Решение:

В данной задаче нужно было сделать граф ациклическим путем разворота некоторых ребер. Если граф ациклический, то у него есть топологическая сортировка. Значит, если бы позволяли ограничения, можно было бы перебрать все перестановки вершин графа в качестве топологических сортировок и найти ту, которая получается минимальным числом разворотов ребер. Для того, чтобы узнать для фиксированной топологической сортировки, сколько ребер нужно перевернуть, можно пройти двумя циклами по всем парам вершин и если первая вершина идет в топологической сортировке раньше, чем вторая, то нам необходимо развернуть все ребра из второй вершины в первую, чтобы топологическая сортировка была корректна. Соответственно, сумма таких ребер «справа налево» будет ответом для данной топологической сортировки. Таким образом, минимальным числом ребер, которые нужно развернуть, будет минимум из полученных ответов для всех топологических сортировок.

К сожалению, данное решение работает за $O(n! \cdot n^2)$, что не укладывается ограничения этой задачи, поэтому нужно как-то соптимизировать данное решение. Для оптимизации используем метод динамического программирования по подмножествам.

• dp[mask] — минимальное число ребер, которые нужно развернуть, чтобы граф состоящий из вершин в маске mask стал ациклическим

• В пустом графе ничего разворачивать не нужно, значит dp[0] = 0

• $dp[mask] = \min(dp[mask - 2^i] + cnt(i, mask - 2^i))$ для всех i, которые есть в маске. Здесь $cnt(i, mask - 2^i)$ — число ребер, которые направлены из вершины i в вершины, которые есть в маске $mask - 2^i$. Суть данного перехода в том, что мы уже знаем ответ для масок $mask - 2^i$, значит для них существует некоторая топологическая сортировка, к этой топологической сортировке, мы как-будто дописываем в конец вершину i. Данное решение будет работать за $O(2^n \cdot n^2)$, что в свою очередь уложится в необходимые ограничения.

Заключительный этап 9 и 10 класса (приведен один из вариантов заданий)

1. Кодирование информации. Системы счисления (2 балла)

[Уравнение с тремя неизвестными]

Дано равенство:

$$2334_X + 14X7_Y = 12Y4_Z$$

X, Y и Z являются целыми положительными числами, которые равны значениям отдельных цифр чисел в равенстве или образуют основания систем счисления, в которых записаны эти числа в равенстве. Найдите такие значения X, Y и Z, что для них равенство будет выполняться, и запишите в ответ через пробел сначала значение X в десятичной системе счисления, затем значение Y в десятичной системе счисления и затем значение Z в десятичной системе счисления. Если таких комбинаций несколько, запишите ту, в которой значение X минимально. Если таких комбинаций не существует, запишите в ответ NULL.

Ответ: 6 9 11

Решение:

Перепишем равенство, используя формулу развернутой записи числа:

$2*X^3 + 3*X^2 + 3*X + 4 + Y^3 + 4*Y^2 + X*Y + 7 = Z^3 + 2*Z^2 + Y*Z + 4$

Немного упростив, получим:

$2*X^3 + 3*X^2 + 3*X + Y^3 + 4*Y^2 + X*Y + 7 = Z^3 + 2*Z^2 + Y*Z$

Определим ограничения исходя из используемых в записях чисел цифр: X>4, Y>7, Y>X, Z>Y.

Аналитическое решение будет, возможно, слишком громоздким, поэтому подходящие значения удобно подобрать программно, например, так:

```
f=0
i=50809
while f==0:
    answer=[i//(100**2),(i//100)%100,i%100]
    A1=[2,3,3,4]
    A2=[1,4,answer[0],7]
    A3=[1,2,answer[1],4]
    R1=A1[0]*answer[0]**3+A1[1]*answer[0]**2+A1[2]*answer[0]+A1[3]
    R2=A2[0]*answer[1]**3+A2[1]*answer[1]**2+A2[2]*answer[1]+A2[3]
    R3=A3[0]*answer[2]**3+A3[1]*answer[2]**2+A3[2]*answer[2]+A3[3]
    if (R1+R2)==R3 and answer[0]>max(A1) and answer[1]>max(A2) and answer[2]>max(A3):
        f=1
        print(answer)
    i+=1
```

Результатом выполнения программы будет [6, 9, 11]. Подставив значения в исходное равенство, легко убедиться в правильности решения.

