Задача A. Яичница

Для решения этой задачи требуется вывести $X - Y$.

Однако в этой задаче есть две вспомогательные группы тестов, позволяющие решать ее поэтапно.

Для прохождения 1-й группы тестов можно не считывать вводимые числа вовсе. Достаточно написать вывод числа 7 — ответа на задачу при $X = 7, Y = 0$.

Во 2-й группе тестов $Y = 0$, поэтому ответ равен $X - 0 = X$. Таким образом, для прохождения второй группы тестов достаточно считать только число X и вывести его.

Критерии оценивания:

№	Баллы	Ограничения		Необх. группы
		X	Y	
1	17	$X = 7$	$Y = 0$	—
2	34	$X \leqslant 50$	$Y = 0$	1
3	49	$X \leqslant 50$	$Y < X$	1, 2

Баллы выставляются автоматически проверяющей системой.

Задача B. Сумма двух

Решение задачи предполагает разбор всех случаев. Для этого требуется сравнить D с тремя числами: $A + B$, $B + C$ и $A + C$. Если D равно хотя бы одному из них, то ответ — «`YES`», иначе — «`NO`».

Для решения на частичный балл заметим, что в первых трёх группах тестов сумма оставшихся чисел не зависит от того, какое число будет выбрано для стирания, так как числа A, B и C равны. Поэтому достаточно единственной проверки на равенство чисел $2A$ и D.

В первой группе тестов эту проверку можно осуществить на листочке: $2A = 10$, а $D = 11$. Поэтому достаточно написать программу, которая выводит «`NO`».

Для прохождения второй группы тестов нужно считать число D. Так как число D вводится последним, до этого необходимо считать и остальные числа. Однако известно, что $A = B = C = 5$, поэтому D достаточно сравнить с константой 10.

Для прохождения третьей группы тестов D нужно сравнить с числом $2A$.

В четвёртой группе тестов сумма двух оставшихся чисел после стирания уже зависит от выбранного числа. Но так как $A = B$, достаточно только разобрать два случая из трёх: если стереть число A и если стереть число C.

Критерии оценивания:

№	Баллы	Ограничения		Необх. группы
		D	Дополнительно	
1	7	$D = 11$	$A = B = C = 5$	—
2	38	$D \leqslant 300$	$A = B = C = 5$	1
3	16	$D \leqslant 300$	$A = B = C$	1−2
4	29	$D \leqslant 300$	$A = B$	1−3
5	10	$D \leqslant 300$	—	1−4

Баллы выставляются автоматически проверяющей системой.

Задача C. Забор

Для решения задачи необходимо написать алгоритм, генерирующий подходящую строку. Примером такой строки является: «`queryqueryquery...`» (всего 100 букв).

Но как такую строку придумать? Начнем поэтапно решать задачу.

Чтобы пройти первые 3 теста можно попробовать дописать несколько букв к строке из примера. Многие строки подошли бы. При этом можно убедиться в корректности строки как ручной проверкой на листочке, так и просто путем отправки придуманного слова на проверку.

Изучим какую-нибудь хорошую строку, например, «`dogabc`». Почему эта строка подходит? Первое наблюдение: для того, чтобы в строке не появлялись палиндромы (будем считать, что по умолчанию имеются ввиду палиндромы нееденичной длины) достаточно, чтобы все буквы были различны. Действительно, если никакие буквы не совпадают, то все подстроки гарантированно читаются по-разному слева направо и справа налево, например, из-за того, что первая и последняя буквы будут отличаться. Таким образом, слово, в котором все буквы различны, точно подходит.

Значит, для прохождения первых 5 тестов достаточно вывести весь латинский алфавит. Но попробуем приписать еще какую-то букву к алфавиту, и почти наверняка такое решение пройдет и 6-й тест. Действительно, даже если мы найдем подстроку, в которой первая и последняя буквы будут совпадать, необходимо чтобы совпали и некоторые другие пары букв, например, вторая и предпоследняя буквы.

Ключевая идея: если в строке найдена подстрока-палиндром длины l, то можно убрать первую и последнюю буквы в этой подстроке, и тогда полученная подстрока длины $l-2$ должна остаться палиндромом. Но так можно делать, пока от исходной подстроки не останется подстрока длины 2 или 3, которая должна быть палиндромом. Значит, чтобы в строке отсутствовали подстроки-палиндромы, достаточно только гарантировать, чтобы все подстроки длины 2 и 3 не являлись палиндромами.

