**Муниципальный этап всероссийской олимпиады школьников по информатике 2024/2025 года в Камчатском крае.**

Оглавление

[I. Условия задач. 2](#_Toc175926927)

[Задача A. Пирамида снизу 2](#_Toc175926928)

[Задача B. Видеокамера 3](#_Toc175926929)

[Задача C. Таблица Пифагора 4](#_Toc175926930)

[Задача D. Го…ол! 5](#_Toc175926931)

[Задача E. Ковбои и кино 6](#_Toc175926932)

[II. Рекомендации по оцениванию 7](#_Toc175926933)

[III. Краткие рекомендации по решению задач, примеры решений 8](#_Toc175926934)

[IV. Набор тестов к задачам 14](#_Toc175926935)

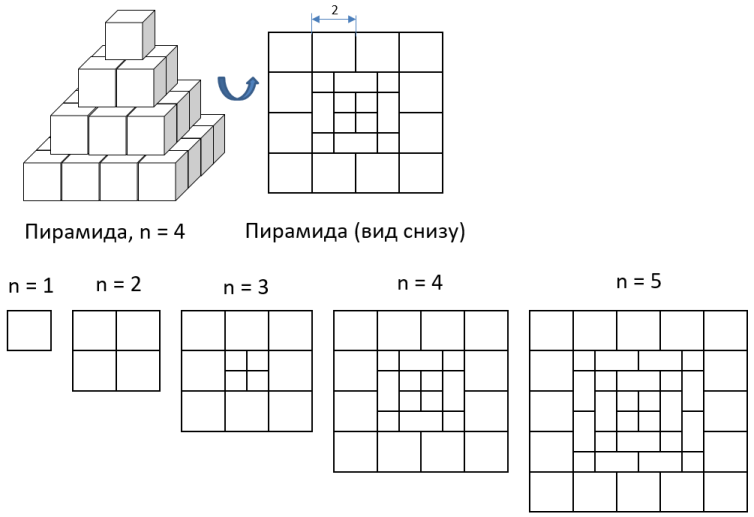
# Условия задач.

## Задача A. Пирамида снизу

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Автор: | Антон Карабанов |  | Ограничение времени: | 1 сек |
| Входной файл: | Стандартный вход |  | Ограничение памяти: | 256 Мб |
| Выходной файл: | Стандартный выход |  |  |  |
| Максимальный балл: | 100 |  |  |  |

### *Условие*

### Тимофей склеил из кубиков 2×2×2 пустую изнутри *n*-этажную пирамиду (смотри рисунок). Потом он посмотрел на неё снизу. Получившийся узор оказался настолько красивым, что мальчик незамедлительно перерисовал его в тетрадь. Определите суммарную длину всех проведённых им линий.



### *Формат входных данных*

Единственная строка входных данных содержит натуральное число *n*.Обратите внимание, что при заданных ограничениях для хранения ответа необходимо использовать 64-битный тип данных, например, long long в C++, int64 в Free Pascal, long в Java.

### *Формат выходных данных*

Выведите одно натуральное число — ответ на вопрос задачи.

### *Ограничения*

1 ≤ *n* ≤ 108

### *Система оценки и описание подзадач*

Решения, верно работающие при *n* ≤ 100, получат не менее 40 баллов.

### *Примеры тестов*

|  |  |  |
| --- | --- | --- |
| **№** | **Стандартный вход** | **Стандартный выход** |
| 1 | 4 | 92 |

## Задача B. Видеокамера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Автор: | Антон Карабанов |  | Ограничение времени: | 1 сек |
| Входной файл: | Стандартный вход |  | Ограничение памяти: | 256 Мб |
| Выходной файл: | Стандартный выход |  |  |  |
| Максимальный балл: | 100 |  |  |  |

### *Условие*

В целях недопущения, предотвращения, профилактики, упреждения и защиты, на стене кабинета информатики повесили видеокамеру.

Кабинет имеет размеры ***a*** ×***b*** метров. Камера имеет угол обзора 90°, её ось перпендикулярна стене. По вертикальной проекции определите площадь, попадающую в зону съемки.

