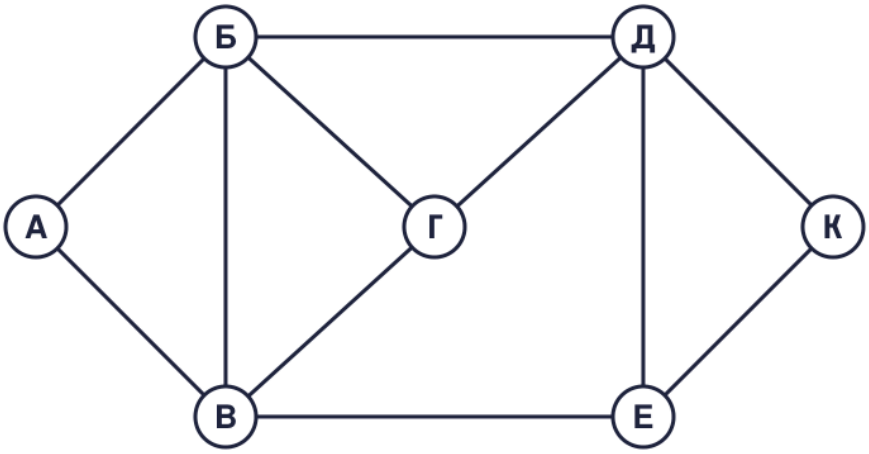
ЕГЭНИЙ · Информатика

Вариант 9 · 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		31	45		23		18
2	31				28	30	41
3	45			24		29	
4			24			17	
5	23	28					
6		30	29	17			56
7	18	41				56	



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 ВГ отличается от длины дороги БГ.

В ответе запишите целое число.

Ответ 23

Посчитаем степени вершин (количество дорог из пункта). На графе: А — 2, Б — 4, В — 4, Г — 3, Д — 4, Е — 3, К — 2. В таблице: П1 — 4, П2 — 4, П3 — 3, П4 — 2, П5 — 2, П6 — 4, П7 — 3.

Сопоставляем вершины по степеням и по соседям: А — вершина степени 2, степени её соседей: 4, 4; так устроен только П5; Б — вершина степени 4, степени её соседей: 2, 3, 4, 4; так устроен только П2; В — вершина степени 4, соединённая с А, Б; этому соответствует только П1; Г — вершина степени 3, соединённая с Б, В; этому соответствует только П7.

Дорога ВГ — это П1–П7, её длина 18 км. Дорога БГ — это П2–П7, её длина 41 км.

Разность: $|18 - 41| = \mathbf{23}$.

Задание 2

Миша заполнял таблицу истинности логической функции F

¬(x → z) ∨ (y ≡ w) ∨ (w ∧ ¬x),

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	1		1	0
0		0		0
	0	0		0

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 wzxy

Функция равна 0 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 0.

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 (not (x <= z)) or (y == w) or (w and not x)

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

Программа выводит единственную перестановку: **wzxy**.

Задание 3

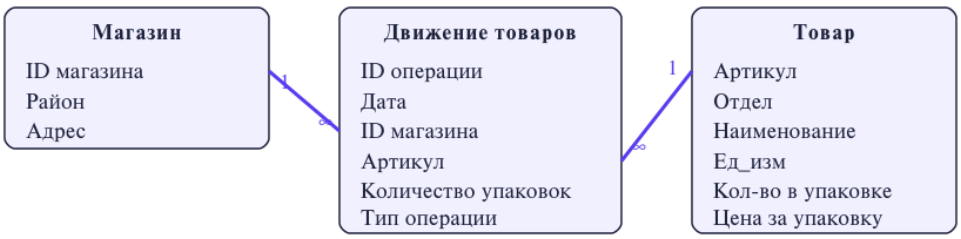
Задание выполняется с использованием прилагаемого файла.

В файле приведён фрагмент базы данных «Продукты» 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 общую выручку (в рублях) от продажи гречки в магазинах Центрального района за период с 10 по 20 июня включительно. В ответе запишите только число.