Другой вариант программного решения:

```
from itertools import product

digits = [x for x in range(5, 37)]
for x, y, z in product(digits, repeat=3):
    if 7 < y < z and x < y and \
            2 * x ** 3 + 3 * x ** 2 + 3 * x + 4 + y ** 3 + 4 * y ** 2 + x * y + 7 == z ** 3 + 2 * z ** 2 + y * z
+ 4:
        print(x, y, z)
        break
```

2. Кодирование информации. Объем информации (2 балла) [100 картинок]

У Пети есть 100 квадратных растровых изображений со стороной в N пикселей каждое. Петя хранит каждое изображение в виде последовательности кодов оттенков пикселей, используя стандартную цветовую модель TrueColor, то есть, затрачивая для хранения отдельного кода 24 бита. Вася решил помочь Пете уменьшить хранимый объем данных. Он обратил внимание, что все изображения можно разбить на три группы. В первую группу попала ровно половина исходных изображений. В этой группе в каждом изображении встречается только 65536 различных оттенков. Во второй группе ровно четверть исходных изображений. В этой группе в каждом изображении встречается только 16384 различных оттенков. В третью группу попала оставшаяся четверть исходных изображений и в ней в каждом изображении встречается только 1024 различных оттенка. Вася решил хранить изображения следующим образом. Сначала в каждом изображении он хранит его палитру – последовательность из 24-хбитных кодов TrueColor такой длины, сколько различных оттенков встречается в соответствующем изображении. Затем он хранит коды для каждого пикселя, определяющие номер оттенка в хранимой палитре так, что для каждого кода используется минимальное, одинаковое для всех кодов в палитре данного изображения количество бит. Вася выяснил, что всего он сэкономил 117125 КБайт данных на всем наборе изображений Пети. Определите N, при котором это возможно.

В ответе укажите целое число.

Примечание. 1 КБайт = 1024 байт.

Ответ: 1024

Решение:

Составим уравнение по условию:

$100*N^2*24 – 50*(2^{16}*24 + N^2*\log_2 65536) – 25*(2^{14}*24 + N^2*\log_2 16384) – 25*(2^{10}*24 + N^2*\log_2 1024) = 117125*2^{10}*8$

Осуществим алгебраические преобразования и вычисления:

$2400*N^2 – 800*N^2 – 350*N^2 – 250*N^2 = 117125*2^{13} + 9600*2^{13} + 1200*2^{13} + 75*2^{13}$

$1000*N^2 = 128000*2^{13}$

$N^2 = 2^{20}$

$N = 2^{10} = 1024$

Это и будет правильным ответом.

3. Основы логики (3 балла) [Исключающие ИЛИ]

Дано логическое равенство:

(A xor B) → (A xor F(A,B)) = F(A,B) → (B xor A)

Найдите логическую функцию F(A,B), такую, что указанное равенство будет выполняться.

Если таких функций несколько – запишите любую из них.

В ответе запишите формулу, которая может содержать логические переменные A и B и не более чем три логические операции. Если таких функций не существует, запишите в ответ NULL.

Комментарий по вводу ответа: операнды вводятся большими латинскими буквами; логические операции обозначаются, соответственно как not, and и or. Запись не должна содержать скобок. Пример записи ответа: A or not B

Ответ: not A and B || B and not A

Решение:

Поскольку F(A,B) является функцией от двух логических аргументов, построим её таблицу истинности. Для этого последовательно подставим в логическое равенство четыре возможные комбинации значений A и B.

При A=0 и B=0 получим:

(0 xor 0) → (0 xor F(A,B)) = F(A,B) → (0 xor 0)

Вычислим скобки, в которых только константы:

0 → (0 xor F(A,B)) = F(A,B) → 0

Упростим левую и правую часть, раскрыв импликации:

1 = not F(A,B)

Следовательно, при A=0 и B=0, F(A,B)=0

При A=0 и B=1 получим:

(0 xor 1) → (0 xor F(A,B)) = F(A,B) → (1 xor 0)

Вычислим скобки, в которых только константы:

1 → (0 xor F(A,B)) = F(A,B) → 1

Упростим левую и правую часть, раскрыв импликации:

0 xor F(A,B) = 1

F(A,B) = 1

Следовательно, при A=0 и B=1, F(A,B)=1

При A=1 и B=0 получим:

(1 xor 0) → (1 xor F(A,B)) = F(A,B) → (0 xor 1)

Вычислим скобки, в которых только константы:
1 → (1 xor F(A,B)) = F(A,B) → 1
Упростим левую и правую часть, раскрыв импликации:
1 xor F(A,B) = 1
not F(A,B) = 1
Следовательно, при A=1 и B=0, F(A,B)=0

При A=1 и B=1 получим:
(1 xor 1) → (1 xor F(A,B)) = F(A,B) → (1 xor 1)
Вычислим скобки, в которых только константы:
0 → (1 xor F(A,B)) = F(A,B) → 0
Упростим левую и правую часть, раскрыв импликации:
1 = not F(A,B)
Следовательно, при A=1 и B=1, F(A,B)=0

Таким образом, мы получили значения F(A,B) для всех возможных комбинаций значений аргументов и можем построить таблицу истинности:

A	B	F(A,B)
0	0	0
0	1	1
1	0	0
1	1	0

Эта таблица истинности содержит только одну строку с истинным значением, следовательно искомая логическая функция является конъюнкцией. Поскольку истинное значение она принимает при A=0 и B=1, это функция **not A and B**.

Другой вариант решения

Обозначим операцию **xor** как $\oplus$, а искомую функцию $\boldsymbol{F(A,B) = X}$. Тогда наше уравнение будет иметь вид:

$$(\boldsymbol{A \oplus B}) \rightarrow (\boldsymbol{A \oplus X}) = \boldsymbol{X} \rightarrow (\boldsymbol{A \oplus B})$$

По сути, мы имеем левую и правую части нашего равенства, которые представляют собой две логические функции от трех переменных ($\boldsymbol{A, B, X}$). Построим таблицы истинности этих функций, чтобы найти наборы переменных, на которых эти функции равны между собой.

$\boldsymbol{A}$	$\boldsymbol{B}$	$\boldsymbol{X}$	$(\boldsymbol{A \oplus B})$	$(\boldsymbol{A \oplus X})$	$(\boldsymbol{A \oplus B}) \rightarrow (\boldsymbol{A \oplus X})$	$\boldsymbol{X} \rightarrow (\boldsymbol{A \oplus B})$
0	0	0	0	0	1	1
0	0	1	0	1	1	0
0	1	0	1	0	0	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	1	0	0	1
1	1	0	0	1	1	1
1	1	1	0	0	1	0

Выберем те строки в этой таблице, в которых значения левой и правой части равенства совпадают.

$\boldsymbol{A}$	$\boldsymbol{B}$	$\boldsymbol{X}$	$(\boldsymbol{A \oplus B})$	$(\boldsymbol{A \oplus X})$	$(\boldsymbol{A \oplus B}) \rightarrow (\boldsymbol{A \oplus X})$	$\boldsymbol{X} \rightarrow (\boldsymbol{A \oplus B})$
0	0	0	0	0	1	1
0	0	1	0	1	1	0
0	1	0	1	0	0	1
0	1	1	1	1	1	1
1	0	0	1	1	1	1
1	0	1	1	0	0	1
1	1	0	0	1	1	1
1	1	1	0	0	1	0

Выпишем эти строки, вспомнив, что $\boldsymbol{F(A,B) = X}$

$\boldsymbol{A}$	$\boldsymbol{B}$	$\boldsymbol{F(A,B) = X}$
0	0	0
0	1	1
1	0	0
1	1	0

Очевидно, что $\boldsymbol{F(A,B) = \neg A \wedge B}$

4. Алгоритмизация и программирование. Формальные исполнители (2 балла) [Сходящаяся последовательность]

Дан алгоритм обработки числовой последовательности, на вход которому подали возрастающую последовательность натуральных чисел от 1 до N:

1. Если первый элемент последовательности – нечетное число, то из последнего элемента последовательности вычесть значение первого элемента последовательности, в противном случае, к последнему элементу последовательности прибавить значение первого элемента последовательности.
2. Удалить первый элемент последовательности.
3. Если в последовательности осталось более одного элемента, перейти на шаг 1, иначе завершить работу алгоритма и вывести получившуюся последовательность.