### *Формат входных данных*

### Четыре строки входных данных содержит четыре натуральных неотрицательных целых числа ***a***, ***b***, ***x*** и ***y*** — размеры кабинета и координаты камеры (считайте, что точка начала координат совпадает с левым нижним углом кабинета, а его стороны параллельны осям). Гарантируется, что камера расположена на стене и не в углу.

### *Формат выходных данных*

### Выведите одно число — ответ на вопрос задачи. Если ответ выражается натуральным числом — выведите его, если дробным — с точностью до десятых.

### *Ограничения*

### 2 ≤ ***a*** ≤ 109

### 2 ≤ ***b*** ≤ 109

### 0 ≤ ***x*** ≤ ***a***

### 0 ≤ ***y*** ≤ ***b***

### *Система оценки и описание подзадач*

### Баллы за каждый тест начисляются независимо.

### Решения, верно работающие, когда область съёмки не ограничивается боковыми стенами кабинета, получат не менее 20 баллов.

### *Пояснение к примеру*

### Смотри рисунок.

### 

### *Примеры тестов*

|  |  |  |
| --- | --- | --- |
| **№** | **Стандартный вход** | **Стандартный выход** |
| 1 | 8  4  5  4 | 15.5 |

## Задача C. Таблица Пифагора

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Автор: | Антон Карабанов |  | Ограничение времени: | 1 сек |
| Входной файл: | Стандартный вход |  | Ограничение памяти: | 256 Мб |
| Выходной файл: | Стандартный выход |  |  |  |
| Максимальный балл: | 100 |  |  |  |

### *Условие*

### У Тимофея, как и у многих других учеников, на последней странице тетради по математике напечатана Таблица Пифагора. Обычно её печатают размером 10×10, но у Тимофея есть тетради с нестандартными размерами этой таблицы. Определите количество различных чисел внутри такой таблицы размером ***n*** ×***n***, которые оканчиваются на цифру ***d***.

### *Формат входных данных*

### Две строки входных данных содержат натуральное число ***n*** и десятичную цифру ***d***.

### *Формат выходных данных*

### Выведите одно натуральное число — ответ на вопрос задачи.

### *Ограничения*

### 1 ≤ ***n*** ≤ 109

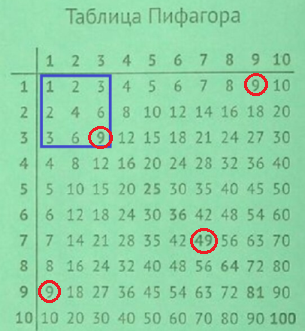
### 0 ≤ ***d*** ≤ 9

### *Система оценки и описание подзадач*

### Баллы за каждый тест начисляются независимо.

Решения, верно работающие при *n* ≤ 100, получат не менее 40 баллов.

### *Пояснение к примеру*

Смотри рисунок. В первом примере дано *n*= ***3***. В таблице Пифагора ***3*** ×***3*** шесть различных чисел: 1, 2, 3, 4, 6, 9. Ни одно из них не оканчивается на цифру ***0***.

Во втором примере дано *n*= ***10***. В таблице Пифагора ***10*** ×***10*** на цифру ***9*** оканчиваются 4 числа: 9 (встречается трижды) и 49.

### *Примеры тестов*

|  |  |  |
| --- | --- | --- |
| **№** | **Стандартный вход** | **Стандартный выход** |
| 1 | 3  0 | 0 |
| 2 | 10  9 | 4 |

## Задача D. Го…ол!

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Автор: | Антон Карабанов |  | Ограничение времени: | 1 сек |
| Входной файл: | Стандартный вход |  | Ограничение памяти: | 256 Мб |
| Выходной файл: | Стандартный выход |  |  |  |
| Максимальный балл: | 100 |  |  |  |

### *Условие*

### Тимофей — страстный футбольный болельщик. Недавно он заметил, что цифра 0 и буква О очень похожи по написанию. А поскольку его любимое футбольное событие часто сопровождается определённым эмоциональным возгласом, некоторые числа приобрели для него особый смысл.