Но тогда не имеют значения повторения символов, если они расположены хотя через два символа друг от друга. Поэтому достаточно придумать строку длины хотя бы 3, в которой все символы различны, а потом ее дублировать до тех пор, пока ее длина не будет равна 100 (при превышении лишние символы можно отбросить). Пример такой строки приведен в начале разбора.

Критерии оценивания:

№	Баллы за тест	N
1	11	$N = 4$
2	9	$N = 5$
3	5	$N = 6$
4	8	$N = 10$
5	17	$N = 26$
6	22	$N = 27$
7	11	$N = 30$
8	8	$N = 50$
9	9	$N = 100$

Баллы выставляются автоматически проверяющей системой

Задача D. Команда

Первые три группы тестов позволяют убедиться в правильном понимании и реализации сравнения кандидатов. Для прохождения этих групп достаточно сохранить всех кандидатов в переменные и сравнить их при помощи условных конструкций.

Для прохождения четвёртой группы тестов достаточно найти двух самых сильных кандидатов по единственному критерию — по числу a_i. Для этого заведём две переменные, в которых сохраним результаты первого и второго по силе кандидатов среди уже прочитанных, а также две переменные для вывода ответа — их номера. Чтобы обновить эти значения при чтении результатов очередного кандидата, нужно сравнить новые результаты с теми, что лежат в переменных. Возможны три исхода: либо результаты в переменных не меняются, либо обновляется результат второго по силе, либо обновляется результат самого сильного, но тогда его старые значения (количество задач и номер) необходимо переписать в значения второго по силе кандидата. Заметим, что проверку на номер кандидата можно реализовать и через строгость неравенств при обновлении значений переменных (если решено столько же задач, то обновлять не нужно, т.к. номер больше).

Для прохождения пятой группы тестов нужно сравнивать кандидатов не только по a_i, но и по $a_i + b_i$, для чего необходимо завести дополнительные переменные для суммарного количества задач, решённых на двух соревнованиях.

Полное решение задачи отличается от решения для пятой группы тестов только тем, что до завершения чтения данных неизвестно, какое из соревнований сложнее. Поэтому во время чтения данных удвоим число переменных и будем считать два разных ответа: для случая, когда первое соревнование сложнее второго, и наоборот. Считав все данные, посмотрим, какое соревнование сложнее, и выберем из двух посчитанных ответов нужный.

Таким образом, задачу можно решить на полный балл и без использования массивов. Однако также можно решить задачу, если сохранить данные кандидатов в виде массива троек чисел (суммарное количество задач, количество задач на более сложном соревновании, номер), после чего отсортировать этот массив и взять двух лучших кандидатов. Но, во-первых, при $N \leqslant 50\,000$ нужно использовать эффективный алгоритм сортировки. Можно сортировать и выбором/пузырьком, но тогда надо прерывать сортировку сразу, как только два максимальных элемента займут свои места. Во-вторых, надо не напутать с порядком, т.к. по количеству задач надо сортировать по возрастанию, а по номеру — по убыванию.

Критерии оценивания:

№	Баллы	Ограничения		Необх. группы
		N	Дополнительно	
1	8	$N = 2$	$b_i = 0$	—
2	27	$N = 2$	—	1
3	6	$N \leqslant 3$	—	1, 2
4	26	$N \leqslant 50000$	$b_i = 0$	1
5	15	$N \leqslant 50000$	$a_i \geqslant b_i$	1, 5
6	18	$N \leqslant 50000$	—	1–6

Баллы выставляются автоматически проверяющей системой.

Задача E. Произведение на отрезке

Попробуем сначала понять, что в целом происходит в задаче. Для этого можно решить первые две группы тестов, причем для их решения требуются только знания условных операторов. Действительно, все возможные тесты можно перебрать на листочке (их всего 12) и решить каждый из них как математическую задачу, например, полным перебором. Аналогично можно решить и третью группу тестов.

Однако как только становится ясным, что происходит в задаче и как написать полный перебор, следует реализовать его любым способом для прохождения групп 1–4. Например, функцию $s(x)$ можно реализовать только для однозначных и двузначных x как сумму двух чисел: остатка от деления x на 10 и целой части от деления x на 10. Эти вычисления не обязательно реализовывать отдельной функцией — можно просто считать это значение как формулу.