Например, для N=5 получится следующая цепочка преобразований:
[1,2,3,4,5] → [2,3,4,4] → [3,4,6] → [4,3] → [7] (последнее значение будет выведено как результат работы алгоритма).
Легко заметить, что результатом работы алгоритма всегда будет последовательность, состоящая из одного элемента.
Найдите все такие N, при которых в результате работы алгоритма этот единственный элемент последовательности будет иметь значение 67. В ответе укажите все подходящие значения N через пробел в порядке возрастания. Если таких значений больше 5, укажите первые пять из них. Если таких значений не существует, укажите в ответ NULL.

Ответ: 45 134

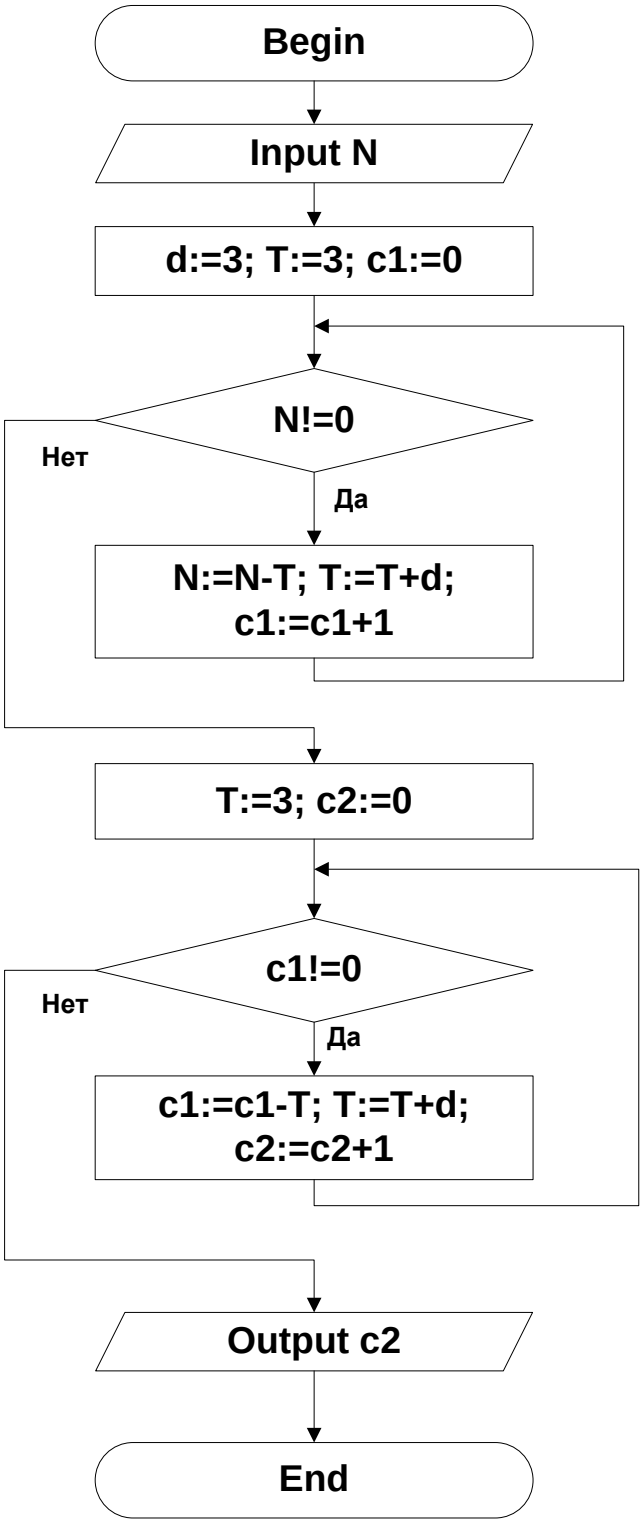
Решение:

Рассмотрим примеры построения последовательностей для нескольких первых значений N, начиная с 3:
[1,2,3] → [2,2] → [4]
[1,2,3,4] → [2,3,3] → [3,5] → [2]
[1,2,3,4,5] → [2,3,4,4] → [3,4,6] → [4,3] → [7]
[1,2,3,4,5,6] → [2,3,4,5,5] → [3,4,5,7] → [4,5,4] → [5,8] → [3]
[1,2,3,4,5,6,7] → [2,3,4,5,6,6] → [3,4,5,6,8] → [4,5,6,5] → [5,6,9] → [6,4] → [10]
[1,2,3,4,5,6,7,8] → [2,3,4,5,6,7,7] → [3,4,5,6,7,9] → [4,5,6,7,6] → [5,6,7,10] → [6,7,5] → [7,11] → [4]
Заметим закономерность. Если N – четное число, результирующий элемент последовательности будет равен N/2. Если же N – нечетное число, то к значению, полученному для предыдущего нечетного N, прибавляется 3, то есть это значение может быть выражено как (3*N-1)/2.
Таким образом, может быть максимум два значения N – четное и нечетное.
Для четного значения N может быть получено умножением данного в условии значения последнего элемента на два, то есть 67*2=134. Для нечетного значения N можно рассчитать по формуле (67*2+1)/3=45. Остается записать полученные значения в порядке возрастания и получить ответ «45 134»

