Муниципальный этап Всероссийской олимпиады школьников по информатике

в 2015 – 2016 учебном году

Разборы решений и идеи тестов

Муниципальный этап Всероссийской олимпиады школьников по информатике в 2015 – 2016 учебном году

9 класс

Время выполнения задач — 4 часа
Ограничение по времени — 2 секунды на тест
Ограничение по памяти — 64 мегабайта

9.1. «Наибольший хвост». Заданы три натуральных числа n_1, n_2, n_3, не меньших 10. Выведите то из них, две последние цифры которого, будучи рассмотрены в том же порядке, что и в десятичной записи числа, дают наибольшее двузначное число.

Формат входа: В единственной строке через пробел заданы три натуральных числа n_1, n_2, n_3 ($10 \leqslant n_i \leqslant 10^6$).

Формат выхода: Выведите единственное число, имеющее наибольшее двузначное число в разряде десятков и единиц. Если два или все три числа имеют одинаковые числа в этих разрядах, выведите любое из них.

Пример

Вход:	*Выход:*
`198 691 19`	`198`

9.2. «Победа в бою». В свободное от уроков и тренировок по плаванию время Петя Торопыжкин играет в компьютерные игры. В очередном бою у Пети имеется отряд из k_1 пехотинцев, каждый из которых имеет силу удара s_1 и здоровье h_1. У компьютерного противника также имеется отряд из k_2 пехотинцев, каждый с ударом s_2 и здоровьем h_2. Отряд Пети первым наносит удар по отряду противника, затем атакует отряд противника, потом снова Петин отряд и т.д. Отряд из k человек (неважно, есть ли среди них раненый боец или нет) с ударом s наносит $k \cdot s$ единиц урона, которые последовательно вычитаются из здоровья наиболее раненного члена атакуемого отряда. Если единиц урона больше, чем здоровье такого члена отряда, то он умирает и здоровье начинает списываться у другого солдата (который также может погибнуть, если урона слишком осталось много) — до тех пор, пока есть живые солдаты или есть очки урона.

Напишите программу, которая определяет по указанным входным данным, какой отряд выживет в таком бою и сколько солдат останется в его составе.

Формат входа: В первой входной строке через пробел указаны три целых числа k_1, s_1, h_1. Во второй строке так же через пробел указаны числа k_2, s_2, h_2. При этом $1 \leqslant k_i$, s_i, $h_i \leqslant 1000$.

Формат выхода: Выведите через пробел два целых числа. Первое — 1, если выживет Петин отряд, или 2, если выиграет противник. Второе число — численность выжившего отряда (включая и раненного солдата, если такой есть).

Пример 1

Вход:	*Выход:*
1 1 100 1 1 100	1 1

Пример 2

Вход:	*Выход:*
1 1 100 1 1 101	2 1

9.3. «Новый химический элемент». Сидя на уроке химии, Петя Торопыжкин мечтал о том, как он откроет новый химический элемент. Потом он задумался о его названии и обозначении. И тут он заметил на парте в кабинете химии длинное слово, состоящее из заглавных латинских букв, и решил, что обозначением будет сочетание двух последовательных букв из увиденного слова. Но сочетаний много, и Петя решил выбрать из них минимальное в лексикографическом порядке.

Напомним, что лексикографический порядок сравнения слов — это порядок, принятый в словарях: слова сравниваются по первым буквам, при их равенстве по вторым, при равенстве вторых — по третьим и т.д., пока будет найдена несовпадающая пара букв, которая определит порядок слов, или одно из слов не кончится; в последнем случае более короткое слово, совпадающее с началом более длинного, считается меньшим.

Помогите Пете, написав программу, которая по введённому слову найдёт минимальное в лексикографическом порядке двухбуквенное подслово.

Формат входа: В единственной строке задано слово, состоящее из заглавных букв латиницы. Длина слова не менее 2 и не более 255 символов.

Формат выхода: Выведите единственную строку, содержащую минимальное в лексикографическом порядке двухбуквенное подслово входной строки.

Пример

Вход:	*Выход:*
ZCYBWA	BW

9.4. «Странное сравнение». На математическом кружке Петя Торопыжкин прослушал лекцию на тему отношений порядка и узнал, что целые числа можно сравнивать не только так, как это принято на уроках математики. Например, можно считать, что любое чётное число меньше любого нечётного, а пару чётных чисел или пару нечётных чисел уже сравнивать по обычным правилам — и это тоже будет порядком на натуральных числах! Правда, непривычным. В портфеле у него завалялся листочек с каким-то набором целых чисел, и он заинтересовался, какое из чисел меньше (в смысле обычного порядка): k-е число в наборе, отсортированном по обычным правилам, или k-е в наборе, отсортированном в смысле порядка, описанного выше (когда любое чётное число меньше любого нечётного). Помогите ему, написав программу, которая будет отвечать на этот вопрос.

