Оглавление

Кузнечик на линейке

Кузнечик находится в точке 0 числовой прямой. Он может совершать два типа прыжков:

- Прыжок вправо на 3 единицы
- Прыжок влево на 2 единицы

Найти минимальное количество прыжков для достижения точки D, где $-1000 \leq D \leq 1000$.

Пусть a — количество прыжков вправо ($+3$), b — количество прыжков влево (-2). Тогда:

$$3a - 2b = D, \quad a, b \geq 0$$

Обозначим $n = a + b$ — общее количество прыжков. Так как $b = n - a$, подставляем:

$$3a - 2(n - a) = D \ \Rightarrow \ 5a - 2n = D \ \Rightarrow \ 5a = D + 2n$$

Заметим, что a должно быть целым и $0 \leq a \leq n$. Тогда для заданного D достаточно: перебирать n от 0 до некоторого момента и проверять, делится ли $D + 2n$ на 5. Если делится, вычислять $a = \frac{D+2n}{5}$ и проверять условие $0 \leq a \leq n$. Первое найденное такое n будет ответом.

Так как $|D| \leq 1000$, а прыжки имеют длины 2 и 3, ответ всегда будет конечным. Можно ограничить перебор $n \leq 10000$, этого достаточно для данных ограничений.

Сплошная защита

Дана строка длины n, состоящая из символов 'B' (целый блок) и 'W' (повреждённый блок). Необходимо найти минимальное количество замен 'W' на 'B', чтобы в строке появился непрерывный отрезок из ровно k подряд идущих символов 'B'.

Определим массив a, где $a_i = 1$, если i-й символ строки равен 'W' (блок повреждён), и $a_i = 0$, если символ равен 'B' (блок целый). Для фиксированного отрезка $[i, i+k-1]$ количество повреждённых блоков равно:

$$S_i = a_i + a_{i+1} + \cdots + a_{i+k-1}$$

Чтобы сделать этот отрезок полностью целым, нужно заменить все повреждённые блоки, то есть выполнить S_i замен. Таким образом, задача сводится к нахождению минимума суммы S_i по всем i от 0 до $n - k$.

Для эффективного решения используем метод скользящего окна: вычисляем сумму первых k элементов массива a: $S_0 = a_0 + a_1 + \cdots + a_{k-1}$. Для каждого следующего i от 1 до $n - k$ обновляем сумму: $S_i = S_{i-1} - a_{i-1} + a_{i+k-1}$. Находим минимальное значение среди всех S_i - это значение и будет ответом.

Игра с монеткой

Дана строка s длины n, состоящая из строчных латинских букв. Игра начинается с первой буквы строки (позиция 1). На каждом шаге:

- Если текущая буква встречается в строке ещё хотя бы один раз (кроме текущей позиции), можно переместить монетку на любую другую позицию с такой же буквой.
- Иначе (если текущая буква уникальна в строке), нужно переместить монетку на следующую букву (если она существует).

Игра завершается, когда монетка должна перейти на следующую букву, но её не существует. Необходимо определить, завершается ли игра для данной строки.

Ключевое наблюдение: игра может продолжаться бесконечно тогда и только тогда, когда в строке существует хотя бы одна буква, которая встречается более одного раза. В этом случае игрок может выбрать стратегию, при которой он постоянно перемещается между позициями с этой буквой, избегая перехода к уникальным буквам, которые вынуждают двигаться к концу строки.

Если же все буквы в строке уникальны (каждая буква встречается ровно один раз), то на каждом шаге игрок вынужден переходить к следующей букве, и игра гарантированно завершится после n шагов.

Таким образом, для решения нужно подсчитать частоту каждой буквы в строке. Если все буквы имеют частоту 1, игра завершится — ответ `YES`. Если существует хотя бы одна буква с частотой больше 1, игра может продолжаться бесконечно — ответ `NO`.

Посчитай отрезки

Дан массив A из n положительных целых чисел. Необходимо найти минимальную длину подряд идущих элементов, сумма которых больше заданного числа k. Если такого отрезка не существует, вывести 0.

Пусть $S(l, r) = A_l + A_{l+1} + \cdots + A_r$ — сумма элементов на отрезке $[l, r]$. Требуется найти минимальное значение $(r - l + 1)$ такое, что $S(l, r) > k$ для некоторых $l \leq r$.

