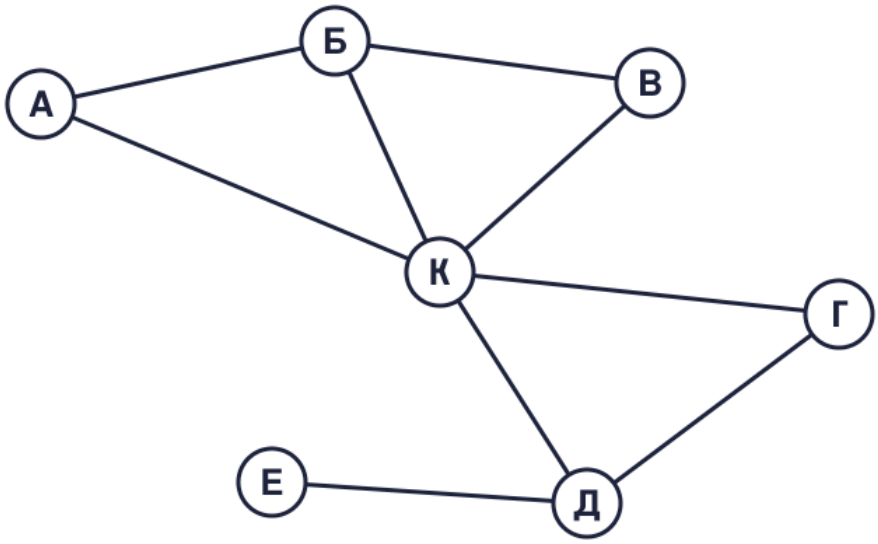
ЕГЭНИЙ · Информатика

Вариант 28 · Вариант с ответами и решениями

Задание 1

На рисунке схема дорог N-ского района изображена в виде графа, в таблице содержатся сведения о протяжённости каждой из этих дорог (в километрах).

Номер пункта	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1						21	
2			10	47	42	20	9
3		10				57	
4		47			56		
5		42		56			19
6	21	20	57				
7		9			19		



Так как таблицу и схему рисовали независимо друг от друга, нумерация населённых пунктов в таблице никак не связана с буквенными обозначениями на графе. Определите, на сколько километров длина дороги БК отличается от длины дороги ДЕ.

В ответе запишите целое число.

Ответ 21

Посчитаем степени вершин (количество дорог из пункта). На графе: А — 2, Б — 3, В — 2, Г — 2, Д — 3, Е — 1, К — 5. В таблице: П1 — 1, П2 — 5, П3 — 2, П4 — 2, П5 — 3, П6 — 3, П7 — 2.

Сопоставляем вершины по степеням и по соседям: Б — вершина степени 3, степени её соседей: 2, 2, 5; так устроен только П5; Г — вершина степени 2, не соединённая с Б; этому соответствует только П3; Д —

вершина степени 3, соединённая с Г; этому соответствует только П6; степень 1 на графе только у вершины Е, в таблице — только у пункта 1: Е = П1; степень 5 на графе только у вершины К, в таблице — только у пункта 2: К = П2.

Дорога БК — это П5–П2, её длина 42 км. Дорога ДЕ — это П6–П1, её длина 21 км.

Разность: |42 – 21| = **21**.

Задание 2

Миша заполнял таблицу истинности логической функции F

(z ≡ ¬x) ∧ (w → z) ∧ (y ∨ w),

но успел заполнить лишь фрагмент из трёх различных её строк, даже не указав, какому столбцу таблицы соответствует каждая из переменных w, x, y, z.

???	???	???	???	F
	0		0	1
1	1	1		1
1	0			1

Определите, какому столбцу таблицы соответствует каждая из переменных w, x, y, z.

В ответе напишите буквы w, x, y, z в том порядке, в котором идут соответствующие им столбцы (сначала буква, соответствующая первому столбцу; затем буква, соответствующая второму столбцу, и т. д.). Буквы в ответе пишите подряд, никаких разделителей между буквами ставить не нужно.

Например, если функция задана выражением ¬x ∨ y, зависящим от двух переменных, а фрагмент таблицы имеет вид

???	???	F
0	1	0

то первому столбцу соответствует переменная y, а второму — переменная x. В ответе следует написать: yx.

Ответ ywzx

Функция равна 1 только в 4 строках полной таблицы истинности из 16, поэтому фрагмент почти однозначно задаёт расстановку переменных. Переберём программой все способы заполнить пустые клетки и все перестановки переменных по столбцам: подходят только заполнения, при которых три строки различны, а функция во всех трёх строках равна 1.

Импликацию a → b в Python удобно записывать как `a <= b`, тождество — как `a == b`.

Программа на Python

```
from itertools import product, permutations

def F(x, y, z, w):
    return (z == (not x)) and (w <= z) and (y or w)

answers = set()
for a1, a2, a3, a4, a5 in product([0, 1], repeat=5):
    t = [(a1, 0, a2, 0), (1, 1, 1, a3), (1, 0, a4, a5)]
    if len(set(t)) == 3:              # строки фрагмента различны
        for p in permutations('xyzw'):
            if all(F(**dict(zip(p, r))) == 1 for r in t):
                answers.add(''.join(p))
print(*answers)
```

Программа выводит единственную перестановку: **ywzx**.

Задание 3

Задание выполняется с использованием прилагаемого файла.

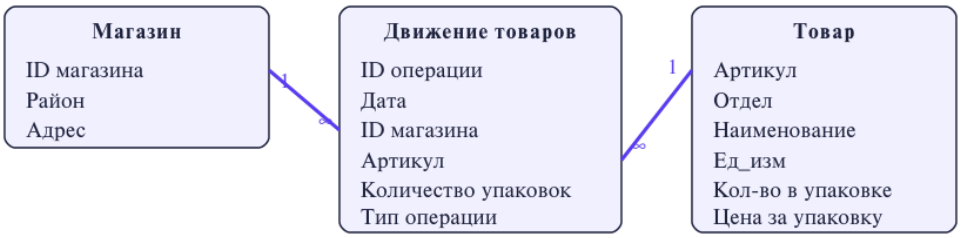
В файле приведён фрагмент базы данных «Канцтовары» о поставках товаров в магазины районов города. База данных состоит из трёх таблиц.

