Максимальное количество баллов за олимпиаду — 72
Максимальный балл за задание — 12

Задание 1.

Исследователь предполагает, что значения 17, 1, 23, 8, 2, 19, 16 получены из двух разных источников, причём значения одного источника близки. Чтобы изучать источники отдельно, он разбивает выборку на два непустых кластера (каждое число попадает ровно в один). Он минимизирует функцию потерь, равную сумме размахов кластеров; размах кластера — это разность его наибольшего и наименьшего элементов. Найдите минимум функции потерь по всем разбиениям.

Ответ. 14.

Критерии оценивания. Верный ответ — 12 баллов; неверный или отсутствующий ответ — 0 баллов.

Решение.

Пусть A — кластер, содержащий число 1, а m — наименьшее число кластера B. Все числа, меньшие m, лежат в A, а a — наибольшее из них.

Если 23 лежит в A, то размах A равен 22 и сумма не меньше 22.

Иначе $23 \in B$. Кластер A содержит числа 1 и a, кластер B — числа m и 23, поэтому

$$\text{размах } A + \text{размах } B \geqslant (a-1) + (23-m) = 22 - (m-a).$$

Величина $m - a$ — это один из зазоров между соседними числами выборки: 1, 6, 8, 1, 2, 4. Наибольший равен 8, поэтому сумма размахов не меньше $22 - 8 = 14$.

Оценка достигается: $\{1, 2, 8\}$ и $\{16, 17, 19, 23\}$ дают $7 + 7 = 14$.

Задание 2.

Три модели независимо друг от друга определяют, есть ли на картинке светофор.

Первая и вторая модели отвечают правильно с вероятностью $\dfrac{1}{2}$, третья — с вероятностью $\dfrac{3}{4}$. Итоговый ответ определяется большинством голосов, то есть считается правильным, если правильно ответили хотя бы две из трёх моделей.

С какой вероятностью итоговый ответ будет правильным?

Ответ. $\frac{5}{8}$.

Критерии оценивания. Верный ответ — 12 баллов; неверный или отсутствующий ответ — 0 баллов.

Решение.

Разберём случаи по первым двум моделям.

Обе правы — вероятность $\frac{1}{4}$; большинство право независимо от третьей.

Ровно одна права — вероятность $\frac{1}{2}$; большинство право тогда и только тогда, когда права третья, это ещё $\frac{1}{2} \cdot \frac{3}{4} = \frac{3}{8}$.

Обе ошиблись — вероятность $\frac{1}{4}$; большинство ошибается.

Итого $\frac{1}{4} + \frac{3}{8} = \frac{5}{8}$.

Задание 3.

На заводе проверили 9 деталей. У каждой детали измерили ширину зазора x в миллиметрах. Получились значения от 1 до 9. Детали со значениями 1, 3, 5, 7, 9 оказались бракованными, остальные — исправными.

Для автоматической проверки используют дерево решений. Сначала оно проверяет условие «$x \leqslant k_1$?». Если ответ «да», то проверяется условие «$x \leqslant k_2$?», иначе — «$x \leqslant k_3$?». После второй проверки дерево объявляет деталь исправной или бракованной.

Пороги k_1, k_2, k_3 и ответы дерева можно выбрать как угодно.

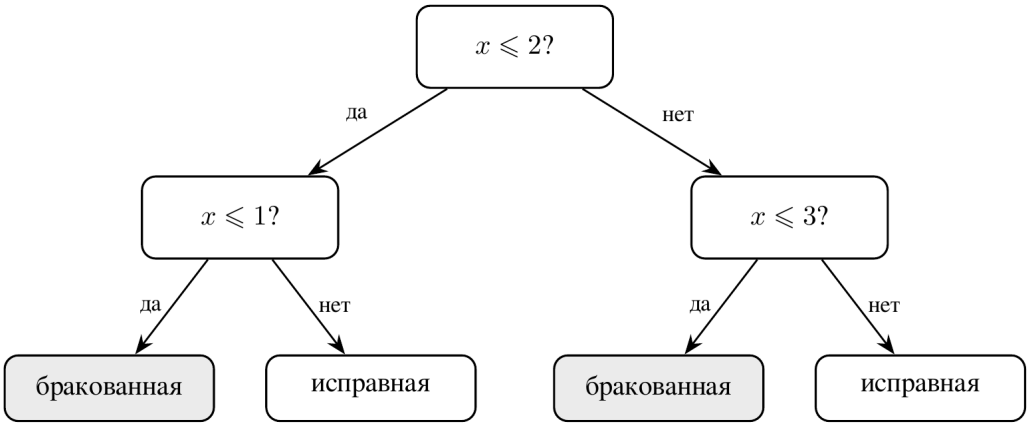
Какое наименьшее число ошибок может допустить такое дерево на этих девяти деталях?