Для решения задачи используем метод двух указателей: инициализируем два указателя: $l = 0$ и $r = 0$, а также текущую сумму $sum = A_0$. Будем двигать правый указатель r вправо, увеличивая сумму, пока она не станет больше k. Как только сумма стала больше k, фиксируем длину текущего отрезка $(r-l+1)$ и пытаемся уменьшить её, сдвигая левый указатель l вправо, пока сумма остаётся больше k. После каждого увеличения r повторяем процесс: если сумма стала больше k, пытаемся сдвинуть l для уменьшения длины отрезка. Минимальная найденная длина и будет ответом.

Спуск груза

Дана последовательность из n башен с высотами $h_1, h_2, \ldots, h_n$. На башне с номером m расположен груз, который необходимо спустить на землю. Доступны следующие правила перемещения:

1. С грузом можно только спускаться (переходить от башни большей или равной высоты к соседней башне меньшей или равной высоты).

2. Без груза можно перемещаться свободно между вершинами соседних башен.

3. Чтобы подняться на любую башню с земли, необходима лестница длиной не меньше высоты этой башни.

Необходимо найти минимальную длину лестницы, достаточную для спуска груза на землю.

Заметим, что для спуска груза нужно найти такую точку старта, до которой можно непрерывно спускаться, начиная в точке m. Рассмотрим два возможных направления спуска: влево и вправо. В каждом направлении ищем ближайшую "точку перелома позицию на которой заканчивается непрерывный спуск.

Обозначим:

- l — наименьший индекс такой, что все башни на отрезке $[l, m]$ образуют неубывающую последовательность к m, то есть $h_i \leq h_{i+1}$ для $i = l, l+1, \ldots, m-1$

- r — наибольший индекс такой, что все башни на отрезке $[m, r]$ образуют невозрастающую последовательность от m, то есть $h_{i+1} \leq h_i$ для $i = m, m+1, \ldots, r-1$

До позиций l и r можно начать непрерывный спуск от точки m. Тогда минимальная необходимая длина лестницы - это минимум из высот h_l и h_r, так как именно на такую высоту нужно подняться, чтобы потом был возможен спуск.

Анализ аллергенов

Имеется n различных химических веществ. Для эксперимента выбрано подмножество из m веществ. Известно, что у лаборанта аллергия на k конкретных веществ, и при проведении эксперимента аллергической реакции не возникло. Для p учёных, у каждого из которых есть свой набор аллергенных веществ, необходимо определить возможность аллергической реакции на сегодняшний эксперимент.

Пусть:

- A — множество всех n веществ.
- E — множество веществ, использованных в эксперименте ($|E| = m$).
- L — множество веществ, на которые у лаборанта аллергия ($|L| = k$).
- S_i — множество веществ, на которые у i-го учёного аллергия.

Из условия известно, что эксперимент прошёл без аллергической реакции у лаборанта, значит: $E \cap L = \emptyset$. То есть ни одно из веществ, на которые у лаборанта аллергия, не входит в эксперимент.

Для каждого учёного нужно определить:

- NO — если эксперимент гарантированно безопасен: $E \cap S_i = \emptyset$
- YES — если эксперимент гарантированно вызовет аллергию: существует вещество $x \in E$ такое, что $x \in S_i$
- MAYBE — если ситуацию точно определить нельзя

Из условия $E \cap L = \emptyset$ следует, что все вещества из L гарантированно не входят в эксперимент. Однако о других веществах ($A \setminus L$) ничего неизвестно — они могут как входить, так и не входить в эксперимент.

Рассмотрим возможные случаи для i-го учёного:

- Если $S_i \subseteq L$, то все аллергены учёного гарантированно не входят в эксперимент, так как они содержатся в L, а $E \cap L = \emptyset$. Ответ `NO`.
- Если существует вещество $x \in S_i$ такое, что $x \notin L$, то это вещество потенциально может входить в эксперимент. Заметим, что эксперимент содержит m веществ, ни одно из которых не принадлежит L. Всего веществ, не принадлежащих L, имеется $n - k$.

  Следовательно, максимальное количество веществ в эксперименте, которые могут вызвать аллергию у учёного, равно $\min(m, |S_i \setminus L|)$. Если $|S_i \setminus L| > n - k - m$, то хотя бы одно вещество из $S_i \setminus L$ гарантированно попало в эксперимент, так как недостаточно "свободных"веществ для формирования эксперимента без пересечения с S_i. В этом случае ответ `YES`. Иначе ответ `MAYBE`, так как возможно как наличие, так и отсутствие аллергенов учёного в эксперименте.