5. Алгоритмизация и программирование. Анализ алгоритма, заданного в виде блок-схемы (1 балл)

[До нуля]

Дана блок-схема алгоритма:



Какое целое положительное число нужно подать на вход, чтобы на выходе получилось значение число 18? В ответе укажите целое число.

Примечание. Оператор «!=» означает «не равно».

Ответ: 395523

Решение:

Рассмотрим второй цикл. В этом цикле переменная c1 уменьшается на каждом шаге на значение, равное переменной T, которая, в свою очередь, исходно равна 3 и затем увеличивается на 3 на каждом шаге. Выход из цикла возможен, только если переменная c1 станет строго равна нулю. Обратим внимание на переменную c2. Она представляет собой счетчик шагов этого цикла. Следовательно, успешный выход из цикла за c2 шагов означает, что значение c1 перед началом цикла было суммой арифметической прогрессии с первым элементом, равным 3, с шагом 3 и длиной в 18 элементов. Воспользуемся формулой для вычисления суммы первых n членов арифметической прогрессии c1=(2*3+3*(18-1))/2*18=513.

Заметим, что первый цикл устроен точно также. Только уменьшается до нуля введенная на входе переменная N, а начальное значение T и шаг остаются такими же. При этом c1 является счетчиком количества шагов в первом цикле, то есть определяет количество элементов в другой арифметической прогрессии, сумме членов которой и должно равняться N. Тогда можно вычислить N по формуле: N=(2*3+3*(513-1))/2*513=395523. Это и будет ответом на задание.

6. Телекоммуникационные технологии (2 балла). [Приоритезация трафика]

Два приложения – A и B – передают данные по сети используя один неразделяемый канал передачи данных со скоростью 4096 бит в секунду. Данные передаются пакетами. Приложение A передает данные пакетами, размером в 2 КБайт каждый, а приложение B передает данные пакетами, размером, в 6 КБайт каждый. У каждого приложения есть буфер. Приложение A создает и помещает в свой буфер пакет в начальный момент времени и далее по истечении каждой 5-ой секунды. Приложение B создает и помещает в свой буфер пакет по истечении 11-ой секунды от начального момента времени и далее по истечении каждой 20-ой секунды от момента создания первого пакета. Например, если считать начальным моментом времени секунду с номером 0, то приложение A будет помещать пакеты в свой буфер в начале секунд с номерами: 0, 5, 10, 15, 20, …, а приложение B в начале секунд с номерами: 11, 31, 51, 71, …

Передача любого пакета происходит целиком, и не может быть прервана. Как только передача пакета завершена, производится проверка наличия готовых к передаче пакетов в буферах приложений. Если хотя бы в одном буфере есть готовый к передаче пакет, в том числе появившийся в буфере в этот момент времени, незамедлительно начинается передача очередного пакета. У приложения B есть приоритет передачи данных по отношению к приложению A. Это означает, что, если есть готовые к передаче пакеты в его буфере, его пакет будет выбран для передачи. Пакет из буфера приложения A будет выбран для передачи только, если в момент выбора нет готовых к передаче пакетов в буфере приложения B. Пакет удаляется из буфера, как только закончена его передача. Какое количество пакетов будет в буфере приложения A в момент окончания передачи 31-го пакета приложения B? В ответе укажите целое число. Если в указанный момент времени в буфере приложения A не будет пакетов, укажите в ответ 0.