Ответ. 3.

Критерии оценивания. Верный ответ — 12 баллов; неверный или отсутствующий ответ — 0 баллов.

Решение.

Пример. Подойдёт такое дерево.



Оно объявляет бракованными детали со значениями 1 и 3, а исправными — детали со значениями 2 и $4, 5, \ldots, 9$. Ошибки только на деталях 5, 7 и 9 — три ошибки.

Оценка. Каждый исход задаётся парой условий вида «$x \leqslant k$» или «$x > k$», а пересечение таких условий — промежуток. Исходов четыре, поэтому значения от 1 до 9 разбиты не более чем на четыре промежутка и на всех деталях одного промежутка дерево даёт один и тот же ответ.

Бракованные и исправные детали чередуются, поэтому в промежутке из m значений дерево ошибается хотя бы на $\lfloor m/2 \rfloor \geqslant \dfrac{m-1}{2}$ деталях.

Пусть промежутков $t \leqslant 4$, а их длины $m_1, \ldots, m_t$ в сумме дают 9. Тогда ошибок не меньше

$$\frac{m_1 - 1}{2} + \cdots + \frac{m_t - 1}{2} = \frac{9 - t}{2} \geqslant \frac{9 - 4}{2} = 2{,}5.$$

Число ошибок целое, значит, их не меньше трёх.

Задание 4.

Файл: similarity.csv

Во время олимпиады система проверки сравнивает решения участников и оценивает их схожесть. Результаты сравнений находятся в файле similarity.csv. Файл содержит четыре столбца:

student_1 — идентификатор первого участника (целое число);

student_2 — идентификатор второго участника (целое число);

task_id — номер задачи (целое число);

similarity — оценка схожести двух решений (вещественное число).

Один участник мог отправить несколько решений одной задачи. Каждая строка соответствует сравнению двух отправленных решений. Поэтому в строках могут совпадать идентификаторы участников и номер задачи, а различаться только значения similarity.

В данных возможны ошибки. Сначала очистите данные:

1. В столбце similarity замените пропущенные значения на 0. Значения меньше 0 замените на 0, а значения больше 1 — на 1.

2. Удалите строки, в которых student_1 = student_2.

3. Оставьте только строки, в которых similarity не меньше 0.92. Такие строки будем называть подозрительными. Если несколько подозрительных строк относятся к тем же двум участникам и одной задаче, при ответе на первый вопрос учитывайте каждую такую строку.

По очищенным данным определите нарушителей по следующим правилам:

порядок участников в паре не имеет значения: пары (17, 42) и (42, 17) считаются одной и той же парой;

пара участников считается нарушившей правила, если подозрительные строки для этой пары относятся не менее чем к трём различным задачам;

если для одной задачи осталось несколько подозрительных строк с одной и той же парой участников, эта задача учитывается только один раз;

если пара признана нарушившей правила, с олимпиады снимаются оба участника этой пары.

а) Сколько подозрительных строк осталось после очистки данных?

б) Сколько различных пар участников признано нарушившими правила?

в) Сколько различных участников необходимо снять с олимпиады?

Ответ. По порядку вопросов: 19; 4; 7.

Критерии оценивания. Баллы за правильные ответы по порядку вопросов: 2; 3; 7 соответственно. Баллы за пункты суммируются, максимум — 12. За неверный или отсутствующий ответ на отдельно оцениваемый пункт — 0 баллов, если частичное оценивание не указано отдельно.

Решение.

Сначала очистим столбец `similarity`, удалим сравнения участника с самим собой и оставим строки с `similarity` не меньше 0.92. Чтобы порядок участников не влиял на пару, в каждой оставшейся строке запишем меньший идентификатор первым, а больший — вторым.

Для каждой пары посчитаем число различных значений `task_id`. Важно использовать `nunique()`, а не число строк: одна задача может встретиться несколько раз. После отбора пар с результатом не меньше 3 объединим идентификаторы обоих участников и посчитаем число различных значений.