Таблица «Движение товаров» содержит записи о поступлении товаров со склада в магазины в течение

июня 2026 г., а также информацию о проданных товарах. Поле *Тип операции* содержит значение *Поступление* или *Продажа*, а в соответствующее поле *Количество упаковок, шт.* внесена информация о том, сколько упаковок товара поступило в магазин или было продано в течение дня.

Таблица «Товар» содержит информацию об основных характеристиках каждого товара: артикул, отдел, наименование, единицу измерения, количество в упаковке и цену за упаковку. Таблица «Магазин» содержит информацию о местонахождении магазинов: ID магазина, район, адрес.

На рисунке приведена схема указанной базы данных.



Используя информацию из приведённой базы данных, определите, на сколько увеличилось количество упаковок всех видов тетрадей в магазинах Октябрьского района за период с 7 по 14 июня включительно. В ответе запишите только число.

Файл к заданию: 3.xlsx (114 КБ)

Ответ 702

В электронной таблице. На листе «Движение товаров» добавьте столбцы: район магазина `=ВПР(C2;$Магазин.$A$2:$B$30;2;0)`, наименование `=ВПР(D2;$Товар.$A$2:$F$100;3;0)`.

Включите автофильтр и оставьте: даты с 07.06.2026 по 14.06.2026, район — Октябрьский, наименование начинается со слова «Тетрадь». Увеличение = все поступившие упаковки минус все проданные за этот период. В таблице удобно завести столбец со знаком: количество для поступления и минус количество

для продажи. Сумму видимых строк даст `=ПРОМЕЖУТОЧНЫЕ.ИТОГИ(9;…)`.

Программой. Сохраните каждый лист как CSV с разделителем «;» (Файл → Сохранить как → Текст CSV) и выполните программу.

Программа на Python

```
import csv

def read(name):
    with open(name, encoding='utf-8') as f:
        return list(csv.reader(f, delimiter=';'))[1:]

goods = {r[0]: r for r in read('товар.csv')}      # артикул -> строка
shops = {r[0]: r[1] for r in read('магазин.csv')}  # магазин -> район
total = 0
for oid, date, shop, art, cnt, typ in read('движение.csv'):
    day = int(date[:2])
    g = goods[art]
    if 7 <= day <= 14 and shops[shop] == 'Октябрьский' and g[2].startswith('Тетрадь'):
        if typ == 'Поступление':
            total += int(cnt)
        else:
            total -= int(cnt)
print(total)
```

Ответ: **702**.

Задание 4

По каналу связи передаются сообщения, содержащие только буквы из набора: А, Б, Д, Н. Для передачи используется двоичный код, удовлетворяющий условию Фано. Кодовые слова для некоторых букв известны: А – 00, Н – 10. Для двух оставшихся букв Б и Д кодовые слова неизвестны.

Какое наименьшее количество двоичных знаков потребуется для кодирования слова БАНДАНА, если известно, что оно закодировано минимально возможным количеством двоичных знаков?

Примечание. Условие Фано означает, что никакое кодовое слово не является началом другого кодового слова. Это обеспечивает возможность однозначной расшифровки закодированных сообщений.

Ответ 14

Нарисуем двоичное дерево известных кодов. Свободные вершины, с которых может начинаться новый код: 01, 11. Чем чаще буква встречается в слове, тем короче должен быть её код.

Количество букв в слове: А — 3, Б — 1, Д — 1, Н — 2. Оптимально, например: Б – 11, Д – 01.

Длина кода слова: 2·3 + 2·1 + 2·1 + 2·2 = **14**.

Проверим перебором всех кодов длиной до 5 для неизвестных букв:

Программа на Python

```
from itertools import product

known = {'А': '00', 'Н': '10'}
word = 'БАНДАНА'
words = [''.join(t) for n in range(1, 6) for t in product('01', repeat=n)]

def fano(codes):
    return all(not b.startswith(a) for a in codes for b in codes if a != b)

best = 10 ** 9
for combo in product(words, repeat=2):
    codes = dict(known)
    codes.update(zip(['Б', 'Д'], combo))
    if len(set(codes.values())) == len(codes) and fano(codes.values()):
        best = min(best, sum(len(codes[c]) for c in word))
print(best)
```

Ответ: **14**.

Задание 5

На вход алгоритма подаётся натуральное число N. Алгоритм строит по нему новое число R следующим образом.

1. Строится троичная запись числа N.

2. Далее эта запись обрабатывается по следующему правилу:
а) если число N делится на 3, то слева к записи дописывается цифра 1, а справа — цифры 02;
б) если число N на 3 не делится, то справа дописывается остаток от деления на 3 суммы цифр троичной записи числа N.

Полученная таким образом запись является троичной записью искомого числа R.

3. Результат переводится в десятичную систему и выводится на экран.

Например, для исходного числа $25_{10} = 221_3$ результатом является число $2212_3 = 77_{10}$, а для исходного числа $28_{10} = 1001_3$ результатом является число $10012_3 = 86_{10}$.

Укажите минимальное число N, после обработки которого с помощью этого алгоритма получается число R, не меньшее 388. В ответе запишите это число в десятичной системе счисления.

Ответ 18

Результат работы алгоритма растёт вместе с N не строго монотонно (для разных N работают разные правила), поэтому надёжнее перебрать N программой. В записи R на несколько разрядов больше, чем в записи N, так что R больше N и перебора до 1000 с запасом достаточно.

Программа на Python

```
def tern(n):
    s = ''
    while n > 0:
        s = str(n % 3) + s
        n //= 3
    return s

def R(n):
    t = tern(n)
    if n % 3 == 0:
        t = '1' + t + '02'
    else:
        t = t + str(sum(map(int, t)) % 3)
    return int(t, 3)

print(min(n for n in range(1, 1000) if R(n) >= 388))
```

Ответ: **18**.