Ответ: 63

Решение:

Построим диаграмму, раскрывающую первые 25 секунд описанного процесса:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Появление пакета A	1					2					3					4					5					6
Появление пакета B												1														
Передача пакета A	1					2					3															
Передача пакета B																1										
Пакетов в буфере A	1				0	1				0	1				0	1					2					3
Пакетов в буфере B												1														

В первой строке указаны секунды. Во второй и третьей строках – номера появляющихся пакетов у приложений A и B соответственно. В 5 и 6 строках отмечены периоды передачи пакетов и номера передаваемых пакетов. В 8 и 9 строке указано количество пакетов в буферах приложений в те секунды, когда эти значения изменяются.

Заметим, что начиная с 14 секунды передача данных всегда будет вестись непрерывно, поскольку у приложения A непустой буфер и он не будет успевать его опустошать до появления нового пакета в буфере приложения B с учетом появления новых пакетов в буфере приложения A. Проанализируем следующие 40 секунд:

26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
				7					8					9					10					11					12					13					14
					2																				3													
4					5															6				7														
								2																										3					
				3				2	3					4					5					5				4	5					6				7
0					1															0				1													

Заметим, что приложение A успевает передать два пакета, после чего начинается передача более приоритетного пакета приложения B. Причем, интервалы образуют период: очередной пакет приложения B появляется на второй секунде передачи второго из двух передаваемых пакетов приложения A. По окончании передачи второго пакета приложения A начинается передача пакета приложения B. За время его передачи в буфере приложения A появляется три новых пакета, плюс еще один пакет в буфере приложения A появляется пока оно передает свои два пакета паузу до появления очередного пакета приложения B. Затем последовательность из передачи двух пакетов приложения A и одного пакета приложения B повторяется с такими же моментами появления очередных пакетов.

Легко сделать вывод, что к моменту окончания передачи очередного пакета приложения B, в буфере приложения A всегда оказывается на два пакета больше, чем в момент окончания передачи предыдущего пакета приложения B. Исходя из того, что к моменту окончания передачи первого пакета приложения B, в буфере приложения A находилось 3 пакета, полученную закономерность можно описать формулой R=N*2+1, где N – номер пакета приложения B, а R – искомое количество пакетов приложения A, находящихся в его буфере в момент окончания передачи этого пакета приложения B. Таким образом, в момент окончания передачи 31-го пакета приложения B в буфере приложения A будет 31*2+1=63 пакета, что и будет правильным ответом.

Программное решение

```
ah = 5   # Диапазон появления пакетов A
bh = 20   # Диапазон появления пакетов B
```

```
ta = 2 * 2 ** 13 // 4096  # Время передачи пакета A
tb = 6 * 2 ** 13 // 4096  # Время передачи пакета B
a = 0  # Количество пакетов в буфере A
b = 0  # Количество пакетов в буфере B
ib = 0  # Счетчик пакетов B
t = 0  # Счетчик времени
work = 0  # Обратный отсчет времени передачи очередного пакета
flag = None  # Сигнал, какой пакет передается в данный момент
while True:
    if t % ah == 0:  # Появляется новый пакет A
        a += 1
    if (t - 11) % bh == 0:  # Появляется новый пакет B
        b += 1
    if b != 0 and work == 0:  # Если буфер B не пуст и закончена передача очередного пакета
        work = tb  # начинаем передавать пакет B
        flag = 'B'
        ib += 1
    elif a != 0 and work == 0: # Если буфер B пуст, а буфер A не пуст и закончена передача очередного пакета
        work = ta  # начинаем передавать пакет A
        flag = 'A'
    if work > 0:
        work -= 1
        if work == 0:  # Если окончена передача пакета, выясняем какой пакет передавался и удаляем его из буфера
            if flag == 'A':
                a -= 1
            else:
                b -= 1
                if ib == 31:  # Если закончен передаваться 31-й пакет B, то выводим содержимое буфера A и
останавливаемся
                    print(a)
                    break
    t += 1
```

7. Технологии обработки информации в электронных таблицах, технологии сортировки и фильтрации данных (2 балла)

[Три столбца]

Дан фрагмент электронной таблицы в режиме отображения формул:

	A	B	C	D
1				
2	0	=ОСТАТ(ЧАСТНОЕ(A$1;СТЕПЕНЬ(B$1;A2));B$1)	=B2	
3	1		=ЕСЛИ(B2<B3;C2+B3;B3)	
4	2			
5	3			
6	4			
7	5			
8				
9			=СУММ(C2:C7)	
10				

Ячейку B2 скопировали во все ячейки диапазона B3:B7. Ячейку C3 скопировали во все ячейки диапазона C4:C7. В ячейку A1 поместили некоторое целое положительное число, а в ячейку B1 поместили число 8.