Решение на языке Python:

```
import pandas as pd
import numpy as np

df = pd.read_csv("data/similarity.csv")

df["similarity"] = df["similarity"].fillna(0).clip(0, 1)
df = df[df["student_1"] != df["student_2"]].copy()

df["student_min"] = np.minimum(df["student_1"], df["student_2"])
df["student_max"] = np.maximum(df["student_1"], df["student_2"])

suspicious = df[df["similarity"] >= 0.92]
tasks_by_pair = suspicious.groupby(
    ["student_min", "student_max"]
)["task_id"].nunique()

bad_pairs = tasks_by_pair[tasks_by_pair >= 3]
bad_students = np.unique(np.concatenate([
    bad_pairs.index.get_level_values("student_min"),
    bad_pairs.index.get_level_values("student_max"),
]))

print(len(suspicious))
print(len(bad_pairs))
print(len(bad_students))
```

После фильтрации осталось 19 строк. Нарушившими правила признаны 4 различные пары; в них входят 7 различных участников.

Ответ: 19; 4; 7.

Задание 5.

Согласованная группа

Ограничение по времени: 1 секунда

Ограничение по памяти: 256 мегабайт

При работе с нейронной сетью вы получаете эмбеддинги объектов — векторы в трёхмерном пространстве признаков. Для ускорения поиска похожих объектов вы хотите использовать манхэттенское расстояние вместо евклидова: оно считается быстрее, так как не требует извлечения квадратного корня.

Однако замена метрики корректна не всегда: для некоторых пар эмбеддингов манхэттенское и евклидово расстояния совпадают, а для некоторых — нет. Чтобы понять, на каких данных замена безопасна, вы хотите найти максимально большое подмножество эмбеддингов, внутри которого обе метрики дают одинаковый результат для любой пары объектов.

Подмножество называется идеально согласованным, если для любой пары входящих в него точек манхэттенское расстояние в точности равно евклидову.

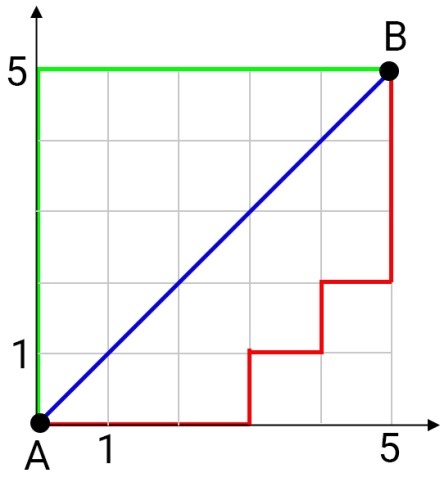
Евклидово расстояние для двух точек с координатами (x_1, y_1, z_1) и (x_2, y_2, z_2) вычисляется как корень из суммы квадратов разностей координат:

$$d_{Euc}(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Манхэттенское расстояние для двух точек с координатами (x_1, y_1, z_1) и (x_2, y_2, z_2) равно сумме абсолютных разностей их координат:

$$d_{Manh}(A, B) = |x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|$$

На изображении представлен пример евклидова расстояния (синим цветом) и манхэттенского (красным и зелёным цветом) в двумерном пространстве:



Вам нужно найти размер максимально возможного подмножества, которое является идеально согласованным.

Формат входных данных

Первая строка входных данных содержит одно целое число n $(2 \leqslant n \leqslant 1000)$.

Следующие n строк: в i-й строке три целых числа x_i, y_i, z_i $(|x_i|, |y_i|, |z_i| \leqslant 10^4)$ — координаты i-й точки.

Формат выходных данных

Программа должна выводить единственное число — размер максимально возможного подмножества, которое является идеально согласованным.

Ответ. Максимальное количество точек с одинаковой парой координат (x, y), (x, z) или (y, z).

Критерии оценивания. Программа выводит максимальный размер идеально согласованного подмножества. Одинаковые входные точки учитываются с их кратностью.

Решение.

Обозначим разности координат $a = x_1 - x_2$, $b = y_1 - y_2$, $c = z_1 - z_2$. Возведём равенство в квадрат: $(|a| + |b| + |c|)^2 = a^2 + b^2 + c^2$

$a^2 + b^2 + c^2 + 2|a||b| + 2|a||c| + 2|b||c| = a^2 + b^2 + c^2$

$2(|a||b| + |a||c| + |b||c|) = 0$

Каждое слагаемое неотрицательно, значит, равенство возможно **только когда хотя бы две из трёх разностей равны нулю**.

Иначе говоря, две точки должны отличаться **не более чем по одной координате**. Следовательно, всё подмножество лежит на одной прямой, параллельной одной из трёх координатных осей (либо все точки совпадают).