Файл к заданию: 3.xlsx (97 КБ)

Ответ 16215

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)` и цену за упаковку (столбец 6 листа «Товар»).

Включите автофильтр и оставьте: тип операции — Продажа, даты с 10.06.2026 по 20.06.2026, район — Центральный, наименование начинается со слова «Гречка». Выручка строки = количество упаковок · цена за упаковку; цена указана за упаковку, поэтому количество товара в упаковке не нужно. 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 typ == 'Продажа' and 10 <= day <= 20 and shops[shop] == 'Центральный' and g[2].startswith('Гречка'):
        total += int(cnt) * int(g[5])      # упаковки * цена за упаковку
print(total)
```

Ответ: **16215**.

Задание 4

Для кодирования некоторой последовательности, состоящей из букв А, Б, В, Г, Д, Е, решили использовать неравномерный двоичный код, удовлетворяющий условию Фано. Для букв А, Б, В, Г использовали такие кодовые слова: А – 01, Б – 11, В – 001, Г – 0000.

Укажите кратчайшее возможное кодовое слово для буквы Д, при котором код будет удовлетворять условию Фано и для буквы Е останется возможность подобрать кодовое слово. Если таких кодов несколько, укажите код с наименьшим числовым значением.

Примечание. Условие Фано означает, что никакое кодовое слово не является началом другого кодового слова.

Ответ 10

Нарисуем двоичное дерево известных кодов. Свободные вершины, с которых может начинаться новый код: 0001, 10.

Код для Д должен начинаться с одной из свободных вершин, и после этого для Е должна остаться хотя бы одна свободная ветвь. Если свободная вершина единственная, код Д должен быть на один знак длиннее, чтобы вторая половина ветви досталась букве Е. Самый короткий подходящий код с наименьшим числовым значением — **10**.

Проверка перебором

```
from itertools import product

known = ['01', '11', '001', '0000']
words = [''.join(t) for n in range(1, 7) 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)

good = []
for d in words:
    if d not in known and fano(known + [d]):
        if any(e not in known + [d] and fano(known + [d, e]) for e in words):
            good.append(d)
good.sort(key=lambda w: (len(w), int(w, 2)))
print(good[0])
```

Ответ: **10**.

Задание 5

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

1. Строится двоичная запись числа N.

2. Далее эта запись обрабатывается по следующему правилу:
а) к этой записи дописывается справа бит чётности: 0, если в двоичной записи числа N чётное количество единиц, и 1, если нечётное;
б) к полученной записи справа дописывается ещё один бит чётности, вычисленный уже для новой записи.

Полученная таким образом запись является двоичной записью искомого числа R.

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

Например, для исходного числа $6_{10} = 110_2$ результатом является число $11000_2 = 24_{10}$, а для исходного числа $19_{10} = 10011_2$ результатом является число $1001110_2 = 78_{10}$.

Укажите минимальное число R, большее 109, которое может быть получено с помощью описанного алгоритма. В ответе запишите это число в десятичной системе счисления.

Ответ 114

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 R(n):
    b = bin(n)[2:]
    b = b + str(b.count('1') % 2)
    b = b + str(b.count('1') % 2)
    return int(b, 2)

print(min(R(n) for n in range(1, 1000) if R(n) > 109))
```

Ответ: **114**.

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 6 [Вперёд 21 Направо 90 Вперёд 16 Направо 90]
Поднять хвост
Назад 11 Налево 90 Вперёд 2 Направо 90
Опустить хвост
Повтори 4 [Вперёд 24 Направо 90 Вперёд 13 Направо 90]

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

Ответ 168

Первый цикл рисует прямоугольник: Черепаха идёт вверх на 21, поворачивает направо и идёт на 16. Получается прямоугольник $0 \le x \le 16$, $0 \le y \le 21$.

Перемещение без рисования приводит Черепаху в точку (-2; -11), голова снова смотрит вверх.

Второй цикл: вверх на 24, затем вправо на 13. Второй прямоугольник: $-2 \le x \le 11$, $-11 \le y \le 13$.