Какое максимальное число может быть получено в ячейке C9? Какое минимальное число нужно поместить в ячейку A1 для того, чтобы получить такое значение в ячейке C9? В ответе укажите через пробел два числа: сначала ответ на первый вопрос и затем ответ на второй вопрос.

Ответ: 77 256794

Решение:

Обратим внимание, что формула, записанная в ячейке B2 и скопированная в ячейки диапазона B3:B7 позволяет получить в ячейках диапазона B2:B7 младшие 6 разрядов восьмеричной записи числа, помещенного в ячейку A1. Причем разряды сверху вниз упорядочены от младшего к старшему. Формула, помещенная в ячейку C2 и скопированная в ячейки диапазона C3:C7 считает суммы ячеек диапазона B2:B7, которые расположены по возрастанию.

Соответственно, нам нужно получить максимальную сумму таких сумм, построенных по разрядам числа в восьмеричной системе счисления.

Легко заметить, что максимальную сумму мы получим, если будем максимально долго накапливать сумму в ячейках диапазона C2:C7, то есть если в ячейках диапазона B2:B7 буду подряд расположенные по возрастанию цифры вплоть до максимально возможной в восьмеричной системе счисления цифры 7. Тогда сумма ячеек диапазона C2:C7 составит 2+(2+3)+(2+3+4)+(2+3+4+5)+(2+3+4+5+6)+(2+3+4+5+6+7)=77. Следовательно, минимальное число, которое нужно поместить

в ячейку A1, чтобы получить такую сумму, это десятичное число, соответствующее восьмеричному числу 765432_8. Переведем его в десятичную систему счисления и получим 256794. Запишем ответы в требуемом порядке и получим «77 256794»

8. Технологии программирования (2 балла) [Финальная стоимость]

Имя входного файла	стандартный ввод
Имя выходного файла	стандартный вывод
Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

Представьте, что вас наняли в IT отдел сетевого продуктового магазина. Со следующего месяца планируется ввести скидку в 20% на некоторые товары. От вас требуется написать программу расчета финальной стоимости покупки с учетом всех скидок.

Покупка задается в виде списка товаров, где каждый товар описывается в формате <НазваниеТовара> <цена> x<количество>. Если название товара заканчивается на подстроку «sale», то цена на товар при финальном расчете покупки уменьшается на 20% с округлением до целого числа в меньшую сторону. Так, например, если цена товара была 12 рублей, с учетом скидки она составит 9 рублей. На остальные товары скидка не распространяется.

Суммарная стоимость каждого товара вычисляется по формуле finalPrice·x, где finalPrice - финальная цена, возможно, с учетом скидки и округлением, а x - количество товара в списке покупок.

Ваша задача - вычислить итоговую стоимость покупки.

Формат входных данных

В первой строке дано число n - число наименований товаров в покупке ($1 \le n \le 1000$).

В следующих n строках заданы описания товаров в покупке в формате: <НазваниеТовара> <цена> x<количество>. Название товара состоит из не более чем 20 строчных латинских букв. Цена - натуральное число, которое строго больше 1, но не больше 1000. Количество - натуральное число не больше 100.

Формат выходных данных

Выведите одно число - итоговую стоимость покупки.

Пример

Стандартный ввод	Стандартный вывод
5 bananassale 10 x2 breadfixed 5 x5 colafixed 20 x2 gumsale 9 x10 tea 2 x15	181

Замечание

В тесте из примера итоговые стоимости товаров составят: 8, 5, 20, 7 и 2 соответственно. Итоговая стоимость покупки: $8 \cdot 2 + 5 \cdot 5 + 20 \cdot 2 + 7 \cdot 10 + 2 \cdot 15 = 181$.

Решение:

В данной задаче нужно было аккуратно написать ровно то, что просилось в условии. Распарсить входные данные после чего либо проверить, что последние четыре буквы названия товара составляют подстроку «sale», либо что последние пять букв названия товара составляют подстроку «fixed». Далее необходимо прибавить к ответу либо price · x, либо (price · 4 ÷ 5) · x в зависимости от условия задачи. Здесь price — цена товара в списке покупок, x — количество товара, а ÷ — операция целочисленного деления. Также корректно домножать price на 0.8, после чего приводить результат к целочисленному типу данных.