Задание 6

Исполнитель Черепаха действует на плоскости с декартовой системой координат. В начальный момент Черепаха находится в начале координат, её голова направлена вдоль положительного направления оси ординат, хвост опущен. При опущенном хвосте Черепаха оставляет на поле след в виде линии. В каждый конкретный момент известно положение исполнителя и направление его движения. У исполнителя существует 6 команд: Поднять хвост, означающая переход к перемещению без рисования; Опустить хвост, означающая переход в режим рисования; Вперёд n (где n – целое число), вызывающая передвижение Черепахи на n единиц в том направлении, куда указывает её голова; Назад n (где n – целое число), вызывающая передвижение в противоположном голове направлении; Направо m (где m – целое число), вызывающая изменение направления движения на m градусов по часовой стрелке; Налево m (где m – целое число), вызывающая изменение направления движения на m градусов против часовой стрелки.

Запись Повтори k [Команда1 Команда2 … КомандаS] означает, что последовательность из S команд повторится k раз.

Черепахе был дан для исполнения следующий алгоритм.

Повтори 4 [Вперёд 29 Направо 90 Вперёд 17 Направо 90]
Поднять хвост
Вперёд 12 Направо 90 Вперёд 3 Налево 90
Опустить хвост
Повтори 2 [Вперёд 16 Направо 90 Вперёд 8 Направо 90]

Определите, сколько точек с целочисленными координатами находятся внутри области пересечения фигур, ограниченных заданными алгоритмом линиями, включая точки на границах этого пересечения.

Ответ 153

Первый цикл рисует прямоугольник: Черепаха идёт вверх на 29, поворачивает направо и идёт на 17. Получается прямоугольник $0 \le x \le 17$, $0 \le y \le 29$.

Перемещение без рисования приводит Черепаху в точку (3; 12), голова снова смотрит вверх.

Второй цикл: вверх на 16, затем вправо на 8. Второй прямоугольник: $3 \le x \le 11$, $12 \le y \le 28$.

Пересечение: 3 ≤ x ≤ 11, 12 ≤ y ≤ 28. Целых точек: (11 – (3) + 1) · (28 – (12) + 1) = 9 · 17 = **153**.

Проверка программой

```
A = (0, 17, 0, 29)      # x1, x2, y1, y2
B = (3, 11, 12, 28)
inside = lambda r, x, y: r[0] <= x <= r[1] and r[2] <= y <= r[3]
print(sum(1 for x in range(-100, 101) for y in range(-100, 101) if inside(A, x, y) and inside(B, x, y)))
```

Задание 7

Голосовое сообщение оцифровывается в формате моно с частотой дискретизации 24 000 Гц и глубиной кодирования 32 бит. Определите наименьшее количество Кбайт, необходимое для сохранения сообщения в памяти (без учёта заголовка), если его длительность — 4 мин 26 с. В ответе укажите только число.

Ответ 24938

Длительность: 4 · 60 + 26 = 266 с. Объём: 1 · 24 000 · 4 байта · 266 = 25 536 000 байт.

В Кбайтах: 25 536 000 / 1024 ≈ 24937,50. Нужно целое число Кбайт, чтобы поместилось всё сообщение, — округляем вверх: **24938**.

Задание 8

Все четырёхбуквенные слова, составленные из букв Н, К, Т, Ц, А, Е, записаны в алфавитном порядке и пронумерованы. Вот начало списка:

1. АААА
2. АААЕ
3. АААК
4. АААН
5. АААТ
6. АААЦ

……

Определите, под каким номером в этом списке стоит последнее слово с нечётным номером, которое не содержит двух букв Ц подряд и содержит не менее двух букв А.

Примечание. Слово — последовательность идущих подряд букв, не обязательно осмысленная.

Ответ 1225

Важно взять буквы в алфавитном порядке: А, Е, К, Н, Т, Ц — тогда product перебирает слова в том же порядке, что и в списке, а нумерацию начинаем с 1.

Программа на Python

```
from itertools import product

letters = sorted('АЕКНТЦ')
last = 0
for i, w in enumerate(product(letters, repeat=4), 1):
    w = ''.join(w)
    if i % 2 == 1 and 'ЦЦ' not in w and w.count('А') >= 2:
        last = i
print(last)
```

Ответ: **1225**.

Задание 9

Задание выполняется с использованием прилагаемого файла.

Откройте файл электронной таблицы, содержащей в каждой строке семь натуральных чисел. Определите количество строк таблицы, для которых выполнены оба условия:

– в строке все числа различны;
– квадрат наибольшего числа строки меньше суммы квадратов остальных шести.

В ответе запишите только число.

Файл к заданию: 9.xlsx (226 КБ)

Ответ 5672

В электронной таблице. Числа строки удобно упорядочить формулами `=НАИМЕНЬШИЙ($A1:$G1;1)` … `=НАИМЕНЬШИЙ($A1:$G1;7)` в соседних столбцах. Все числа различны, если для каждой ячейки

`СЧЁТЕСЛИ($A1:$G1;A1)=1`. Итог по строке — 1, если выполнены оба условия, ответ — сумма этого столбца.

Программой. Сохраните лист как CSV с разделителем «;».

Программа на Python

```
c = 0
for line in open('9.csv'):
    a = sorted(int(x) for x in line.split(';'))
    if len(set(a)) == 7:
        if a[6] ** 2 < sum(x * x for x in a[:6]):
            c += 1
print(c)
```

Ответ: **5672**.

Задание 10

Файл 10-2.odt содержит первые три главы поэмы Н. В. Гоголя «Мёртвые души». С помощью текстового редактора определите, сколько раз во всём тексте файла встречается слово «Чичиков» (именно в этой форме, с учётом регистра) как отдельное слово. Слова, записанные через дефис (например, «что-то»), считаются состоящими из отдельных слов. В ответе запишите только число.

Текст для проверки программой: TXT.

Ответ 103

Откройте ODT в текстовом редакторе. Используйте поиск целых слов, учитывая регистр и указанный в условии фрагмент.

```
import re
L = 'A-Za-zA-Яа-яЁё0-9'           # символы, из которых состоят слова
def word(t, w, flags=0):          # вхождения w как отдельного слова
    return len(re.findall(rf'(?<![{L}]){w}(?![{L}])', t, flags))

t = open('10-2.txt', encoding='utf-8').read()
print(word(t, 'Чичиков'))
```

Задание 11

При регистрации изделий каждому из них присваивается серийный номер длиной 44 символа. Для записи номеров используется алфавит мощностью 607.