Пересечение: $0 \le x \le 11$, $0 \le y \le 13$. Целых точек: $(11 - (0) + 1) \cdot (13 - (0) + 1) = 12 \cdot 14 =$ **168**.

Проверка программой

```
A = (0, 16, 0, 21)      # x1, x2, y1, y2
B = (-2, 11, -11, 13)
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

Для хранения произвольного растрового изображения размером 1920×1080 пикселей отведено 1867 Кбайт памяти без учёта размера заголовка файла. Для кодирования цвета каждого пикселя используется одинаковое количество бит, коды пикселей записываются в файл один за другим без промежутков.

Какое максимальное количество цветов можно использовать в изображении?

Ответ 128

Всего бит: 1867 · 1024 · 8 = 15 294 464. Пикселей: 1920 · 1080 = 2 073 600.

На пиксель приходится не больше 15 294 464 / 2 073 600 ≈ 7,38 бита, то есть 7 бит (целое число, округляем вниз).

Количество цветов: 2^7 = **128**.

Задание 8

Определите количество пятизначных чисел, записанных в восьмеричной системе счисления, в записи которых цифра 6 встречается ровно один раз, а никакие две нечётные цифры не стоят рядом.

Ответ 4713

Переберём все строки из 5 цифр восьмеричной системы (0–7), отбросим начинающиеся с нуля и проверим условие. Всего строк 8^5 = 32 768 — перебор мгновенный.

Программа на Python

```
from itertools import product

c = 0
for t in product('01234567', repeat=5):
    if t[0] != '0' and t.count('6') == 1 and not any(int(a) % 2 == 1 and int(b) % 2 == 1 for a, b in zip(t,
        c += 1
print(c)
```

Ответ: **4713**.

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;
– среднее арифметическое трёх наибольших чисел строки меньше удвоенного среднего арифметического четырёх остальных её чисел.

В ответе запишите только число.

Файл к заданию: 9.xlsx (191 КБ)

Ответ 1284

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 sum(a[4:]) / 3 < 2 * sum(a[:4]) / 4:
            c += 1
print(c)
```

Ответ: **1284**.

Задание 10

Файл 10-3.odt содержит тексты рассказов А. П. Чехова «Дама с собачкой» и «Ионыч». Определите, сколько раз в тексте главы III рассказа «Ионыч» (от строки «III» после заголовка «ИОНЫЧ» до строки «IV») встречается слово «что», записанное со строчной или с прописной буквы, 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 10

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('10-3.txt', encoding='utf-8').read()
t = t[t.index('ИОНЫЧ'):]
t = t[t.index('\nIII\n'):t.index('\nIV\n')]
print(word(t, 'что', re.IGNORECASE))
```

Задание 11

При регистрации в компьютерной системе каждому объекту сопоставляется идентификатор из 30 символов, для записи которого используется алфавит мощностью 201. В базе данных для хранения каждого идентификатора отведено одинаковое и минимально возможное целое число байт, все символы кодируются одинаковым и минимально возможным количеством бит.

Кроме идентификатора, для каждого объекта в базе хранятся дополнительные сведения, для которых отведено одинаковое для всех объектов целое число байт. Для хранения сведений о 80 объектах потребовалось 4320 байт. Сколько байт выделено для хранения дополнительных сведений об одном объекте? В ответе запишите только целое число.

Ответ 24

Символов 201: $2^7 < 201 \le 2^8$ — по 8 бит на символ. Идентификатор: $30 \cdot 8 = 240$ бит → 30 байт (округляем вверх).

На один объект: 4320 / 80 = 54 байт. Дополнительные сведения: 54 – 30 = **24**.

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 3559 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, L, q2	0, L, q3
q2	1, S, q2	1, S, q2	0, L, q2
q3	0, L, q2	0, L, q2	1, L, q2

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

Ответ 3562

$3559 = 110111100111_2$.

q0 сдвигает головку на последнюю цифру. q1 прибавляет 1 к младшему разряду: 0 → 1 и переход в q2, 1 → 0 и переход в q3 (перенос). q2 прибавляет 1 к текущему разряду (на 1 пишет 0 и идёт влево с

переносом). q3 пропускает текущий разряд и переходит в q2 на следующем — то есть добавляет единицу в разряд двоек. Итого программа прибавляет 1 + 2 = 3.

Проверим, проследив такты программой.

Программа на Python