Таким образом, нам нужно найти максимальное число точек на прямой, параллельной одной из осей:

- прямые вдоль z задаются парой (x, y),
- прямые вдоль y задаются парой (x, z),
- прямые вдоль x задаются парой (y, z).

Для каждой из трёх пар координат считаем, сколько раз встречается каждая пара значений. Ответ — максимум по всем трём наборам счётчиков.

Таким образом, все что нам нужно сделать:

1. Завести три словаря:
   - $xy[(x, y)]$ — счётчик точек с одинаковыми x, y (прямые вдоль z),
   - $xz[(x, z)]$ — счётчик точек с одинаковыми x, z (прямые вдоль y),
   - $yz[(y, z)]$ — счётчик точек с одинаковыми y, z (прямые вдоль x).
2. Для каждой точки увеличить соответствующие три счётчика.
3. Ответ — максимум по значениям всех трёх словарей.

Сложность: в среднем $O(n)$ по времени и $O(n)$ по памяти.

Код решения на языке Python:

```
n = int(input())

xy = {}
xz = {}
yz = {}

for _ in range(n):
    x, y, z = map(int, input().split())

    xy[(x, y)] = xy.get((x, y), 0) + 1
    xz[(x, z)] = xz.get((x, z), 0) + 1
    yz[(y, z)] = yz.get((y, z), 0) + 1

res = 0
for d in (xy, xz, yz):
    for v in d.values():
        if v > res:
            res = v

print(res)
```

Задание 6.

Слои признаков

Ограничение по времени: 1 секунда

Ограничение по памяти: 256 мегабайт

Для обучения модели используются данные, полученные от датчика, считающего количество людей в аудитории каждую секунду. В ходе предварительной обработки необходимо скорректировать значения этого датчика из-за имевших место случаев пропуска человека (один и тот же человек или несколько человек не учитывались во все моменты времени).

Даны две последовательности целых неотрицательных чисел одинаковой длины: исходные значения датчика и значения, которые должны быть получены после коррекции (эти значения переданы контрольным наблюдателем).

Система коррекции позволяет за одну операцию выбрать некоторый непустой отрезок последовательности и увеличить текущие значения во все моменты времени этого интервала на 1. Такая операция моделирует добавление единичной поправки, действующей на датчик в течение некоторого промежутка времени (человек в это время находился в комнате, но не был зафиксирован датчиком).

Необходимо определить, можно ли получить требуемую последовательность значений из исходной с помощью таких операций. Если это возможно, найдите минимальное количество операций коррекции.

Формат входных данных

В первой строке вводится натуральное число n ($1 \leqslant n \leqslant 200\,000$).

В следующих n строках записаны исходные значения признака: в каждой строке ровно одно целое число.

В следующих n строках записаны целевые значения признака: в каждой строке ровно одно целое число.

Все числа целые неотрицательные и не превосходят 10^9.

Формат выходных данных

Если сделать требуемое невозможно, выведите `NO SOLUTION`. В противном случае выведите минимальное число операций.

Ответ. Пусть $d_i = b_i - a_i$. Если хотя бы одно $d_i < 0$, ответ — `NO SOLUTION`. Иначе ответ равен $d_1 + \sum_{i=2}^{n} \max(0, d_i - d_{i-1})$.

Критерии оценивания. При невозможности преобразования выводится `NO SOLUTION`; иначе — именно минимальное число операций.

Решение.

Обозначим через $d_i = b_i - a_i$ необходимое увеличение в позиции i. Если хотя бы одно $d_i < 0$, ответ невозможен, поскольку разрешены только увеличения.

Каждый положительный подъём последовательности d требует начать соответствующее число новых отрезков. Поэтому минимум равен

$$d_1 + \sum_{i=2}^{n} \max(0, d_i - d_{i-1}).$$

Отрезки можно заканчивать при уменьшении d_i, поэтому эта оценка достижима.

Сложность: $O(n)$, память $O(n)$.

```
import sys


def main():
    data = sys.stdin.buffer.read().split()
    n = int(data[0])
    a = list(map(int, data[1:n + 1]))
    b = list(map(int, data[n + 1:2 * n + 1]))
    prev = 0
    ans = 0
    for x, y in zip(a, b):
        d = y - x
        if d < 0:
            print("NO SOLUTION")
            return
        if d > prev:
            ans += d - prev
        prev = d
    print(ans)


if __name__ == "__main__":
    main()
```