В базе данных для хранения каждого серийного номера отведено одинаковое и минимально возможное целое число байт. При этом используется посимвольное кодирование серийных номеров, все символы кодируются одинаковым и минимально возможным числом бит.

Определите объём памяти (в Кбайт), необходимый для хранения 8192 серийных номеров. В ответе запишите только целое число — количество Кбайт.

Ответ 440

$2^9 = 512 < 607 \le 1024 = 2^{10}$, значит, на символ нужно 10 бит.

Номер: $44 \cdot 10 = 440$ бит = 55 байт.

8192 номеров: $55 \cdot 8192 = 450\,560$ байт = 450 560 / 1024 = **440** Кбайт.

Задание 12

Исполнитель МТ представляет собой читающую и записывающую головку, которая может передвигаться вдоль бесконечной горизонтальной ленты, разделённой на равные ячейки. В каждой ячейке находится ровно один символ из алфавита исполнителя, включая специальный пустой символ λ.

Время работы исполнителя делится на такты. На каждом такте головка находится в одном из допустимых состояний q0, q1, …; в начальный момент — в состоянии q0. За один такт головка может изменить символ в текущей ячейке, сдвинуться в соседнюю ячейку слева или справа и перейти в новое состояние (или остаться в прежнем).

Программа задаётся таблицей: в первой строке перечислены символы, в первом столбце — состояния. На пересечении стоит команда из трёх элементов через запятую: записываемый символ; L, R или S — сдвиг влево, сдвиг вправо или остановка после выполнения команды; новое состояние. Например, команда 0, L, q3: записать в текущую ячейку «0», сдвинуться влево и перейти в состояние q3. Пустая клетка означает, что такая пара «символ – состояние» невозможна.

Выполните задание. На ленте в соседних ячейках записано двоичное представление числа 3314 без ведущих нулей. Ячейки справа и слева от последовательности заполнены пустыми символами «λ». В начальный момент времени головка расположена в ближайшей справа к последовательности ячейке.

Программа работы исполнителя:

	λ	**0**	**1**
q0	λ, L, q1		
q1	1, S, q1	1, S, q1	0, L, q1

Определите результат выполнения программы. В ответе запишите получившееся на ленте двоичное число в десятичной системе счисления.

Ответ 3315

$3314 = 110011110010_2$.

q0 сдвигает головку на последнюю цифру. q1 прибавляет единицу: единицы заменяет нулями и идёт влево (перенос), на первом нуле или пустой ячейке пишет 1 и останавливается. Программа прибавляет к числу 1.

Проверим, проследив такты программой.

Программа на Python

```
prog = {
    'q0': {'λ': ('λ', 'L', 'q1')},
    'q1': {'λ': ('1', 'S', 'q1'), '0': ('1', 'S', 'q1'), '1': ('0', 'L', 'q1')},
}
tape = dict(enumerate('110011110010'))
pos, state = 12, 'q0'
while True:
    w, move, state2 = prog[state][tape.get(pos, 'λ')]
    tape[pos] = w
    state = state2
    if move == 'S':
        break
    pos += 1 if move == 'R' else -1
res = ''.join(tape[i] for i in sorted(tape)).strip('λ')
print(res, int(res, 2))
```

На ленте остаётся 110011110011_2 = **3315**.

Задание 13

В терминологии сетей TCP/IP маской сети называют двоичное число, которое показывает, какая часть IP-адреса узла сети относится к адресу сети, а какая — к адресу узла в этой сети. Адрес сети получается в результате применения поразрядной конъюнкции к заданному адресу узла и его маске.

Даны адрес узла 172.76.111.89 и маска 255.255.255.192. Найдите адрес сети.

В ответе укажите сумму числовых значений октетов найденного IP-адреса. Например, если бы найденный адрес был равен 100.20.3.4, то в ответе следовало бы записать: 127.

Ответ 423

Октеты маски, равные 255, переносят соответствующие октеты адреса без изменений, октеты, равные 0, дают 0. Считать нужно только октет, где маска равна 192.

$89 = 01011001_2$, $192 = 11000000_2$. Поразрядная конъюнкция: $01000000_2 = 64$.

Адрес сети: 172.76.111.64. Сумма октетов: 172 + 76 + 111 + 64 = **423**.

Проверка программой

```
a = [172, 76, 111, 89]
m = [255, 255, 255, 192]
net = [x & y for x, y in zip(a, m)]
print(net, sum(net))
```

Задание 14

Значение арифметического выражения

25^(101)+5^(179)−5^(6)−x,

где x — целое положительное число, не превышающее 2000, записали в пятеричной системе счисления.

Определите наименьшее значение x, при котором в данной записи значения арифметического выражения содержится ровно 27 нулей.

В ответе запишите число в десятичной системе счисления.

Ответ 625

25101=5202: в пятеричной записи это единица и 202 нулей. Слагаемое 5179 добавляет ещё одну единицу, а вычитание 56+x «занимает» единицу из старшего разряда: младшие разряды превращаются в цифры 4, а несколько последних зависят от x.

Точное количество нулей для каждого x удобнее посчитать программой — Python работает с длинными числами напрямую.

Программа на Python

```
def zeros(n, b):
    c = 0
    while n > 0:
        if n % b == 0:
            c += 1
        n //= b
    return c

for x in range(1, 2001):
    if zeros(25 ** 101 + 5 ** 179 - 5 ** 6 - x, 5) == 27:
        print(x)
        break
```

Ответ: **625**.

Задание 15

Обозначим через ДЕЛ(n, m) утверждение «натуральное число n делится без остатка на натуральное число m»; пусть на числовой прямой дан отрезок B = [1058; 1623].

Для какого наибольшего натурального числа A логическое выражение

(x ∈ B) → (ДЕЛ(x, A) ∨ ¬ДЕЛ(x, 11) ∨ ¬ДЕЛ(x, 25))

истинно (то есть принимает значение 1) при любом целом положительном значении переменной x?