```
prog = {
    'q0': {'λ': ('λ', 'L', 'q1')},
    'q1': {'0': ('1', 'L', 'q2'), '1': ('0', 'L', 'q3')},
    'q2': {'λ': ('1', 'S', 'q2'), '0': ('1', 'S', 'q2'), '1': ('0', 'L', 'q2')},
    'q3': {'λ': ('0', 'L', 'q2'), '0': ('0', 'L', 'q2'), '1': ('1', 'L', 'q2')},
}
tape = dict(enumerate('110111100111'))
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 110111101010_2 = **3562**.

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 172.110.218.168 и маска 255.255.192.0. 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 474

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

$218 = 11011010_2$, $192 = 11000000_2$. Поразрядная конъюнкция: $11000000_2 = 192$.

Адрес сети: 172.110.192.0. Сумма октетов: 172 + 110 + 192 + 0 = **474**.

Проверка программой

```
a = [172, 110, 218, 168]
m = [255, 255, 192, 0]
net = [x & y for x, y in zip(a, m)]
print(net, sum(net))
```

Задание 14

Значение арифметического выражения

25^(163)+5^(284)−5^(4)−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 44 нуля.

В ответе запишите число в десятичной системе счисления.

Ответ 25

25163=5326: в пятеричной записи это единица и 326 нулей. Слагаемое 5284 добавляет ещё одну единицу, а вычитание 54+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 ** 163 + 5 ** 284 - 5 ** 4 - x, 5) == 44:
        print(x)
        break
```

Ответ: **25**.

Задание 15

Обозначим через m & n поразрядную конъюнкцию неотрицательных целых чисел m и n. Так, например, 14 & 5 = 1110_2 & 0101_2 = 0100_2 = 4.

Для какого наименьшего неотрицательного целого числа A формула

((x & 56 ≠ 0) ∧ (x & 8 = 0)) → (x & A ≠ 0)

тождественно истинна (то есть принимает значение 1 при любом неотрицательном целом значении переменной x)?

Ответ 48

56 = 111000_2, 8 = 1000_2. Посылка истинна, когда в x нет ни одного бита числа 8, но есть хотя бы один бит числа 56 — значит, хотя бы один из битов 16, 32 (биты 56, которых нет в 8).

x может состоять ровно из одного такого бита, поэтому A должно содержать их все. Наименьшее A = 16 + 32 = **48**.

Проверка программой

```
def F(x, A):
    return (not ((x & 56) != 0 and (x & 8) == 0)) or (x & A) != 0

for A in range(0, 1024):
    if all(F(x, A) for x in range(0, 1024)):
        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) = F(n / 2) + 1, если n > 1 и n чётное;
F(n) = F(n – 1) + 2n, если n > 1 и n нечётное.

Определите количество натуральных значений n из отрезка [1; 500], для которых значение F(n) кратно 11.

Ответ 46

Глубина рекурсии здесь небольшая (каждые два шага n уменьшается примерно вдвое), поэтому функцию можно вычислить напрямую. Декоратор lru_cache запоминает уже вычисленные значения и ускоряет перебор.

Программа на Python

```
from functools import lru_cache

@lru_cache(None)
def F(n):
    if n == 1:
        return 1
    if n % 2 == 0:
        return F(n // 2) + 1
    return F(n - 1) + 2 * n

print(sum(1 for n in range(1, 501) if F(n) % 11 == 0))
```

Ответ: **46**.

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 остаток от деления на 33 хотя бы одного из элементов равен минимальному элементу последовательности.

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 (57 КБ)

Ответ 576 196426

Минимальный элемент последовательности равен 4. Для каждой пары соседних элементов проверяем остатки от деления на 33.

