Максимальное количество баллов за олимпиаду — 600

Задание 1. Картинки в папке

В папке на компьютере размещено 3000 изображений трёх типов: с кошкой, с собакой, с кошкой и собакой одновременно. Изображений всех типов одинаковое количество. Модель ИИ ответила на два вопроса к каждой картинке: есть ли на ней кошка и есть ли на ней собака. На первый вопрос было получено 2790 ответов «да», а на второй — 2861. Удалили все изображения, для которых хотя бы один ответ ИИ был неверным. Какое наименьшее число изображений могло остаться в папке?

Ответ: 651

Критерий оценивания: точное совпадение ответа — 100 баллов

Максимальный балл за задание — 100

Решение. Оценка. Заметим, что на первый вопрос ответов НЕТ ровно 210, а на второй — 139. Значит, общее число ответов НЕТ 349. Таким образом, из 1000 изображений с кошкой и собакой хотя бы для $1000 - 349 = 651$ было два ответа ДА, эти изображения точно не будут удалены. Пример. Если модель ИИ ответила НЕТ на первый вопрос ровно для 139 изображений с кошкой и собакой и ответила НЕТ на второй вопрос ровно для 210 других изображений с кошкой и собакой, то верно на оба вопроса она ответила в точности для оставшихся 651 изображений с кошкой и собакой.

Задание 2. Сколько расстановок дают положительный определитель

Матрицей $n \times m$ будем называть таблицу из чисел, состоящую из n строк и m столбцов. Определитель матрицы 2×2 — это число, вычисляемое по формуле. Для $\begin{pmatrix} p & q \\ r & s \end{pmatrix}$ его значение равно $ps - qr$.

Сколько существует способов расставить числа 1, 2, 3, 6 по клеткам матрицы 2×2 (каждое число используется ровно один раз) так, чтобы $ps - qr > 0$?

Ответ: 8

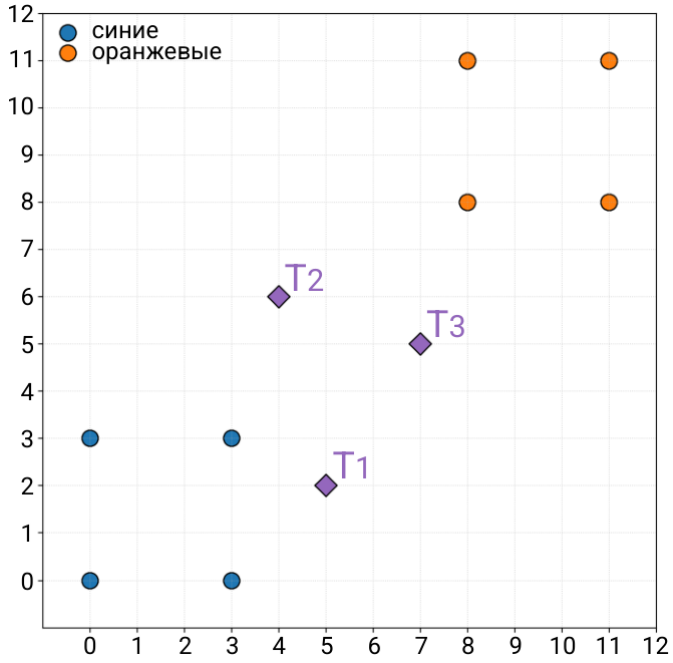
Критерий оценивания: точное совпадение ответа — 100 баллов

Максимальный балл за задание — 100

Решение. Пусть матрица имеет вид $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ и состоит из чисел 1, 2, 3, 6 без повторений. Знак определителя $det = ad - bc$ зависит только от того, какие два числа стоят на главной диагонали (a, d) и какие два — на побочной (b, c): порядок внутри пары не меняет произведения ad и bc. Разбиения множества $\{1, 2, 3, 6\}$ на две пары всего три: $\{1, 6\}\&\{2, 3\}$(6 и 6), $\{1, 2\}\&\{3, 6\}$(2 и 18), $\{1, 3\}\&\{2, 6\}$(3 и 12). В первом случае произведения равны, детерминант равен нулю и не подходит. В двух остальных случаях он положителен тогда и только тогда, когда на главной диагонали стоит пара с большим произведением. Для фиксированного выбора пар по диагоналям порядок внутри каждой пары можно переставлять двумя способами, итого $2! \cdot 2! = 4$ расстановки. Следовательно, положительных расстановок $4 + 4 = 8$.

Задание 3. Свой среди чужих

На плоскости даны восемь обучающих точек — четыре синие и четыре оранжевых, а также три тестовые точки T_1, T_2, T_3, обозначенные фиолетовыми ромбиками.



Цвет тестовой точки будем предсказывать по цветам её соседей.