### Будем называть натуральное число голевым, если у него:

### 1) Последняя цифра не равна первой и не равна нулю;

### 2) Все цифры, кроме самой первой и самой последней, равны нулю.

### Тимофей выписал все голевые числа на листочек в порядке возрастания. По описанию числа определите, на каком месте в списке оно находится.

### *Формат входных данных*

Три строки входных данных содержат три натуральных числа: ***a*** — значение первой цифры числа, ***n*** — количество нулей и ***b*** — значение последней цифры голевого числа. Обратите внимание, что при заданных ограничениях для хранения ответа необходимо использовать 64-битный тип данных, например, long long в C++, int64 в Free Pascal, long в Java.

### *Формат выходных данных*

Выведите одно натуральное число — ответ на вопрос задачи.

*Ограничения*

### 1 ≤ ***a, b*** ≤ 9

### ***a*** ≠ ***b***

### 1 ≤ ***n*** ≤ 1015

### *Система оценки и описание подзадач*

Баллы за каждый тест начисляются независимо.

Решения, верно работающие при ***n*** ≤ 3, получат не менее 20 баллов.

Решения, верно работающие при ***n*** ≤ 104, получат не менее 50 баллов.

### *Пояснение к примеру*

В примере дано ***a***= ***2***, ***n***= ***1*** и ***b***= ***3***. Это описание соответствует числу 203.

В списке голевых чисел:

102, 103, 104, 105, 106, 107, 108, 109, 201, 203, ...

это число десятое по счету.

### *Примеры тестов*

|  |  |  |
| --- | --- | --- |
| **№** | **Стандартный вход** | **Стандартный выход** |
| 1 | 2  1  3 | 10 |

## Задача E. Ковбои и кино

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Автор: | Антон Карабанов |  | Ограничение времени: | 1 сек |
| Входной файл: | Стандартный вход |  | Ограничение памяти: | 256 Мб |
| Выходной файл: | Стандартный выход |  |  |  |
| Максимальный балл: | 100 |  |  |  |

### *Условие*

Джонни Фёст приехал в захолустный городок на Диком Западе с целью открыть там синематограф. У него есть *a* приключенческих фильмов, *b* боевиков и *c* комедий. Местные ковбои потребовали, чтобы фильмов каждого жанра было равное количество (они хотят составить расписание и каждый вечер смотреть по одному новому фильму каждого жанра). Для этого ковбои собрали *n*долларов. Мистер Фёст может купить у приезжего торговца фильм любого жанра за 2 доллара и продать любую имеющуюся у него ленту за 1 доллар. Какое наибольшее количество вечеров ковбои смогут наслаждаться новыми фильмами? Для определённости считайте, что все торговые операции должны быть завершены до начала первого сеанса.

### *Формат входных данных*

Четыре строки входных данных содержат четыре неотрицательных целых числа: *a*, *b*, *c* и *n*. Обратите внимание, что при заданных ограничениях для хранения ответа необходимо использовать 64-битный тип данных, например, long long в C++, int64 в Free Pascal, long в Java.

### *Формат выходных данных*

Выведите одно неотрицательное целое число — ответ на вопрос задачи.

*Ограничения*

0 ≤ *a, b, c* ≤ 109

0 ≤ *n* ≤ 1015

*Система оценки и описание подзадач*

Баллы за каждый тест начисляются независимо.

Решения, верно работающие при *a, b, c* ≤ 100 и *n* ≤ 1000, получат не менее 40 баллов.

*Пояснение к примеру*

В примере у Джонни Фёста 3, 7 и 4 фильма разных жанров. Ковбои собрали для него 10 долларов.

Мистер Фёст доведёт количество фильмов каждого жанра до 6. Для этого он купит три приключенческих фильма, продаст один боевик и купит две комедии. 10 долларов для этого хватит, поскольку затраты составят 3 × 2 – 1 × 1 + 2 × 2 = 9 долларов.

Для доведения количества фильмов каждого жанра до 7 собранных средств не хватит.