Ответ 275

Выражение может стать ложным только при x ∈ [1058; 1623], делящемся и на 11, и на 25, то есть на НОК(11, 25) = 275. Для всех таких x должно выполняться ДЕЛ(x, A).

Кратные 275 на отрезке: 1100, 1375. A должно делить оба числа, наибольшее такое A — НОД = **275**.

Проверка программой

```
def F(x, A):
    return (not (1058 <= x <= 1623)) or x % A == 0 or x % 11 != 0 or x % 25 != 0

for A in range(3000, 0, -1):
    if all(F(x, A) for x in range(1, 4000)):
        print(A)
        break
```

Задание 16

Алгоритм вычисления значения функции F(n), где n — натуральное число, задан следующими соотношениями:

F(n) = 1 при n = 1;
F(n) = (n + 2) × F(n – 1), если n > 1.

Чему равно значение выражения (F(2026) – 2 × F(2025)) / F(2024)?

Ответ 4106702

Выразим F(2026) и F(2025) через F(2024): F(2025) = 2027 · F(2024), F(2026) = 2028 · 2027 · F(2024).

(F(2026) – 2·F(2025)) / F(2024) = 2028 · 2027 – 2 · 2027 = 2027 · 2026 = **4106702**.

Программой тоже можно: Python хранит длинные числа точно, но глубина рекурсии больше стандартного предела 1000, его нужно увеличить.

Проверка программой

```
import sys
```

```
sys.setrecursionlimit(10000)

def F(n):
    if n == 1:
        return 1
    return (n + 2) * F(n - 1)

print((F(2026) - 2 * F(2025)) // F(2024))
```

Задание 17

Задание выполняется с использованием прилагаемого файла.

В файле содержится последовательность натуральных чисел. Её элементы могут принимать целые значения от 1 до 100 000 включительно. Определите количество пар последовательности, в которых ровно один из двух элементов является пятизначным числом, а сумма элементов пары не меньше максимального элемента последовательности, запись которого оканчивается на 64.

В ответе запишите количество найденных пар, затем максимальную из сумм элементов таких пар. В данной задаче под парой подразумевается два идущих подряд элемента последовательности.

Файл к заданию: 17.txt (48 КБ)

Ответ 443 109648

Сначала находим максимальный элемент, оканчивающийся на 64: 96564. Число 100 000 шестизначное, поэтому пятизначность проверяем двойным неравенством. Условие «ровно один из двух» удобно записать как сумму логических значений, равную 1.

Программа на Python

```
a = [int(x) for x in open('17.txt')]
m = max(x for x in a if x % 100 == 64)

def five(x):
    return 10000 <= x <= 99999

c, best = 0, 0
for i in range(len(a) - 1):
    x, y = a[i], a[i + 1]
    if five(x) + five(y) == 1 and x + y >= m:
        c += 1
        best = max(best, x + y)
print(c, best)
```

Ответ: 443 109648.

Задание 18

Задание выполняется с использованием прилагаемого файла.

Квадрат разлинован на N × N клеток (1 < N < 30). Исполнитель Робот может перемещаться по клеткам, выполняя за одно перемещение одну из двух команд: вправо или вниз. По команде вправо Робот перемещается в соседнюю правую клетку, по команде вниз — в соседнюю нижнюю. Квадрат ограничен внешними стенами. Между соседними клетками квадрата также могут быть внутренние стены. Сквозь стену Робот пройти не может.

Перед каждым запуском Робота в каждой клетке квадрата лежит монета достоинством от 1 до 100. Посетив клетку, Робот забирает монету с собой; это также относится к начальной и конечной клеткам маршрута Робота.

Определите максимальную и минимальную денежные суммы, которые может собрать Робот, пройдя из левой верхней клетки в правую нижнюю. Маршруты, которые заканчиваются в тупике (клетке со стенами справа и снизу, отличной от правой нижней), не учитываются. В ответе укажите два числа — сначала максимальную сумму, затем минимальную.

Исходные данные представляют собой электронную таблицу размером N × N (в файле N = 18), каждая ячейка которой соответствует клетке квадрата. Внутренние и внешние стены обозначены утолщёнными линиями.

Файл к заданию: 18.xlsx (7 КБ)

Ответ 2381 1249

В электронной таблице. Рядом с полем (например, начиная со столбца W) постройте таблицу того же размера, где в каждой клетке — максимальная сумма, с которой Робот может в неё прийти. В левой

верхней клетке — значение монеты, в первой строке `=X1+B1`, в первом столбце `=W1+A2`, в остальных клетках `=МАКС(W2;X1)+B2`. Затем исправьте формулы у клеток рядом со стенами: если слева стена, берите только клетку сверху, если сверху — только слева. Формулы удобнее всего протянуть по всему полю и поправить вручную клетки у стен — их немного. Так же постройте таблицу минимумов с функцией МИН.

В этом поле есть тупики (3 шт.): в них Робот может попасть, но выйти не может, и маршрут через них до правой нижней клетки не дойдёт. Формулы учитывают это сами: если у клетки оба соседа отрезаны стенами, в неё прийти нельзя — оставьте её пустой и не ссылайтесь на неё.

Ответ — значения в правой нижней клетке таблиц максимумов и минимумов: 2381 1249.