Иногда разные координаты имеют очень разный масштаб: одна может меняться от 0 до 10000, а другая — от 0 до 0.00001. Тогда при вычислении расстояний первая координата начинает полностью «перебивать» вторую. Чтобы обе координаты влияли честно, первую иногда растягивают или сжимают.

В этой задаче первая координата умножается на положительное число $\lambda > 0$:

$(x,y) \mapsto (\lambda\, x,y)$ Расстояние между точками считаем евклидовым в новых координатах $(\lambda x,\, y)$. Цвет тестовой точки предсказываем так:

- находим 3 ближайших к ней обучающих точки;
- если среди них больше синих, считаем тестовую точку синей, если больше оранжевых — оранжевой;
- если расстояния совпадают, то более близкой считается точка с меньшим λx, а при равных λx — с меньшим y.

Зафиксируем $\lambda = 1$ и посмотрим, какого цвета при этом каждая тестовая точка T_i.

Назовём значение $\lambda > 0$ *интересным* для точки T_i, если при таком λ предсказанный цвет T_i отличается от её цвета при $\lambda = 1$.

Рассмотрим для каждой точки все её интересные значения λ.

Оказывается, что для каждой T_i возможны только два случая:

- интересных значений нет вообще — тогда цвет точки не меняется ни при каком $\lambda > 0$;
- интересные значения существуют и устроены так: есть число $L_i > 0$, что при всех $0 < \lambda < L_i$ цвет точки T_i другой, чем при $\lambda = 1$, а при всех $\lambda > L_i$ цвет уже такой же, как при $\lambda = 1$. При $\lambda = L_i$ цвет может совпадать или не совпадать с цветом при $\lambda = 1$ — это надо аккуратно проверить.

Число L_i будем называть *границей* интересных значений для точки T_i. Для каждой тестовой точки T_i:

- выведите -1, если интересных значений λ нет;
- найдите её границу L_i и выведите число $(L_i)^2$ в ином случае.

Ответ: -1, $\frac{5}{33}$, $\frac{5}{33}$

Критерий оценивания: точное совпадение ответа — 100 баллов

Максимальный балл за задание — 100

Решение. Зададим координаты обучающих точек (как на рисунке):

$$B_1 = (0,0), \quad B_2 = (0,3), \quad B_3 = (3,0), \quad B_4 = (3,3),$$
$$R_1 = (8,8), \quad R_2 = (8,11), \quad R_3 = (11,8), \quad R_4 = (11,11).$$

Обозначим $t = \lambda 2 > 0$. Тогда $d^2\lambda(T,P) = t(x - x_T)^2 + (y - y_T)^2$, поэтому расстояния — линейные функции от t. Критические точки возможны только при совпадении таких выражений.

1. Точка $T_1 = (5,2)$. Из сравнения расстояний видно, что точки B_3 и B_4 всегда ближе к T_1, чем любая оранжевая точка. Значит, среди трёх ближайших всегда есть как минимум две синие точки. Класс не меняется при любом $\lambda > 0$, поэтому не существует.

2. Точка $T_2 = (4, 6)$. Отбрасываем точки, которые всегда дальше трёх других. Остаются только B_4, B_2, R_1, R_3.

Единственное равенство расстояний:

$$d^2(B_2) = d^2(R_3) \iff 49t + 4 = 16t + 9 \iff t_c = \frac{5}{33}, \quad \lambda_c = \sqrt{\frac{5}{33}}$$

Для $0 < t < t_c$ тройка ближайших — R_1,R_3,B_4 (две оранжевых) → класс оранжевый. Для $t \geqslant t_c$ тройка — R_1,B_4,B_2 (две синих) → класс синий.

При $\lambda = 1$ точка синяя, значит, $\lambda_3^* = \sqrt{\frac{5}{33}}$

3. Точка $T_3 = (7, 5)$.

Та же четвёрка кандидатов B_4,B_2,R_1,R_3 и то же единственное критическое значение $t_c = \frac{5}{33}$. Для $0 < t \geqslant t_c$ тройка — R_1,B_4,B_2 (две синих) → класс синий. Для $t > t_c$ тройка — R_1,B_4,R_3 (две оранжевых) → класс оранжевый. При $\lambda = 1$ точка оранжевая, значит, ответ:

$$\lambda_3^* = \sqrt{\frac{5}{33}}.$$

$$T_1 : -1,\ T_2 : \frac{5}{33},\ T_3 : \frac{5}{33}.$$

Задание 4. Мониторинг энергопотребления магазина

В течение нескольких дней подряд в магазине измеряли энергопотребление каждые 15 минут. Таким образом, за одни сутки получается 96 измерений. Результаты вы можете скачать в форматах XLSX, ODS или CSV.