### *Примеры тестов*

|  |  |  |
| --- | --- | --- |
| **№** | **Стандартный вход** | **Стандартный выход** |
| 1 | 3  7  4  10 | 6 |

# Рекомендации по оцениванию

Решением задачи является программа, написанная на одном из доступных на олимпиаде языков программирования. Для проверки и оценивания решений жюри использует автоматическую тестирующую систему. На проверку отправляется исходный текст программы. При отправке решения на проверку участник указывает, с использованием какого языка программирования и компилятора выполнено решение. Разные решения, отправленные на проверку, могут использовать разные языки программирования и/или компиляторы.

Программа запускается на тестах. Для каждого теста, на котором был выполнен запуск, устанавливается результат выполнения на этом тесте. Верный ответ на тест, выданный при соблюдении указанных в условии задачи ограничений, соответствует результату OK.

Когда программа запускается, ей указанным в условии задачи способом передаются входные данные. Для ввода данных используется стандартный поток ввода.

В условии каждой задачи приведены примеры входных и выходных данных для этой задачи. Решение участника запускается на тестах из примеров, приведенных в условии задачи, результат работы на этих тестах сообщается участнику.

Каждая задача оценивается максимум в 100 баллов. Каждый пройденный тест (за исключением тестов из условия) оценивается в 5 баллов. Оценка за задачу вычисляется по формуле: (кол-во пройденных тестов) × 5.

# Краткие рекомендации по решению задач, примеры решений

В состав пакета для каждой задачи входят решения (в электронном виде) на языках программирования Python и Си++.

Ниже представлены краткие рекомендации к решению задач.

**Задача 1. Пирамида снизу**

**Задача на вывод формулы.**

Частичное решение (***n*** <= 100):

Вручную вычислив и проанализировав результаты для первых значений ***n*** можно увидеть закономерность:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***n*** | 1 | 2 | 3 | 4 | 5 |
| ***ans*** | 8 | 24 | 52 | 92 | 144 |
| больше предыдущего значения на … |  | 16 | 28 | 40 | 52 |
| что больше предыдущего увеличения на … |  |  | 12 | 12 | 12 |

Каждый новый узор содержит столько линий, сколько есть в предыдущем, увеличенном на ***d***, которое тоже увеличивается с каждой новой итерацией на постоянное значение 12 (этот факт является следствием того, что неизвестная нам пока зависимость – квадратичная). Эта закономерность позволяет нам построить переборное решение:

n = int(input())

ans = 8

d = 16

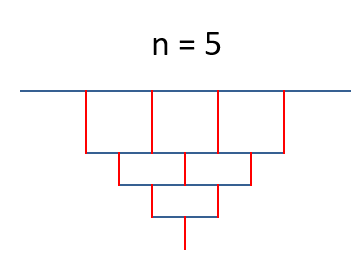
for i in range(2, n + 1):

ans += d

d += 12

print(ans)

Полное решение: Разделим диагоналями изображение на 4 равные части и рассмотрим одну из таких четвертей, например, верхнюю. Раскрасим в ней вертикальные и горизонтальные линии в разные цвета.

Вертикальные линии длины 1 (если идти от центра) образуют арифметическую прогрессию 1 + 2 + ... + (n - 2). Есть ещё вертикальные линии длины 2, их n - 1.

Всего получается (n - 2) \* (n - 1) / 2 + 2 \* n - 2.

Горизонтальные линии длины 2 (если идти от центра) образуют арифметическую прогрессию 1 + 2 + ... + (n - 2). Есть ещё горизонтальные линии длины 2 на самом краю, их n.

Всего получается (n - 2) \* (n - 1) + 2 \* n.

Сложим полученные выражения и умножим на 4. После упрощения получим формулу ans = 6 \* n \*\* 2 - 2 \* n + 4.

Пример программы (Python):

n = int(input())

print(6 \* n \*\* 2 – 2 \* n + 4)

**Задача 2. Видеокамера.**

**Задача на проверку условий.**

Частичное решение: треугольная область обзора камеры имеет площадь a \*\* 2 или b \*\* 2 (в зависимости от того, на какой из стен она висит), поскольку этот треугольник прямоугольный и равносторонний.

a = int(input())

b = int(input())

x = int(input())

y = int(input())

if y == 0 or y == b:

ans = b \*\* 2

if x == 0 or x == a:

ans = a \*\* 2

print(ans)

Полное решение: рассмотрим случай, когда камера висит на верхней или нижней стене (y = 0 или y = b).