Программа на Python

```
a = [int(x) for x in open('17.txt')]
m = min(a)
c, best = 0, 0
for i in range(len(a) - 1):
    x, y = a[i], a[i + 1]
    if x % 33 == m or y % 33 == m:
        c += 1
        best = max(best, x + y)
print(c, best)
```

Ответ: 576 196426.

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.

В «угловых» клетках поля — тех, которые справа и снизу ограничены стенами, Робот не может продолжать движение, поэтому накопленная сумма считается итоговой. Таких конечных клеток на поле может быть несколько, включая правую нижнюю клетку поля. При разных запусках итоговые накопленные суммы могут различаться.

Определите максимальную и минимальную денежные суммы среди всех возможных итоговых сумм, которые может собрать Робот, пройдя из левой верхней клетки в конечную клетку маршрута. В ответе укажите два числа — сначала максимальную сумму, затем минимальную.

Исходные данные представляют собой электронную таблицу размером N × N (в файле N = 16), каждая ячейка которой соответствует клетке квадрата. Внутренние и внешние стены обозначены утолщёнными линиями.

Файл к заданию: 18.xlsx (6 КБ)

Ответ 2075 578

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.

Конечные клетки — те, у которых стены и справа, и снизу. В этом поле их 6: M5, J9, N11, B16, J16, P16. Ответ — максимум из таблицы максимумов и минимум из таблицы минимумов по всем этим клеткам. Частая ошибка — взять только правую нижнюю клетку.

Ответ: 2075 578.

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]
ends = [(i, j) for i in range(n) for j in range(n)
        if wall_right(i, j) and wall_down(i, j) and mx[i][j] is not None]
print(max(mx[i][j] for i, j in ends), min(mn[i][j] for i, j in ends))
```

Задание 19

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

– добавить в кучу один камень;
– добавить в кучу два камня;
– увеличить количество камней в куче в три раза.

Например, имея кучу из 10 камней, за один ход можно получить кучу из 11, 12 или 30 камней. У каждого игрока, чтобы делать ходы, есть неограниченное количество камней.

Игра завершается в тот момент, когда количество камней в куче становится не менее 107. Победителем считается игрок, сделавший последний ход, то есть первым получивший кучу, в которой будет 107 или больше камней. В начальный момент в куче было S камней; $1 \le S \le 106$.

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

Укажите минимальное значение S, при котором Петя не может выиграть за один ход, но при любом ходе Пети Ваня может выиграть своим первым ходом.

Ответ 35

Выиграть одним ходом можно из позиций, где хотя бы один ход даёт 107 или больше камней. Нужна позиция, из которой сам Петя так выиграть не может, а любой его ход ведёт в такую позицию.

Перебор программой (функция game возвращает, кто и каким по счёту ходом выигрывает) даёт S = **35**.

Программа для заданий 19–21

```
from functools import lru_cache

def moves(h):
    return [h + 1, h + 2, h * 3]

@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 h >= 107:
        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, 107) if game(s) == -1))
```

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 33 34

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

Все такие S: 33, 34. Два наименьших: 33 и 34.

Программа на Python

```
from functools import lru_cache

def moves(h):
    return [h + 1, h + 2, h * 3]

@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 h >= 107:
        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, 107) if game(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 32

Нужны S, для которых game возвращает –2: Петя проигрывает, а Ваня при лучшей игре Пети выигрывает своим вторым ходом. Наименьшее такое S — **32**.

Программа на Python

```
from functools import lru_cache

def moves(h):
    return [h + 1, h + 2, h * 3]

@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 h >= 107:
        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, 107) if game(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 (5 КБ)

Ответ 41

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: **41**.

Задание 23

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

A. Прибавить 1
B. Поменять местами
C. Прибавить 11

Первая команда увеличивает число на экране на 1, третья — на 11. Вторая команда применяется только к числу, у которого цифра в разряде десятков по значению меньше цифры, стоящей в разряде единиц, и заменяет число на экране числом, в котором цифры двух младших разрядов поменялись местами (например, 125 → 152).

Программа для исполнителя — это последовательность команд. Сколько существует программ, для которых при исходном числе 21 результатом является число 92?

Траектория вычислений программы — это последовательность результатов выполнения всех команд программы. Например, для программы ABC при исходном числе 13 траектория состоит из чисел 14, 41, 52.

Ответ 132295

Перестановка двух младших цифр при t < u увеличивает число на 9·(u – t), поэтому все команды увеличивают число и рекурсия конечна: как только число превысило цель, путь обрывается.

Вторая команда меняет число так: x – 10·t – u + 10·u + t, где t и u — цифры десятков и единиц.

Программа на Python