Решение начнем с перебора чисел l и r при помощи двух вложенных циклов, причем внутренний должен начать перебор числа r с $r = l$. Затем переберем все числа из отрезка $[l; r]$ еще одним вложенным циклом, и перемножим их между собой (для этого потребуется вспомогательная переменная; обратите внимание, что в отличие от суммы ее начальное значение равно 1, а не 0), а затем проверим делимость на 9 при помощи взятия остатка от деления на 9. Если произведение не поделилось, то к ответу нужно прибавить единицу за найденный отрезок. Так как результат не превосходит $s(1) \cdot s(2) \cdot \ldots \cdot s(12) = 2177280$, хранить вычисления можно в стандартном 32-битном целом числе.

Далее необходимо сделать несколько оптимизаций этого решения. Чтобы пройти группы 5–6 заметим, что текущий полный перебор делает много лишней работы. На итерации с числами (l, r), где $r > l$ число $s(l) \cdot s(l+1) \cdot \ldots \cdot s(r)$ будет посчитано циклом при помощи не менее, чем $r - l$ операций умножения. Однако на прошлой итерации (для чисел $(l, r-1)$) уже было посчитано число $s(l) \cdot s(l+1) \cdot \ldots \cdot s(r-1)$, и для получения требуемого его нужно только умножить на $s(r)$.

Пусть вспомогательная переменная будет сохранять свое значение между итерациями цикла, перебирающего r (для этого ее можно, например, инициализировать в теле внешнего цикла), тогда в его теле можно заменить цикл на единственную операцию — домножение вспомогательной переменной на $s(r)$, и вычисления будут корректными.

Остается только изменить способ, при помощи которого считается $s(x)$. Можно подобно тому, как эта функция реализовывалась ранее, написать версию для трёх- четырёхзначных x. Но можно и написать $s(x)$ для любых x. Для этого нужно завести дополнительную переменную, изначально равную 0, в которую просуммируем все цифры числа x при помощи цикла. Этот цикл должен идти до тех пор, пока x не обнулится. Обновление переменной в цикле происходит следующим образом: прибавим к ней последнюю цифру числа x при помощи взятия остатка от деления x на 10, а после этого уберем последнюю цифру числа x, заменив x на целую часть от деления x на 10. Такой алгоритм обнуляет x, что следует учитывать при реализации, однако его идея и реализация довольно простые.

Но важно отметить, что в $5-10$ группах тестов произведение $s(l) \cdot s(l+1) \cdot \ldots \cdot s(r)$ может превышать даже 64-битный тип данных, притом время работы программы будет сильно увеличено, если использовать неограниченно длинные целые числа.

Отметим, что конечное решение не требует решения этой проблемы.

Более понятное решение проблемы с переполнением — хранить вместо произведения только вхождение степени тройки в произведение: пусть $s(l) \cdot s(l+1) \cdot \ldots \cdot s(r) = 3^k \cdot C$, тогда будем хранить только число k. Действительно, число делится на 9, если степень вхождения тройки в произведение не меньше 2 ($k \geqslant 2$). Но степень вхождения тройки в некоторое число $s(x)$ может быть нетрудно вычислено перебором при помощи цикла, а степень вхождения произведения считается как сумма степеней вхождения множителей.

Есть и другое решение. Вспомним следующий факт: если числа y и z дают одинаковый остаток от деления на любое натуральное число m, то при замене числа y на z в любом арифметическом выражении, операциями которого являются $+$, $-$ и/или $\cdot$, остаток от деления этого арифметического выражения на m не изменится.

Заметим, что числа z и $z \bmod 9$, где mod — операция взятия остатка от деления, дают одинаковый остаток при делении на 9, равный $z \bmod 9$. Значит, можно перемножать не числа $s(l), s(l{+}1), \ldots, s(r)$, а числа $(s(l) \bmod 9), (s(l{+}1) \bmod 9), \ldots, (s(r) \bmod 9)$. Более того, после каждого перемножения можно заменить промежуточный результат на остаток от деления этого промежуточного результата на 9. Но так как $y \bmod z < z$, во вспомогательной переменной можно всегда поддерживать результат, не превышающий $9 \cdot 9 = 81$. Таким образом можно решить проблему с переполнением.