Треугольная область обзора камеры имеет площадь S1 = b \*\* 2. Из неё могут вычитаться треугольные области, ограниченные левой и правой стенами.

Заведем переменную S2 для хранения площади вычитаемой площади. Для сохранения целочисленности будем в ней хранить площадь квадрата, а пополам поделим потом.

Проверим, не пересекает ли наш треугольник левую стенку:

if x - b < 0:

S2 += abs(x - b) \*\* 2

Проверим, не пересекает ли наш треугольник правую стенку:

if x + b > a:

S2 += (x + b - a) \*\* 2

Осталось корректно вычислить итоговую площадь:

if S2 % 2 == 1:

ans = str(S1 - 1 - S2 // 2) + '.5'

else:

ans = str(S1 - S2 // 2)

Аналогично рассматривается случай, когда камера висит на левой или правой стене.

Пример программы:

a = int(input())

b = int(input())

x = int(input())

y = int(input())

S2 = 0

if y == 0 or y == b:

S1 = b \*\* 2

if x - b < 0:

S2 += abs(x - b) \*\* 2

if x + b > a:

S2 += (x + b - a) \*\* 2

if x == 0 or x == a:

S1 = a \*\* 2

if y - a < 0:

S2 += abs(y - a) \*\* 2

if y + a > b:

S2 += (y + a - b) \*\* 2

if S2 % 2 == 1:

ans = str(S1 - 1 - S2 // 2) + '.5'

else:

ans = str(S1 - S2 // 2)

print(ans)

**Задача 3. Таблица Пифагора**

**Задача на сокращение перебора.**

Частичное решение (для небольших ***n***): полный перебор.

n = int(input())

d = int(input())

ans = 0

for i in range(1, n + 1):

for j in range(1, n + 1):

if i \* j % 10 == d:

ans += 1

print(ans)

Полное решение: ясно, что последние цифры чисел во всех таблицах размером 10 на 10 будут повторяться (поскольку их значения зависят только от последних цифр множителей, а они могут принимать значения от 0 до 9). Переберём двумя вложенными циклами все числа (a и b) от 1 до 10.

Если случилось так, что a \* b % 10 == d, то в каждом квадрате размером 10 на 10 на пересечении линий с числами, оканчивающимися на a и b будет стоять цифра d на последнем месте. Всего таких квадратов будет (n % 10) \* (n % 10) = (n % 10) \*\* 2. Добавим это число к ответу.

Возможно, что n не кратно 10. Тогда справа от последнего полного квадрата будет расположен прямоугольник высотой 10 и шириной n % 10. Если b <= n % 10, то в каждом таком прямоугольнике тоже будет пересечение линий с числами, оканчивающимися на a и b, и цифра d тоже будет стоять на последнем месте. Всего таких прямоугольников будет (n % 10). Добавим это число к ответу.

Аналогично в общем случае ниже последнего полного квадрата будет расположен прямоугольник высотой n % 10 и шириной 10. Если a <= n % 10, то в каждом таком прямоугольнике тоже будет пересечение линий с числами, оканчивающимися на a и b, и цифра d тоже будет стоять на последнем месте. Всего таких прямоугольников будет (n % 10). Добавим это число к ответу.

Наконец, в правом нижнем углу будет квадрат размером n % 10 на n % 10. Если a <= n % 10 и b <= n % 10, то в этом квадрате тоже будет пересечение линий с числами, оканчивающимися на a и b, и цифра d тоже будет стоять на последнем месте. Добавим к ответу 1.

Пример программы (Python):

n = int(input())

d = int(input())

ans = 0

for a in range(1, 11):

for b in range(1, 11):

if a \* b % 10 == d:

ans += (n // 10) \*\* 2

if b <= n % 10:

ans += n // 10

if a <= n % 10:

ans += n // 10

if a <= n % 10 and b <= n % 10:

ans += 1

print(ans)

**Задача 4. Го...ол!**

**Задача на сокращение перебора.**

Подзадача 1: будем перебирать все числа от 102 до данного числа a \* 10 \*\* (n + 1) + b и проверять их на указанные признаки, запустив счетчик подходящих чисел.