```
def f(x, y):
    if x > y:
        return 0
    if x == y:
        return 1
    res = f(x + 1, y) + f(x + 11, y)
    t, u = x // 10 % 10, x % 10        # цифры десятков и единиц
    if t < u:
        res += f(x - 10 * t - u + 10 * u + t, y)
    return res

print(f(21, 92))
```

Ответ: **132295**.

Задание 24

Задание выполняется с использованием прилагаемого файла.

Текстовый файл состоит из цифр 0, 1, 4, 5, 6 и знаков арифметических операций «+» и «-» (сложение и вычитание). Определите максимальное количество символов в непрерывной последовательности, которая является корректным арифметическим выражением с целыми неотрицательными числами. В этом выражении никакие два знака арифметических операций не стоят рядом, в записи чисел отсутствуют незначащие (ведущие) нули и число 0 не имеет знака.

В ответе укажите количество символов.

Файл к заданию: 24.txt (244 КБ)

Ответ 143

Идём по строке и поддерживаем длину cur самого длинного корректного выражения, которое заканчивается на текущем символе. Знак продолжает выражение, если перед ним стоит цифра; два знака подряд обрывают его. Цифра после знака начинает новое число. Цифра после цифры продолжает число, если оно не начинается с нуля; иначе выражение приходится начинать заново с текущей цифры.

Ответ обновляем только на цифрах: корректное выражение не может заканчиваться знаком.

Программа на Python

```
s = open('24.txt').read().strip()
best = cur = 0       # cur — длина самого длинного корректного выражения, оканчивающегося здесь
zero = False         # последнее число в нём начинается с нуля
for i, c in enumerate(s):
    if c in '+-':
        if i > 0 and s[i - 1] not in '+-' and cur > 0:
            cur += 1                    # знак после цифры продолжает выражение
        else:
            cur = 0                     # два знака подряд
    else:
        if i > 0 and s[i - 1] in '+-' and cur > 0:
            cur += 1                    # новое число после знака
```

```
            zero = c == '0'
        elif i > 0 and s[i - 1] not in '+-' and not zero:
            cur += 1                    # продолжаем текущее число
        else:
            cur = 1                     # начинаем выражение заново с этой цифры
            zero = c == '0'
        best = max(best, cur)
print(best)
```

Ответ: **143**.

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 32411 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 32411.

Количество строк в таблице для ответа избыточно.

Ответ 7327122359 226069
7772125389 239799
7819121339 241249

Перебирать все числа до 10^{10} слишком долго, а кратных 32411 — около 308 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(32411, 10 ** 10 + 1, 32411):       # перебираем только кратные 32411
    if fnmatch(str(x), '7*12?3?9'):
        print(x, x // 32411)
```

Ответ:
7327122359 226069
7772125389 239799
7819121339 241249

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 (46 КБ)

Ответ 2312 2996

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: 2312 2996.

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 четыре кластера (K = 4) с R = 1,5 для каждого.

Для каждого кластера определите его центр, затем найдите два числа: Q1 — наибольшее евклидово расстояние между частицами одного кластера, имеющими признак VII, и Q2 — максимальное 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 (93 КБ)

Ответ 1655572 177525

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 4

# 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 = 0
for c in clusters:
    f = [p for p in c if p[3] == 'VII']
    for i in range(len(f)):
        for j in range(i + 1, len(f)):
            q1 = max(q1, dist(f[i][1:3], f[j][1:3]))
q2 = max(p[0] for p in centers)
print(int(q1 * 10000), int(q2 * 10000))
```

Ответ: 1655572 177525.