Проверка дома программой (нужна библиотека openpyxl, на экзамене её нет)

```
import openpyxl

ws = openpyxl.load_workbook('18.xlsx').active
n = ws.max_row
a = [[ws.cell(i + 1, j + 1).value for j in range(n)] for i in range(n)]

def thick(side):
    return side is not None and side.style in ('thick', 'medium')

def wall_right(i, j):
    return j == n - 1 or thick(ws.cell(i + 1, j + 1).border.right) or thick(ws.cell(i + 1, j + 2).border.le

def wall_down(i, j):
    return i == n - 1 or thick(ws.cell(i + 1, j + 1).border.bottom) or thick(ws.cell(i + 2, j + 1).border.t

mx = [[None] * n for _ in range(n)]
mn = [[None] * n for _ in range(n)]
mx[0][0] = mn[0][0] = a[0][0]
for i in range(n):
    for j in range(n):
        if i == j == 0:
            continue
        came = []
        if j > 0 and not wall_right(i, j - 1) and mx[i][j - 1] is not None:
            came.append((mx[i][j - 1], mn[i][j - 1]))
        if i > 0 and not wall_down(i - 1, j) and mx[i - 1][j] is not None:
            came.append((mx[i - 1][j], mn[i - 1][j]))
        if came:
            mx[i][j] = max(c[0] for c in came) + a[i][j]
            mn[i][j] = min(c[1] for c in came) + a[i][j]
print(mx[n - 1][n - 1], mn[n - 1][n - 1])
```

Задание 19

Два игрока, Петя и Ваня, играют в следующую игру. Перед игроками лежат две кучи камней. Игроки ходят по очереди, первый ход делает Петя. За один ход игрок может:

– добавить в одну из куч (по своему выбору) один камень;
– увеличить количество камней в одной из куч (по своему выбору) в 2 раза.

Для того чтобы делать ходы, у каждого игрока есть неограниченное количество камней. Например, пусть в одной куче 10 камней, а в другой 5 камней; такую позицию в игре обозначим (10, 5). Тогда за один ход можно получить любую из четырёх позиций: (11, 5), (20, 5), (10, 6), (10, 10).

Игра завершается в тот момент, когда суммарное количество камней в двух кучах становится не менее 58. Победителем считается игрок, сделавший последний ход, то есть первым получивший такую позицию, при которой в кучах суммарно 58 камней или больше. В начальный момент в первой куче было 12 камней, во второй куче — S камней; $1 \le S \le 45$.

Будем говорить, что игрок имеет выигрышную стратегию, если он может выиграть при любых ходах противника.

Известно, что Ваня выиграл своим первым ходом после неудачного первого хода Пети. Укажите минимальное значение S, при котором такая ситуация возможна.

Ответ 10

Ване нужно, чтобы после хода Пети он смог одним ходом довести сумму до 58. Самый сильный ход — удвоить бóльшую кучу. Ищем наименьшее S, при котором у Пети есть ход, после которого (не выиграв сам) он оставляет такую позицию.

Перебор программой (функция game возвращает, кто и каким по счёту ходом выигрывает) даёт S = **10**.

Программа для заданий 19–21

```
from functools import lru_cache

def moves(h):
    a, b = h
    return [(a + 1, b), (a * 2, b), (a, b + 1), (a, b * 2)]

@lru_cache(None)
def game(h):
    # >0: игрок, который ходит, выигрывает этим по счёту своим ходом
    # <0: игрок, который ходит, проигрывает; соперник выигрывает этим по счёту ходом
    #  0: игра уже окончена (выиграл предыдущий игрок)
    if sum(h) >= 58:
        return 0
    res = [game(m) for m in moves(h)]
    lose = [r for r in res if r <= 0]
    if lose:
        return 1 + min(-r for r in lose)
    return -max(res)

# 19: Петя сделал ход (не выигрышный), после которого Ваня выигрывает первым ходом
print(min(s for s in range(1, 46)
          if any(sum(m) < 58 and game(m) == 1 for m in moves((12, s)))))
```

Задание 20

Для игры, описанной в задании 19, найдите два наименьших значения S, при которых у Пети есть выигрышная стратегия, причём одновременно выполняются два условия:

– Петя не может выиграть за один ход;
– Петя может выиграть своим вторым ходом независимо от того, как будет ходить Ваня.

Найденные значения запишите в ответе в порядке возрастания.

Ответ 9 22

Нужны S, для которых game возвращает 2: Петя выигрывает своим вторым ходом и не может выиграть первым.

Все такие S: 9, 22. Два наименьших: 9 и 22.

Программа на Python

```
from functools import lru_cache

def moves(h):
    a, b = h
    return [(a + 1, b), (a * 2, b), (a, b + 1), (a, b * 2)]

@lru_cache(None)
def game(h):
    # >0: игрок, который ходит, выигрывает этим по счёту своим ходом
    # <0: игрок, который ходит, проигрывает; соперник выигрывает этим по счёту ходом
    #  0: игра уже окончена (выиграл предыдущий игрок)
    if sum(h) >= 58:
        return 0
    res = [game(m) for m in moves(h)]
    lose = [r for r in res if r <= 0]
    if lose:
        return 1 + min(-r for r in lose)
    return -max(res)

print(*[s for s in range(1, 46) if game((12, s)) == 2][:2])
```

Задание 21

Для игры, описанной в задании 19, найдите наименьшее значение S, при котором одновременно выполняются два условия:

– у Вани есть выигрышная стратегия, позволяющая ему выиграть первым или вторым ходом при любой игре Пети;
– у Вани нет стратегии, которая позволит ему гарантированно выиграть первым ходом.

Ответ 21

Нужны S, для которых game возвращает –2: Петя проигрывает, а Ваня при лучшей игре Пети выигрывает своим вторым ходом. Наименьшее такое S — **21**.

Программа на Python

```
from functools import lru_cache

def moves(h):
    a, b = h
    return [(a + 1, b), (a * 2, b), (a, b + 1), (a, b * 2)]

@lru_cache(None)
def game(h):
    # >0: игрок, который ходит, выигрывает этим по счёту своим ходом
    # <0: игрок, который ходит, проигрывает; соперник выигрывает этим по счёту ходом
    #  0: игра уже окончена (выиграл предыдущий игрок)
    if sum(h) >= 58:
        return 0
    res = [game(m) for m in moves(h)]
    lose = [r for r in res if r <= 0]
    if lose:
        return 1 + min(-r for r in lose)
    return -max(res)

print(min(s for s in range(1, 46) if game((12, s)) == -2))
```

Задание 22

Задание выполняется с использованием прилагаемого файла.

В файле содержится информация о совокупности N вычислительных процессов, которые могут выполняться параллельно или последовательно. Приостановка выполнения процесса не допускается. Будем говорить, что процесс B зависит от процесса A, если для выполнения процесса B необходимы результаты выполнения процесса A. В этом случае процессы A и B могут выполняться только последовательно.

