьера на основании данных о временах отклонения фактических времён доставки от планируемых D по зонам 500 × 500 метров. Данные о временах отклонения заданы матрицей размера M × N.

**Формат ввода**

В первой строке указан параметр K со значением из диапазона от 1 до 1000.

Во второй строке через пробел указаны размеры матрицы M (1 ≤ M ≤ 1000) и N (1 ≤ N ≤ 100) соответственно.

Далее идут M строк, каждая из которых содержит N целых чисел, записанных через пробел — параметры D для геозон 500 × 500 метров.

**Формат вывода**

Выведите одно число — сумму значений D в найденной зоне (по описанному выше алгоритму) для нового курьера.

**Пример**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 4  2 3  4 -1 -3  -2 4 -3 | 5 |

**G. Бонусы для водителей**

|  |  |
| --- | --- |
| Ограничение времени | 1 секунда |
| Ограничение памяти | 64Mb |
| Ввод | стандартный ввод или input.txt |
| Вывод | стандартный вывод или output.txt |

Водители Яндекс.Такси узнали о раздаче бонусов и выстроились перед офисом. У каждого водителя есть рейтинг. Необходимо раздать водителям бонусы, соблюдая следующие условия:

• Сумма бонуса кратна 500 рублям.

• Каждый водитель должен получить как минимум 500 рублей.

• Водитель с бóльшим рейтингом должен получить бóльшую сумму бонуса, чем его соседи слева или справа с меньшим рейтингом.

Какое минимальное количество денег потребуется на бонусы?

**Формат ввода**

На первой строчке записано число N (1 <= N <= 20000), далее следует N строчек с рейтингами водителей Rn (0 <= Rn < 4096)

**Формат вывода**

Ответ должен содержать минимально необходимое количество денег для выплаты вознаграждений

**Пример 1**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 4  1  2  3  4 | 5000 |

**Пример 2**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 4  5  5  5  5 | 2000 |

**Пример 3**

| **Ввод** |
| --- |

Вывод

|  |  |
| --- | --- |
| 4  4  2  3  3 | 3000 |