

Всероссийская олимпиада школьников по информатике 2024-2025

Районный тур, 7-11 класс
Санкт-Петербург, 2 декабря 2024 года

РАЗБОР ЗАДАЧ

Задача А - Жидкокристаллический дисплей

— — —

Изобразить ЖК дисплей с
использованием букв R, G, B

Букв G в два раза больше

Каждая следующая строка сдвинута
на один относительно предыдущей

R	G	B	G	R	G	B	G	R	G
G	B	G	R	G	B	G	R	G	B
B	G	R	G	B	G	R	G	B	G
G	R	G	B	G	R	G	B	G	R
R	G	B	G	R	G	B	G	R	G
G	B	G	R	G	B	G	R	G	B

Задача А - решение

Рассмотрим строку i и столбец j (нумерация с 0)

Если сумма $i + j$ нечетна, выводим G

Если четна:

- Если делится на 4, выводим R
- Если не делится на 4, выводим B

Задача А - код

— — —

```
h = int(input())
w = int(input())

for i in range(h):
    s = ""
    for j in range(w):
        if (i + j) % 2 != 0:
            s += "G"
        elif (i + j) % 4 == 0:
            s += "R"
        else:
            s += "B"
    print(s)
```

Задача В - Весы

Даны четыре гири и чашечные весы

Необходимо уравновесить весы, поставив на них все гири



Задача В - решение

— — —

Есть два способа уравновесить весы:

- Самая тяжелая гиря на одной чаше, три легких на другой
- Самая легкая и самая тяжелая гиря на одной чаше, две средних на другой

В любом другом способе видно, что чаша, содержащая самую тяжелую гирю, перевешивает

Задача В - решение

— — —

Отсортируем гири по возрастанию веса и проверим два описанных варианта

Если не знакомы с алгоритмами сортировки, можно перебрать все варианты распределения гирь по чашкам:

- 4 вида одна против трех
- 3 вида два против двух

Задача В - код

— — —

```
a = [0] * 4
for i in range(4):
    a[i] = int(input())
a.sort()

if a[0] + a[1] + a[2] == a[3]:
    print(*a[:3])
    print(a[3])
elif a[0] + a[3] == a[1] + a[2]:
    print(a[0], a[3])
    print(a[1], a[2])
else:
    print(-1)
```

Задача С - похожие пары

— — —

Числа из n цифр называются похожими, если у них k последних цифр совпадают

Ведущие нули запрещены

Дано n , найти количество похожих пар

Задача С - решение

Разобьем все n -значные числа на классы по последним k цифрам

Всего будет 10^k классов

В каждом классе всего 10^{n-k} чисел, если бы были разрешены ведущие нули. Так как они запрещены, то чисел $9 \cdot 10^{n-k-1}$.

Если в классе s чисел, они образуют $s \cdot (s-1) / 2$ пар.

Особый случай: $k = n$, тогда пар похожих чисел нет.

Задача C - код

```
n = int(input())
```

```
k = int(input())
```

```
if n == k:
```

```
    ans = 0
```

```
else:
```

```
    cnt = 10 ** k
```

```
    cntp = 9 * 10 ** (n - k - 1)
```

```
    ans = cnt * (cntp * (cntp - 1)) // 2
```

```
print(ans)
```

Задача D - интересные числа

— — —

Рассмотрим число x , в котором зафиксированы все цифры, кроме последней

$x = abc...def?$

Заметим, что чтобы x стало интересным, есть ровно один способ добавить последнюю цифру

Значит интересных чисел примерно $n / 10$

Задача D - решение

Интересных чисел примерно $n / 10$

Что значит примерно 🤔?

- Все однозначные числа интересные, учтем их отдельно

Если заменить в n последнюю цифру на цифру, равную первой, то получится интересное число.

- Если оно больше n , ответ $(n - 10) \div 10 + 9$
- Если оно не больше, то ответ на единицу больше

Особый случай $n < 10$, тогда ответ n

Задача D - решение

```
n = int(input())
```

```
if n < 10:
```

```
    ans = n
```

```
else:
```

```
    ans = (n - 10) // 10 + 9
```

```
    if n % 10 >= int(str(n)[0]):
```

```
        ans += 1
```

```
print(ans)
```

Задача E - быстро возрастающее разбиение

— — —

Будем называть разбиение числа на слагаемые быстро возрастающим, если разность между каждой парой соседних слагаемых все больше

$$1 + 2 + 6$$

$$1 + 8$$

$$2 + 7$$

$$3 + 6$$

Найти все быстро возрастающие разбиения

$$4 + 5$$

$$9$$

Задача E - решение

— — —

Быстро возрастающих разбиений мало (но, кстати, все равно экспоненциально много)

Полным перебором найдем их все

Передаем минимальное слагаемое на очередной позиции

Отсекаем ветки перебора, когда оставшаяся сумма слишком мала

Задача E - код

```
void gen(int n, int p, int f, int d) {
    if (n == 0) {
        for (int i = 0; i < p; i++) {
            cout << a[i];
            if (i < p - 1) {
                cout << "+";
            }
        }
        cout << "\n";
        return;
    }
    for (int v = max(1, f + d); v <= n; v++) {
        a[p] = v;
        gen(n - v, p + 1, v, p == 0 ? 1 : v - f + 1);
    }
}
```

Задача F - красивые разбиения

Необходимо разбить первые n простых чисел на два множества, чтобы разность произведений в них была как можно меньше

стандартный ввод	стандартный вывод
1	1
2	2
3	5
4	14
5	42

Задача F - троллинг от жюри

— — —

Забавное наблюдение: первые числа ведут себя как числа Каталана (количество правильных скобочных последовательностей)

Может и дальше будет так?

Нет 

Задача F - решение

Надо перебрать разбиения, но их слишком много, 2^{30} это порядка 10^9

Воспользуемся методом “встреча посередине” (meet in the middle)

Рассмотрим первые $k=n/2$ простых и последние $n-k$ простых

Для каждого 2^k подмножеств первых k простых и каждого 2^{n-k} подмножеств последних $n-k$ простых найдем произведение, сложим в массив

Отсортируем оба массива

Задача F - решение

У нас есть два отсортированных массива произведений:

- $a[0], a[1], \dots, a[2^k-1]$ и $b[0], b[1], \dots, b[2^{n-k}-1]$

Пусть произведение всех n первых простых равно P

Надо найти два элемента в массивах, чтобы их произведение $S=a[i]*b[j]$ минус P/S по модулю было минимально

Используем, например, метод двух указателей

Задача F - решение

- $a[0], a[1], \dots, a[2^k-1]$ и $b[0], b[1], \dots, b[2^{n-k}-1]$

Надо найти два элемента в массивах, чтобы их произведение $S=a[i]*b[j]$ минус P/S по модулю было минимально

Используем, например, метод двух указателей

- Первый указатель перебирает i
- Второй указатель идет по уменьшению j , пока $S > P/S$
- В моменте перехода от $S > P/S$ к $S < P/S$ два соседних значения j имеет смысл попробовать как кандидаты в оптимум

Задача F - решение

— — —

Время работы $O(2^{n/2} * n)$, что легко проходит даже на Python

При этом надо с большой осторожностью относиться к переполнению, даже 64-битного типа данных недостаточно, чтобы вычислить P

Либо 128-битный тип данных (тоже с осторожностью, даже он может переполниться)

Либо длинная арифметика

Либо реализация на Python