Информация о процессах представлена в файле в виде таблицы. В первом столбце таблицы указан идентификатор процесса (ID), во втором столбце таблицы — время его выполнения в миллисекундах, в третьем столбце перечислены с разделителем «;» ID процессов, от которых зависит данный процесс. Если процесс независимый, то в таблице указано значение 0.

Определите минимальное время (в мс), через которое завершится выполнение всей совокупности процессов, при условии, что все независимые друг от друга процессы могут выполняться параллельно.

Типовой пример организации данных в файле:

ID процесса B	Время выполнения процесса B (мс)	ID процесса(-ов) A
1	3	0
2	4	1
3	2	2;4
4	5	0
5	8	1;4
6	3	1

В этом примере процессы завершатся через 13 мс: процесс 4 закончится к 5-й мс, процесс 5 начнётся после него и займёт ещё 8 мс.

Типовой пример имеет иллюстративный характер. Для выполнения задания используйте данные из прилагаемого файла.

Файл к заданию: 22.xlsx (6 КБ)

Ответ 58

Каждый процесс начинается, как только закончатся все процессы, от которых он зависит: окончание = max(окончаний зависимостей) + время процесса. Ответ — наибольшее время окончания.

В электронной таблице в столбце D считают время окончания: для независимых `=B2`, для зависимых — `=МАКС(D…;D…)+B…` со ссылками на строки процессов из столбца C.

Программой. Сохраните лист как CSV с разделителем «;».

Программа на Python

```
import csv

rows = list(csv.reader(open('22.csv', encoding='utf-8'), delimiter=';'))[1:]
time, deps = {}, {}
for pid, t, d in rows:
    time[int(pid)] = int(t)
    deps[int(pid)] = [] if d == '0' else [int(x) for x in d.split(';')]

finish = {}
def fin(p):                     # момент окончания процесса p
    if p not in finish:
        finish[p] = max([fin(q) for q in deps[p]], default=0) + time[p]
    return finish[p]

print(max(fin(p) for p in time))
```

Ответ: **58**.

Задание 23

Исполнитель преобразует число на экране. У исполнителя есть три команды, которые обозначены латинскими буквами:

A. Прибавить 2
B. Прибавить 3
C. Умножить на 2

Первая команда увеличивает число на экране на 2, вторая увеличивает его на 3, третья умножает на 2. Программа для исполнителя — это последовательность команд.

Сколько существует программ, для которых при исходном числе 5 результатом является число 32, и при этом траектория вычислений содержит число 12 и не содержит числа 19?

Траектория вычислений программы — это последовательность результатов выполнения всех команд программы. Например, для программы CAB при исходном числе 7 траектория будет состоять из чисел 14, 16, 19.

Ответ 292

Траектория проходит через 12, поэтому количество программ = (число программ 5 → 12) · (число программ 12 → 32). Число 19 запрещено — функция возвращает 0, если траектория в него попала.

Программа на Python

```
def f(x, y):
    if x > y or x == 19:
        return 0
    if x == y:
        return 1
    return f(x + 2, y) + f(x + 3, y) + f(x * 2, y)

print(f(5, 12) * f(12, 32))
```

Ответ: **292**.

Задание 24

Задание выполняется с использованием прилагаемого файла.

Текстовый файл состоит из заглавных букв латинского алфавита. Определите максимальное количество идущих подряд символов, среди которых никакие две соседние буквы не совпадают, а буква Z встречается не более одного раза.

В ответе укажите количество символов.

Файл к заданию: 24.txt (244 КБ)

Ответ 214

Метод двух указателей: правая граница идёт по строке, левая сдвигается, когда окно перестаёт удовлетворять условию. Если две соседние буквы совпали, ни одно окно не может их содержать — начинаем окно с текущей буквы. Если букв Z стало больше 1, сдвигаем левую границу, пока одна из них не выйдет из окна.

Программа на Python

```
s = open('24.txt').read().strip()
best = 0
```

```
left = 0          # начало текущего окна
k = 0             # сколько букв Z в окне
for r in range(len(s)):
    if r > 0 and s[r] == s[r - 1]:
        left, k = r, 0               # две одинаковые соседние: окно начинаем заново
    if s[r] == 'Z':
        k += 1
    while k > 1:                     # лишняя Z — сдвигаем левую границу
        if s[left] == 'Z':
            k -= 1
        left += 1
    best = max(best, r - left + 1)
print(best)
```

Ответ: **214**.

Задание 25

Назовём маской числа последовательность цифр, в которой также могут встречаться следующие символы:

– символ «?» означает ровно одну произвольную цифру;
– символ «*» означает любую последовательность цифр произвольной длины; в том числе «*» может задавать и пустую последовательность.

Например, маске 123*4?5 соответствуют числа 123405 и 12300405.

Среди натуральных чисел, не превышающих 10^{10}, найдите все числа, соответствующие маске 7*12?3?9, делящиеся на 27761 без остатка.

В ответе запишите в первом столбце таблицы все найденные числа в порядке возрастания, а во втором столбце — соответствующие им результаты деления этих чисел на 27761.

Количество строк в таблице для ответа избыточно.

Ответ 7281127319 262279
7555128389 272149
7560125369 272329
7565122349 272509
7844120399 282559

Перебирать все числа до 10^{10} слишком долго, а кратных 27761 — около 360 тысяч, это быстро. Модуль fnmatch понимает маски с «?» и «*» в том же смысле, что и в условии.

Другой способ — перебрать сами числа, подходящие под маску (вместо «*» — несколько цифр, вместо «?» — любая цифра), и проверить делимость.

Программа на Python

```
from fnmatch import fnmatch

for x in range(27761, 10 ** 10 + 1, 27761):       # перебираем только кратные 27761
    if fnmatch(str(x), '7*12?3?9'):
        print(x, x // 27761)
```

Ответ:
7281127319 262279
7555128389 272149
7560125369 272329
7565122349 272509
7844120399 282559

Задание 26

Задание выполняется с использованием прилагаемого файла.