В таблице содержится непрерывный ряд измерений без пропусков; в каждой строке указаны t (номер измерения, начиная с 0) и power (потребление в ваттах). Обозначим $x_t = \mathsf{power}[t]$.

Чтобы проанализировать динамику потребления, рассмотрим несколько производных величин.

Во-первых, будем отслеживать изменение мощности от шага к шагу. Для каждого $t \geqslant 1$ определим разность: $d_t = x_t - x_{t-1}$. Во-вторых, нас интересуют сглаженные характеристики. Скользящие средние вычисляются по «хвостовым» окнам, то есть берут текущий момент и несколько предыдущих:

- среднее потребление за последние 8 шагов: $\mathsf{ma\_8}[t] = \frac{1}{8}\sum_{j=0}^{7} x_{t-j} \quad (t \geqslant 7)$,
- среднее изменение потребления за последние 4 шага: $\mathsf{ma\_diff\_4}[t] = \frac{1}{4}\sum_{j=0}^{3} d_{t-j} \quad (t \geqslant 4)$.

Кроме того, по всему ряду значений x_t необходимо найти медиану med. Медиана — это число, которое стоит посередине после сортировки всех значений (или среднее двух центральных при чётном числе точек).

Поскольку измерения выполнялись много дней подряд, выражение t `mod` 96 указывает, на каком 15-минутном интервале суток находится момент t. Значения от 36 до 84 соответствуют промежутку с 09:00 до 21:00 включительно.

Сколько моментов времени t одновременно удовлетворяют трём условиям:

$\mathsf{ma\_diff\_4}[t] \geqslant 3.0$, $\mathsf{ma\_8}[t] \geqslant \mathsf{med}$, $36 \leqslant (t \bmod 96) \leqslant 84$?

Ответ: 193

Критерий оценивания: точное совпадение ответа — 100 баллов

Максимальный балл за задание — 100

Решение.

Сначала по столбцу power строим разностный ряд `diff_1`: это изменение мощности за один шаг (15 минут). Далее считаем два скользящих средних по хвостовым окнам, то есть окно всегда заканчивается в текущем моменте `t`: `ma_8` усредняет последние 8 значений power, а `ma_diff_4` — последние 4 значения `diff_1`. Оба скользящих средних считаем только при полном окне, поэтому первые несколько позиций дают пропуски и в подсчёт не попадают. После этого находим медиану исходного ряда power и отбираем те моменты времени, которые одновременно удовлетворяют трём условиям: средний прирост за последнее окно не меньше 3, сглаженное значение мощности не ниже медианы и шаг относится к «рабочему времени», то есть $36 \leqslant (t \ mod \ 96) \leqslant 84$. Количество таких моментов и является ответом.

```python
import pandas as pd

df = pd.read_csv("ts6_raw.csv")

s = df["power"]

diff_1 = s.diff()

ma_8      = s.rolling(window=8,  min_periods=8).mean()
ma_diff_4 = diff_1.rolling(window=4, min_periods=4).mean()

med = float(s.median())

t = df["t"].to_numpy()
in_hours = ((t % 96) >= 36) & ((t % 96) <= 84)

answer = ((ma_diff_4 >= 3.0) & (ma_8 >= med) & in_hours).sum()
print(answer)
```

—

Задание 5. Оптимальная конфигурация

Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт
Обучение больших нейронных сетей происходит на кластере — вычислительной инфраструктуре, которая позволяет использовать одновременно большое количество видеокарт. Видеокарты на кластере объединяются в ноды (блоки по 8 карт).

Для обучения больших языковых моделей методом обучения с подкреплением видеокарты делятся на два типа: одни генерируют данные, другие обновляют веса модели на основе генерированных данных. Вторая задача гораздо сложнее, поэтому при наилучшем распределении ресурсов на неё должно выделяться в k раз больше видеокарт. Обучение с оптимальным распределением ресурсов мы будем называть оптимальным.

Вы работаете в очень большой компании, поэтому ваши ресурсы не ограничены. Вы хотите выделить минимальное количество нод так, чтобы их видеокарты можно было разбить на два вышеописанных типа так, чтобы обучение при этом было оптимальным.

Формат входных данных

В первой и единственной строке входных данных вводится одно целое число k ($1 \leqslant k \leqslant 1000000$).

Формат выходных данных

Выведите одно число: минимальное число нод, которые нужно выделить так, чтобы обучение было оптимальным.

Примеры

стандартный ввод	стандартный вывод
5	3

Максимальный балл за задание — 100

Решение

От нас требуется найти минимальное такое число N, которое будет делиться на 8 и на k. Это ровно $N = \mathrm{lcm}(8, k)$.

Так как $8 = 2^3$, можно обойтись без алгоритма Евклида: достаточно домножать k на 2, пока получившееся число не начнёт делиться на 8. Это даст минимальное общее кратное с 8, потому что мы добавляем ровно недостающую степень двойки.