Система передачи

Рассматривается конвейерная система из $3n$ позиций. Устройства расположены над позициями $n + 1$ до $2n$. Каждое устройство i отправляет пакет устройству a_i ($a_i \neq i$). Изначально пакет находится под отправителем на позиции $n + i$. Конвейер может двигаться влево или вправо на одну позицию за одно нажатие. Все пакеты одновременно сдвигаются с конвейером. Пакет доставлен, когда он оказывается под устройством-получателем. Необходимо найти минимальное количество нажатий для доставки всех пакетов.

Для каждого устройства i вычисляем необходимое перемещение пакета: $|a_i - i|$. Поскольку все пакеты движутся вместе с конвейером, определяем максимальные необходимые смещения:

$$L = \max_{\substack{1 \le i \le n \\ a_i < i}} (i - a_i), \quad R = \max_{\substack{1 \le i \le n \\ a_i > i}} (a_i - i)$$

При отсутствии пакетов, требующих движения в каком-либо направлении, соответствующая величина считается равной 0. После определения L и R, минимальное количество нажатий вычисляется как:

$$\min(L + 2R,\ 2L + R)$$

Объяснение:

- Если преобладает движение вправо ($R > L$), оптимально начать с движения вправо, потом вернуться влево: $L + 2R$
- Если преобладает движение влево ($L > R$), оптимально начать с движения влево, потом вернуться вправо: $2L + R$

Лабиринт из парных элементов

Дан массив из n элементов, где каждый элемент встречается ровно два раза. Начинаем слева от первого элемента, нужно выйти справа от последнего элемента. За один ход можно "телепортироваться"к парному элементу: если находимся слева от элемента i, попадаем между его парным элементом j и элементом $j+1$; если справа от элемента i, попадаем между его парным элементом j и элементом $j-1$. Необходимо найти минимальное количество переходов для выхода.

Пронумеруем позиции между элементами массива от 0 до n, где позиция 0 — слева от первого элемента, позиция n — справа от последнего элемента (целевая позиция).

Для каждого элемента i (в 1-индексации) определим его парный элемент p_i, где $p_i \neq i$ и $a_i = a_{p_i}$.

Из каждой позиции i (кроме крайних) можно смотреть на два соседних элемента:

- Если $0 \leq i \leq n-1$, то слева от элемента a_{i+1}
- Если $1 \leq i \leq n$, то справа от элемента a_i

Таким образом, из позиции i (где $1 \leq i \leq n-1$) возможны **два перехода**:

1. Рассматривая элемент a_i (справа от него):
   - Если $p_i > i$, переход в позицию $p_i - 1$
   - Если $p_i < i$, переход в позицию p_i
2. Рассматривая элемент a_{i+1} (слева от него):
   - Если $p_{i+1} > i + 1$, переход в позицию p_{i+1}
   - Если $p_{i+1} < i + 1$, переход в позицию $p_{i+1} - 1$

Из крайних позиций 0 и n возможен только один переход:

- Из 0: только рассматривая элемент a_1 (слева от него)
- Из n: только рассматривая элемент a_n (справа от него)

Теперь задача сводится к поиску кратчайшего пути в ориентированном графе, где вершины — позиции $0..n$, а из каждой внутренней вершины i исходит два ребра. Ключевое наблюдение: Хотя из каждой внутренней позиции доступно два варианта, оптимальный путь всегда будет использовать переход, ведущий вправо.

Доказательство:

Пусть из позиции i доступны переходы влево и вправо. Рассмотрим переход влево (точку x) - мы оказываемся в позиции, которую уже посещали ранее (так как двигались вправо до достижения i). Из позиции x снова доступны два перехода. Чтобы продолжить путь к n, необходимо вновь достичь i (либо же пойти еще раз влево и оказаться в уже посещенной точке x' и т д), а для этого потребуется как минимум один дополнительный переход по сравнению с прямым переходом из i в y. Таким образом, переход влево никогда не может быть частью оптимального пути. Таким образом, на каждом шаге оптимально выбирать переход, ведущий вправо.

Для реализации алгоритма можно построить массив переходов go, где go_i содержит следующую позицию при нахождении рядом с элементом на позиции $i + 1$, а затем просто следовать по этим переходам от начальной позиции до достижения конца.