9. Технологии программирования (4 балла) [Напитки и стаканы]

Имя входного файла	стандартный ввод
Имя выходного файла	стандартный вывод
Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

К Коле на день рождения придут n друзей. По этому случаю он заготовил n бутылок различных напитков, а каждый из друзей принесет свой стакан.

Коля знает размеры стаканов каждого друга, однако, может случиться так, что содержимое конкретной бутылки может не поместиться полностью в конкретный стакан, если объем бутылки больше объема стакана. При этом Коля не хочет, чтобы после угощения что-то осталось, поэтому будет разливать напитки только так, чтобы они полностью поместились в стаканы. Напиток полностью помещается в стакан, если объем, содержащей его бутылки, не превосходит объем стакана.

Коля хочет своеобразно оценить сколько способов есть разлить напитки по стаканам друзей так, чтобы все напитки полностью поместились в стаканы. Так как число способов может быть довольно большим, Коля хочет знать его по модулю 1 000 000 007. Помогите Коле посчитать это число.

Обратите внимание на ограничения, если написать неэффективное решение, например перебор, вы получите превышение ограничения по времени!

Формат входных данных

В первое строке дано число t - число тестовых наборов ($1 \le t \le 100$).

Каждый тестовый набор задается тремя строками. В первой из них дано число n_i - число бутылок напитков и стаканов в i-м наборе ($1 \le n_i \le 10^5$).

Во второй строке заданы n_i чисел $a_{i,j}$ - объемы бутылок в i-м наборе.

В третьей строке даны n_i чисел $b_{i,j}$ - объемы стаканов в i-м наборе ($1 \le a_{i,j}, b_{i,j} \le 100$).

Гарантируется, что сумма n_i по всем тестовым наборам не превосходит $3 \cdot 10^5$.

Формат выходных данных

Для каждого тестового набора выведите одно число - число способов разлить напитки по стаканам друзей так, чтобы все напитки полностью поместились в стаканы, взятое по модулю 1 000 000 007.

Пример

Стандартный ввод	Стандартный вывод
3 3 1 1 1 1 2 1 5 1 2 3 4 5 1 2 3 4 5 3 2 2 2 2 1 2	6 1 0

Замечание

В первом тестовом наборе любой напиток помещается в любой стакан. Во втором тестовом наборе, существует лишь один способ разлить напитки. В третьем наборе ни один напиток не поместится во второй стакан, поэтому ответ будет ноль.

Решение:

Для решения данной задачи выведем комбинаторную формулу. Начнем с самого маленького стакана. Сколько различных напитков можно в него налить, не нарушая условие задачи? Ровно столько, сколько существует бутылок напитков с объемом не больше объема этого стакана. Сколько различных напитков можно налить во второй по величине стакан? Столько, сколько существует бутылок напитков с объемом не больше объема этого стакана минус один, потому что один напиток мы уже налили в первый стакан. Сколько различных...

По такой логике получаем искомую формулу $ans = cnt_1 \cdot (cnt_2 - 1) \cdot (cnt_3 - 2)...$ Здесь cnt_i число бутылок напитков, объем которых меньше, чем объем i-го по размеру стакана. К сожалению, если считать эту величину «в лоб», попарно сравнивая объемы бутылок и стаканов, можно получить ошибку «Превышен лимит времени исполнения». Посчитать эту величину быстрее можно, например, методом двух указателей, после сортировки обоих массивов. Таким образом получим решение, которое работает за O(n log n + n).

Заключительный этап 7 и 8 класса (приведен один из вариантов заданий)

1. Системы счисления. 1 балл. [Система неравенств]

Известно, что некоторое целое положительное число f может быть получено в результате вычисления следующего выражения:

$f = 2000_N - 100_N$, где N – неизвестное основание позиционной системы счисления.

Дана система неравенств:

$$\begin{cases} 258_{16} \le f < 7D0_{16} \\ 3E8_{16} \ge f > C8_{16} \end{cases}$$

Определите, для каких N эта система неравенств будет верна.

В ответе запишите через пробел все подходящие основания систем счисления N в порядке возрастания.

Ответ: 7 8

Решение:

Сначала объединим условия системы неравенств:

1. $7D0_{16} > 3E8_{16}$, следовательно $f <= 3E8_{16}$
2. $C8_{16} < 258_{16}$, следовательно $258_{16} <= f$