Пример программы (Python):

def is\_goal(number):

s = str(number)

if s[0] != s[-1] and s[-1] != '0' and s[1:-1] == '0' \* (len(s) - 2):

return True

return False

a = int(input())

n = int(input())

b = int(input())

ans = 0

for i in range(102, a \* 10 \*\* (n + 1) + b + 1):

if is\_goal(i):

ans += 1

print(ans)

Подзадача 2: будем формировать все подходящие числа, не превышающие данного числа, запустив счетчик чисел.

Пример программы (Python):

a = int(input())

n = int(input())

b = int(input())

number = a \* 10 \*\* (n + 1) + b

ans = 0

for x in range(1, n + 1):

for y in range(1, 10):

for z in range(1, 10):

if y != z:

p = y \* 10 \*\* (x + 1) + z

if p <= number:

ans += 1

print(ans)

Полное решение: выведем формулу.

Каждое увеличение n на 1 даёт 72 новых голевых числа (на первое и последнее место можно поставить любую цифру от 1 до 9 (всего 81 комбинация), но нужно убрать все случаи, когда эти цифры равны (таких "плохих" комбинаций 9).

Каждое увеличение a на 1 (при том же n) даёт 8 новых голевых чисел (на последнее место можно поставить любую цифру от 1 до 9 (всего 9 комбинаций), но нужно убрать один случай, когда эта цифра равна первой.

Каждое увеличение b на 1 (при тех же же a и n) даёт одно новое голевое число. Однако, если b > a, то одно число не будет голевым (если b = a).

Окончательно, формула выглядит так:

1) если b < a:

ans = 72 \* (n - 1) + 8 \* (a - 1) + b

2) если b > a:

ans = 72 \* (n - 1) + 8 \* (a - 1) + b – 1

Пример программы (Python):

a = int(input())

n = int(input())

b = int(input())

ans = (n - 1) \* 72 + (a - 1) \* 8 + (b - 1)

if b < a:

ans += 1

print(ans)

**Задача 5. Ковбои и кино.**

**Бинарный поиск.**

Частичное решение: перебор всех возможных ответов, начиная с наименьшего из чисел a, b, c. Для каждого ответа определяем, хватит ли нам собранных средств. Как только средств не хватит, выходим из цикла.

Пример программы (Python):

a = int(input())

b = int(input())

c = int(input())

n = int(input())

ans = min(a, b, c)

summa = 3 \* ans - (a + b + c)

while summa <= n:

ans += 1

summa = 0

if a > ans:

summa -= a - ans

else:

summa += 2 \* (ans - a)

if b > ans:

summa -= b - ans

else:

summa += 2 \* (ans - b)

if c > ans:

summa -= c - ans

else:

summa += 2 \* (ans - c)

print(ans - 1)

Полное решение. Бинарный поиск по ответу.

Чем больше количество вечеров, тем больше долларов для просмотра фильмов потребуется.

Для каждого количества вечеров мы можем быстро узнать количество необходимых денег, значит, бинарный поиск применим. Реализуем его.

a = int(input())

b = int(input())

c = int(input())

n = int(input())

left = 0

right = 10 \*\* 18

while right - left > 1:

middle = (right + left) // 2

summa = 0

if a < middle:

summa += 2 \* (middle - a)

else:

summa -= a - middle

if b < middle:

summa += 2 \* (middle - b)

else:

summa -= b - middle

if c < middle:

summa += 2 \* (middle - c)

else:

summa -= c - middle

if summa <= n:

left = middle

else:

right = middle

print(left)

# Набор тестов к задачам

В состав пакета для каждой задачи входят тесты из условия и 20 уникальных тестов, на которых рекомендуется оценивать решение участников олимпиады, а также верные ответы на эти тесты.

В тестирующей системе также проверяются решения участников олимпиады на тестах из условия задачи. Баллы за такие тесты не даются.