Для решения $7-8$ групп нужно придумать еще какую-нибудь оптимизацию. Тогда воспользуемся фактом о взятии остатка от деления немного с другой стороны. Для этого вспомним (обобщенный) признак делимости на 9, который утверждает, что $s(x)$ и x дают одинаковый остаток при делении на 9. Значит, посчитать остаток от деления $s(l) \cdot s(l+1) \cdot \ldots \cdot s(r)$ на 9 — это то же самое, что и посчитать остаток от деления $l \cdot (l+1) \cdot \ldots \cdot r$ на 9.

Но заметим, что среди любых 9 последовательных натуральных чисел обязательно встретится число, которое делится на 9. Но если такое число нашлось на отрезке $[l, r]$, то оно обязательно найдется на всех отрезках $[l, r']$ для всех $r' > r$. Тогда оптимизация состоит в следующем: не всегда нужно увеличивать правую границу r для заданной левой границы l до B — можно выйти из вложенного цикла еще в тот момент, когда произведение впервые поделилось на 9. Другой вариант: задать границей перебора r число $\min(l+8, B)$.

Чтобы решить задачу на полный балл, заметим, что все подходящие отрезки находятся между числами, кратными 9, и, более того, между любыми двумя соседними числами, кратными 9, которые содержатся в $[A; B]$, количество отрезков одинаково и равно 27.

Первая идея напрямую следует из того, что мы уже знаем про подходящие отрезки — если на отрезке есть число, кратное 9, то произведение чисел на этом отрезке точно делится на 9. Но тогда если разрезать весь отрезок $[A; B]$ числами кратными 9 (для определенности будем считать, что числа, делящиеся на 9, уничтожаются после разрезания), то для каждого отрезка из полученных задачу можно решать независимо, а итоговый ответ посчитать как сумму по всем отрезкам, которые получились разрезанием.

Вторая идея основана на том, что числа z и $z+9$ дают одинаковый остаток от деления на 9, а значит остаток от деления на 9 произведения чисел $l, l{+}1, \ldots, r$ равен остатку от деления на 9 произведения чисел $l+9, l+10, \ldots, r+9$. Можно показать, что ответ на задачу при $A = 1, B = 8$ равен 27, но тогда из рассуждений следует, что ответ при сдвиге границ на 9, то есть для $A = 9k{+}1, B = 9k{+}8$ при любом натуральном k, тоже равен 27.

Заметим, что тогда большинство отрезков, получившихся после разрезания, будут давать в общий ответ число 27 из-за того, что их границы были получены разрезаниями по числам, которые делятся на 9 (то есть их вид: $[9k+1; 9k+8]$). Поэтому остается только перебором посчитать ответ для самого левого и правого отрезка.

Понятно, что явно строить такое разбиение не нужно. Для решения задачи на полный балл нужно только выделить из $[A; B]$ самый длинный подотрезок $[L; R]$, начинающийся с числа, кратного 9, и заканчивающийся числом, кратным 9. Тогда результат на $[L; R]$ можно посчитать как результат на отрезке $[1; 8]$ (равный 27), умноженный на $(R - L)/9$. Оставшиеся подотрезки с начала и с конца можно проверить простым перебором, например как в решении 4-й подзадачи.

Критерии оценивания:

№	Баллы	Ограничения		Необх. группы
		A	B	
1	8	$A = 1$	$B \leqslant 5$	—
2	11	$A = 1$	$B \leqslant 12$	1
3	4	$A \leqslant B$	$B \leqslant 5$	1
4	18	$A \leqslant B$	$B \leqslant 12$	1−3
5	15	$A \leqslant B$	$B \leqslant 500$	1−4
6	5	$A \leqslant B$	$B \leqslant 2\,000$	1−5
7	6	$A \leqslant B$	$B \leqslant 50\,000$	1−6
8	7	$A \leqslant B$	$B \leqslant 150\,000$	1−7
9	14	$A = 1$	$B \leqslant 5 \cdot 10^8$	1, 2
10	12	$A \leqslant B$	$B \leqslant 5 \cdot 10^8$	1−9

Баллы выставляются автоматически проверяющей системой