Формат входа: В первой строке через пробел задано два целых числа: n — количество чисел на Петином листочке — и k — позиция чисел в отсортированных наборах ($1 \leqslant n \leqslant 10^5$, $1 \leqslant k \leqslant n$). Во второй строке через пробел задано n целых

чисел, по модулю не превосходящих 10^6. меньшее из чисел, стоящих на k-м месте в наборах, отсортированных по указанным правилам (считаем, что первое число в наборе имеет индекс 1).

Пример 1		**Пример 2**	
Вход:	*Выход:*	*Вход:*	*Выход:*
3 3 5 100 6	5	3 3 100 6 2	100

Примечание: В первом примере на третьем месте будет в необычном порядке будет стоять 5, поскольку в смысле необычного порядка любое нечётное число больше любого чётного; при обычной сортировке — число 100; меньшее из них — 5. Во втором примере нет нечётных чисел, поэтому обычный и необычный порядки просто совпадают и на третьем месте в обоих случаях стоит 100.

9.5. «Кратные делители». Петя Торопыжкин изучил программирование и стал решать разные математические задачи на компьютере. В частности, он вспомнил задачку из 6-го класса на понятие делимости: пусть имеется набор натуральных чисел $\{n_i\}$ и натуральное число m. Вопрос: делителем какой степени является число m для совокупного НОК чисел n_i?

Совокупное НОК (наименьшее общее кратное) набора натуральных чисел — это наименьшее натуральное число, для которого любое число из имеющегося набора будет делителем.

Помогите Пете, напишите соответствующую программу.

Формат входа: В первой строке через пробел задано два целых числа: n — количество чисел в наборе — и число m ($1 \leqslant n \leqslant 10^5$, $2 \leqslant m \leqslant 10^6$). Во второй строке через пробел задано n натуральных чисел, не превосходящих 10^6.

Формат выхода: Выведите единственное неотрицательное целое число — степень делителя m у совокупного НОК чисел n_i.

Пример 1		**Пример 2**	
Вход:	*Выход:*	*Вход:*	*Выход:*
3 2 100 5 2	2	3 3 100 5 2	0

Примечание: В примерах $\text{НОК}(100, 5, 2) = 100$. В первом примере 100 делится на $4 = 2^2$, но не делится на $8 = 2^3$. Во втором примере 100 не делится на 3, то есть делится на $1 = 3^0$.

Муниципальный этап Всероссийской олимпиады школьников по информатике в 2015 – 2016 учебном году
9 класс. Разбор решений и идеи тестов

9.1. «Наибольший хвост». *Заданы три натуральных числа n_1, n_2, n_3, не меньших 10. Выведите то из них, две последние цифры которого, будучи рассмотрены в том же порядке, что и в десятичной записи числа, дают наибольшее двузначное число.*

Задача, по мнению программного комитета, является утешительной. Для решения требуется сравнивать остатки от деления данных чисел на 100.

Идеи тестов:

1–3. Случайные тесты, максимум единственен.
4–8. Случайные тесты, два числа различны, но являются максимумом в указанном смысле.
9–10. Случайные тесты, три числа различны, но все являются максимумом в указанном смысле.

9.2. «Победа в бою». *В свободное от уроков и тренировок по плаванию время Петя Торопыжкин играет в компьютерные игры. В очередном бою у Пети имеется отряд из k_1 пехотинцев, каждый из которых имеет силу удара s_1 и здоровье h_1. У компьютерного противника также имеется отряд из k_2 пехотинцев, каждый с ударом s_2 и здоровьем h_2. Отряд Пети первым наносит удар по отряду противника, затем атакует отряд противника, потом снова Петин отряд и т.д. Отряд из k человек (неважно, есть ли среди них раненый боец или нет) с ударом s наносит $k \cdot s$ единиц урона, которые последовательно вычитаются из здоровья наиболее раненного члена атакуемого отряда. Если единиц урона больше, чем здоровье такого члена отряда, то он умирает и здоровье начинает списываться у другого солдата (который также может погибнуть, если урона слишком осталось много) — до тех пор, пока есть живые солдаты или есть очки урона.*

Напишите программу, которая определяет по указанным входным данным, какой отряд выживет в таком бою и сколько солдат останется в его составе.

Вероятно, можно вывести формулу, оценивающую по входным данным игрока, чей отряд победит, и остаточную численность его отряда. Однако прямое моделирование боя является представляется более простым, поэтому данную задачу следует отнести к классу симуляторов, то есть задач, требующих аккуратной технической реализации приведённых правил изменения состояния системы. Программный комитет считает данную задачу несложной.

Как представляется программному комитету, для моделирования каждого отряда удобнее представить его в виде двух величин: здоровье w раненого бойца (величина, меньшая максимального здоровья, возможно, равная нулю, если в отряде нет раненых бойцов) и количество g полностью здоровых бойцов. Вначале $w = 0$ и $g = k$.