Системный администратор раз в неделю создаёт архив пользовательских файлов. Однако объём диска, куда он помещает архив, может быть меньше, чем суммарный объём архивируемых файлов. Известно, какой объём занимает файл каждого пользователя.

По заданной информации об объёме файлов пользователей и свободном объёме на архивном диске определите максимальное число пользователей, чьи файлы можно сохранить в архиве, а также максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.

Входные данные. В первой строке входного файла находятся два числа: S — размер свободного места на диске (натуральное число, не превышающее 100 000 000) и N — количество пользователей (натуральное число, не превышающее 10 000). В следующих N строках находятся значения объёмов файлов каждого пользователя (все числа натуральные, не превышающие 10 000), каждое в отдельной строке.

Выходные данные. Запишите в ответе два числа: сначала наибольшее число пользователей, чьи файлы могут быть помещены в архив, затем максимальный размер имеющегося файла, который может быть сохранён в архиве, при условии, что сохранены файлы максимально возможного числа пользователей.

Типовой пример организации данных во входном файле:

```
100 4
80
30
50
40
```

При таких исходных данных можно сохранить файлы максимум двух пользователей. Возможные объёмы этих двух файлов 30 и 40, 30 и 50 или 40 и 50. Наибольший объём файла из перечисленных пар — 50, поэтому ответ для приведённого примера: 2 50.

Файл к заданию: 26.txt (22 КБ)

Ответ 1341 3554

Чтобы сохранить как можно больше файлов, берём самые маленькие: сортируем и набираем, пока помещаются. Так получаем наибольшее количество k.

Чтобы в архиве оказался как можно больший файл, оставляем k – 1 самых маленьких, а вместо k-го берём наибольший файл, который помещается в оставшееся место.

Программа на Python

```
f = open('26.txt')
s, n = map(int, f.readline().split())
a = sorted(int(f.readline()) for _ in range(n))

total, k = 0, 0
while k < n and total + a[k] <= s:     # берём самые маленькие файлы
    total += a[k]
    k += 1

# заменяем самый большой из взятых файлов на наибольший, который ещё помещается
free = s - (total - a[k - 1])
best = max(x for x in a[k - 1:] if x <= free)
print(k, best)
```

Ответ: 1341 3554.

Задание 27

Задание выполняется с использованием прилагаемого файла.

Рассматриваются частицы на плоскости, обладающие следующими характеристиками: декартовы координаты, вектор скорости, масса, а также признак, характеризующий внутреннее строение частицы, обозначаемый числами от I до VII (в римской системе счисления).

Учёный решил провести кластеризацию частиц по модулю импульса, то есть разбить их множество на K непересекающихся непустых подмножеств (кластеров), таких, что модуль разности модулей импульса любых двух частиц каждого подмножества не превосходит значения R. Гарантируется, что такое разбиение существует и единственно для заданного R.

Будем называть центром кластера такую его частицу, для которой сумма модулей разности модулей импульса со всеми остальными частицами этого кластера минимальна. Для каждого кластера гарантируется единственность его центра.

В каждой строке текстового файла хранится информация об одной частице: координаты x и y, проекции вектора скорости V_x и V_y, масса m и признак. Значения в строке разделяются одним или несколькими пробелами и/или символами табуляции, дробная часть отделяется точкой. Количество строк в файле не превышает 10 000. Абсолютная величина каждого числового значения не превышает 100,0.

Известно, что все описанные в файле частицы подразделяются ровно на три кластера (K = 3) с R = 1,5 для каждого.

Для каждого кластера определите его центр, затем найдите два числа: Q1 — наименьшее евклидово расстояние между центрами кластеров, и Q2 — минимальное значение модуля импульса среди центров

кластеров.

В ответе запишите два числа: сначала целую часть произведения Q1 × 10 000, затем целую часть произведения Q2 × 10 000.

Для справки. Модуль импульса p частицы массы m, обладающей скоростью v = (V_x, V_y), вычисляется по формуле: $p = m\sqrt{(V_x^2 + V_y^2)}$.

Евклидово расстояние между двумя точками на плоскости A(x_1, y_1) и B(x_2, y_2) вычисляется по формуле: $d(A,B) = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)}$.

Типовой пример организации данных во входном файле (три строки файла для трёх частиц):

```
0.67  -2.14  3.0  -4.0  0.2  V
3.14  7.22  3.2  4.3  0.7  II
1.33  5.56  0.00  5.22  0.456  IV
```

Для частицы из первой строки примера модуль импульса равен 1,0.

Типовой пример имеет иллюстративный характер. Для выполнения задания используйте данные из прилагаемого файла.

Файл к заданию: 27.txt (72 КБ)

Ответ 251054 35115

Посчитаем для каждой частицы модуль импульса и отсортируем частицы по этой величине. Разбиение единственно: внутри кластера соседние значения отличаются не больше чем на R, а между кластерами разрыв больше R. Поэтому кластеры получаются разрезанием отсортированного списка там, где разность соседних значений больше 1,5.

Центр кластера ищем по определению — перебором сумм модулей разностей (это медиана кластера). Затем считаем Q1 и Q2 по условию.

Программа на Python

```
from math import dist, sqrt

parts = []
for line in open('27.txt'):
    x, y, vx, vy, m, f = line.split()
    x, y, vx, vy, m = map(float, (x, y, vx, vy, m))
    parts.append((m * sqrt(vx ** 2 + vy ** 2), x, y, f))     # величина, координаты, признак

parts.sort()                                # по величине
clusters, cur = [], [parts[0]]
for p in parts[1:]:
    if p[0] - cur[-1][0] > 1.5:             # разрыв больше R — новый кластер
        clusters.append(cur)
        cur = []
    cur.append(p)
clusters.append(cur)
print(len(clusters))                        # должно быть 3

# центр: минимальная сумма модулей разностей
centers = [min(c, key=lambda p: sum(abs(p[0] - r[0]) for r in c)) for c in clusters]
q1 = min(dist(a[1:3], b[1:3]) for i, a in enumerate(centers) for b in centers[i + 1:])
q2 = min(p[0] for p in centers)
print(int(q1 * 10000), int(q2 * 10000))
```

Ответ: 251054 35115.