Поиск слова

Дана матрица размером $n \times m$, состоящая из заглавных латинских букв. В матрице ровно один раз встречается слово "HOTEL причём буквы этого слова расположены в соседних по стороне клетках. После слова "HOTEL"(начиная с последней буквы "L") тем же шаблоном смещений записано название отеля — слово из 5 букв, начинающееся на "L". Необходимо найти название отеля.

Перебираем все клетки матрицы (i, j) - потенциальные начала слова "HOTEL"и составляем всевозможные слова из пяти букв, которые можно получить. Если есть последовательность переходов, дающая слово "HOTEL то запоминаем направления, которые приводят к этому.

После нахождения слова "HOTEL"и определения начальной клетки с набором переходов, мы знаем координаты последней буквы "L". Из этой клетки продолжаем движение в тех же направлениях, собирая буквы. Полученные 5 букв и будут названием отеля.

Оптимальное распределение

Имеется n объектов, расположенных в ряд. Каждый объект i имеет базовую ценность a_i. Некоторые пары соседних объектов конфликтуют: если объекты i и $i + 1$ оба активны, возникает конфликт стоимостью q_i. Можно деактивировать некоторые объекты. Деактивированный объект не приносит ценность и не участвует в конфликтах. Необходимо найти максимальную суммарную ценность.

Обозначим g_i как стоимость конфликта между объектами i и $i + 1$, где $g_i = 0$, если конфликта нет.

Состояние можно описывать динамическим программированием. Пусть $dp[i][0]$ — максимальная ценность для первых i объектов, если объект i деактивирован, и $dp[i][1]$ — если объект i активен.

Базовые случаи:

- $dp[1][0] = 0$ (первый объект деактивирован)
- $dp[1][1] = a_1$ (первый объект активен)

Переходы для $i \geq 2$:

- Если объект i деактивирован ($dp[i][0]$), то предыдущий объект может быть как активен, так и деактивирован:
$$dp[i][0] = \max(dp[i-1][0], dp[i-1][1])$$
- Если объект i активен ($dp[i][1]$), то:
  - Если предыдущий объект деактивирован: $dp[i-1][0] + a_i$
  - Если предыдущий объект активен: $dp[i-1][1] + a_i - g_{i-1}$ (вычитаем стоимость конфликта)

$$dp[i][1] = \max(dp[i-1][0] + a_i, dp[i-1][1] + a_i - g_{i-1})$$

Ответом будет $\max(dp[n][0], dp[n][1])$.

Включенные лампочки

Имеется n лампочек, изначально все включены. Выполняются n операций: для каждого i от 1 до n меняется состояние всех лампочек с номерами, кратными i. После всех операций некоторые лампочки остаются включёнными. Для заданного k нужно найти минимальное n, при котором ровно k лампочек останутся включёнными.

Рассмотрим лампочку с номером x. Её состояние меняется при каждой операции i, где i — делитель x. Количество таких операций равно количеству делителей $d(x)$. Начальное состояние — включено, поэтому лампочка останется включённой, если $d(x)$ чётно.

Количество делителей числа нечётно только для полных квадратов. Поэтому: если x — полный квадрат, то $d(x)$ нечётно, лампочка выключится; если x — не квадрат, то $d(x)$ чётно, лампочка останется включённой.

Таким образом, после всех операций включёнными останутся только лампочки с номерами, не являющимися полными квадратами. Количество таких лампочек для n равно:

$$f(n) = n - \lfloor \sqrt{n} \rfloor$$

где $\lfloor \sqrt{n} \rfloor$ — количество полных квадратов от 1 до n.

Теперь задача: для заданного k найти минимальное n, при котором $f(n) \geq k$. Заметим, что функция $f(n)$ монотонно возрастает. Поэтому минимальное такое n можно найти бинарным поиском.

Пары точек

Дан массив A размера n, состоящий из целых чисел. Необходимо посчитать количество пар индексов (i, j) таких, что $i \neq j$ и $|a_i - a_j| \leq k$.

Для решения задачи отсортируем массив A по возрастанию. После сортировки условие $|a_i - a_j| \leq k$ для $i < j$ эквивалентно $a_j \leq a_i + k$.

Используем метод двух указателей. Зафиксируем левый указатель i. Найдём максимальный индекс r такой, что $a_r \leq a_i + k$. Тогда для фиксированного i все индексы j от $i + 1$ до r удовлетворяют условию.

Таким образом, количество пар для фиксированного i равно $(r - i)$. Суммируя по всем i, получим общее количество пар.