В ответе нужно вывести $\mathrm{lcm}(k, 8)/8$.

```python
from math import lcm

k = int(input())
print(lcm(k, 8) // 8)
```

Задание 6. Распознавание объектов

Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт
Одна из задач машинного обучения — распознавание объектов. Она заключается в том, чтобы выделить на изображении зоны, в которых находятся объекты, и сказать, что это за объекты.

Гоша запустил умную нейронную сеть для распознавания объектов. Она вернула таблицу $n \times m$, где в каждой клетке указан номер класса, к которому принадлежит находящийся в клетке объект по мнению модели, или 0, если модель считает, что в этой клетке нет никаких интересующих её объектов.

К сожалению, этого недостаточно: Гоша хочет выделить в таблице прямоугольники-рамки со сторонами, параллельными её краям, так, чтобы внутри этих рамок были распознанные объекты. Объектами Гоша считает все клетки, которые модель распознала как клетки одного конкретного класса.

Помогите Гоше: выделите в таблице все распознанные умной нейронной сетью объекты. Так как объектов в таблице может быть очень много, посчитайте и выведите всего одно число: сумму площадей всех прямоугольников-рамок объектов.

Формат входных данных

В первой строке вводятся два целых числа n, m ($1 \leqslant n, m \leqslant 1000$).
В следующих n строках вводится по m чисел a_{ij} — номера классов объектов, которые модель нашла в конкретной клетке, или 0, если в этой клетке не был распознан ни один объект.

Формат выходных данных

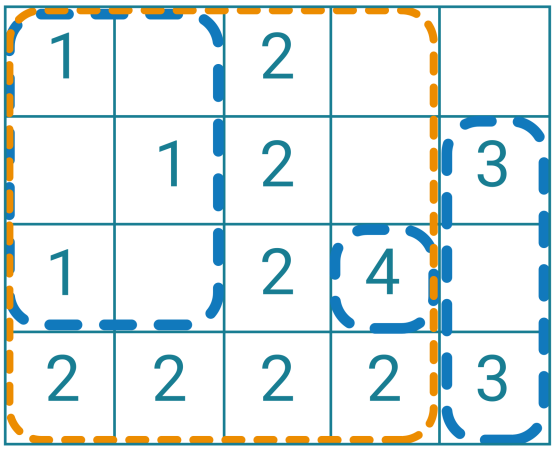
Выведите одно целое число: сумму площадей прямоугольников-рамок для всех распознанных объектов.

Примеры

стандартный ввод	стандартный вывод
1 0 2 0 0 0 1 2 0 3 1 0 2 4 0 2 2 2 2 3	26

Замечание

Рассмотрим первый пример из условия.



Решение

Для каждого цвета клетки требуется найти минимальную рамку, которая покрывает все клетки этого цвета. Будем для каждого цвета хранить минимальные и максимальные координаты по строке и столбцу: `min_x`, `max_x`, `min_y`, `max_y`. Тогда площадь рамки для цвета c равна $(\texttt{max\_x}[c] - \texttt{min\_x}[c] + 1) \cdot (\texttt{max\_y}[c] - \texttt{min\_y}[c] + 1)$, а ответ — сумма этих площадей по всем цветам (цвет 0 игнорируем).

Чтобы не хранить огромные `map` по ключу-цвету, будем динамически присваивать каждому встреченному цвету компактный индекс $0..q-1$ через `unordered_map` и обновлять границы по этому индексу за $O(1)$ на клетку.

```cpp
#include <iostream>
#include <vector>
#include <unordered_map>
#include <algorithm>

using namespace std;

int main() {
    ios::sync_with_stdio(false);
    cin.tie(nullptr);

    int n, m;
    cin >> n >> m;

    vector<int> mxi(n * m + 1, -1);
    vector<int> mni(n * m + 1, n);
    vector<int> mxj(n * m + 1, -1);
    vector<int> mnj(n * m + 1, m);

    unordered_map<int, int> id;
    id.reserve(n * m + 1);

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < m; ++j) {
            int x;
            cin >> x;

            int k;
            auto it = id.find(x);
            if (it == id.end()) {
                k = (int)id.size();
                id[x] = k;
            } else {
                k = it->second;
            }

            mxi[k] = max(mxi[k], i);
            mni[k] = min(mni[k], i);
            mxj[k] = max(mxj[k], j);
            mnj[k] = min(mnj[k], j);
        }
    }

    long long sum = 0;
    for (auto &[color, idx] : id) {
        if (color == 0) continue;
        sum += 1LL * (mxi[idx] - mni[idx] + 1) * (mxj[idx] - mnj[idx] + 1);
    }

    cout << sum << '\n';
    return 0;
}
```

Максимальный балл за задание — 100