Пусть при очередной атаке на отряд ему наносится P очков урона. Если $P \leqslant w$, вычитаем из w величину P, на этом атака заканчивается. В противном случае вычитаем из P величину w и кладём $w = 0$. Результат вычитания будем обозначать P'. Если P' не меньше суммарного здоровья оставшегося отряда, то есть, если $P' \geqslant g \cdot h$, то отряд уничтожен полностью, бой заканчивается. Иначе вычисляем количество бойцов, убитых этой атакой $P' \operatorname{div} h$ и вычитаем эту величину из g. Затем, если $P' \bmod h \neq 0$, то один боец получает частичные ранения в размере $P' \bmod h$, то есть уменьшаем результирующее значение g ещё на 1 и присваиваем $w = h - P' \bmod h$. Если же $P' \bmod h = 0$, то оставляем результирующее значение g, как есть, и полагаем $w = 0$. Атака закончена, переходит к расчёту следующей атаки до тех пор, пока один из отрядов не будет полностью уничтожен.

Ответом будет величина g для оставшегося отряда, увеличенная на 1, если для этого отряда $w \neq 0$ (учитываем раненого бойца).

Идеи тестов:

1. Параметры бойцов обоих отрядов одинаковы, удар кратен здоровью, выигрывает Петин отряд.
2. Параметры бойцов отрядов неодинаковы, удары кратны здоровью бойцов противника, выигрывает Петин отряд.
3. Параметры бойцов неодинаковы, удар Петиных бойцов кратен здоровью бойцов противника, удар бойцов противника не кратен здоровью Петиных бойцов, выигрывает Петин отряд, имея только одного раненого бойца.
4. Параметры бойцов неодинаковы, удар Петиных бойцов кратен здоровью бойцов противника, удар бойцов противника не кратен здоровью Петиных бойцов, выигрывает Петин отряд, имея только одного здорового бойца.
5. Параметры бойцов неодинаковы, удар Петиных бойцов кратен здоровью бойцов противника, удар бойцов противника не кратен здоровью Петиных бойцов, выигрывает Петин отряд, имея несколько здоровых бойцов.
6. Параметры бойцов неодинаковы, удар Петиных бойцов кратен здоровью бойцов противника, удар бойцов противника не кратен здоровью Петиных бойцов, выигрывает Петин отряд, имея здоровых и раненого бойцов.

7–10. То же, что в тестах 3–6, только удар бойцов противника не кратен здоровью Петиных бойцов.

11–20. То же, что в тестах 1–10, только ситуация симметрична — выигрывает противник Пети.

21–25. Случайные тесты.

9.3. «Новый химический элемент». *Сидя на уроке химии, Петя Торопыжкин мечтал о том, как он откроет новый химический элемент. Потом он задумался о его названии и обозначении. И тут он заметил на парте в кабинете химии длинное слово, состоящее из заглавных латинских букв, и решил, что обозначением будет сочетание двух последовательных букв из увиденного слова. Но сочетаний много, и Петя решил выбрать из них минимальное в лексикографическом порядке.*

Напомним, что лексикографический порядок сравнения слов — это порядок, принятый в словарях: слова сравниваются по первым буквам, при их равенстве по вторым, при равенстве вторых — по третьим и т.д., пока будет найдена несовпадающая пара букв, которая определит порядок слов, или одно из слов не кончится; в последнем случае более короткое слово, совпадающее с началом более длинного, считается меньшим.

Помогите Пете, написав программу, которая по введённому слову найдёт минимальное в лексикографическом порядке двухбуквенное подслово.

Данная задача, в каком-то смысле, походит на задачу 9.1 — здесь тоже надо найти экстремальный элемент из набора. Однако разница состоит в том, что в первой задаче набор состоит из трёх элементов и при её решении можно обойтись несколькими сравнениями, а в данной задаче необходимо реализовать алгоритм поиска экстремального элемента в наборе, размер которого заранее неизвестен. Также необходимо реализовать корректное сравнение двух двухбуквенных слов.

Пусть l — длина заданной строки s. Тогда будем задавать двухбуквенное подслово индексом i его первого символа, $1 \leqslant i < l$ (считаем, что индексы отсчитываются от 1). Пусть $i_{\min}$ — индекс текущего найденного минимального подслова. Вначале $i_{\min} = 1$. Перебирая индекс i текущего подслова от 2 до $l - 1$ и сравнивая подслово, задаваемое этим индексом, с текущим минимальным подсловом, задаваемым индексом $i_{\min}$, найдём минимальное подслово во всей строке.

Слово, задаваемое индексом k, меньше слова, задаваемого индексом m, если $s[k] < s[m]$ или если $s[k] = s[m]$ и $s[k+1] < s[m+1]$.

Идеи тестов:

1. Минимальный тест — двухбуквенная строка.
2. Два вида символов, строка чётной длины вида $ABAB\ldots AB$.
3. Два вида символов, строка чётной длины вида $BABA\ldots BA$.
4. Два вида символов, строка нечётной длины вида $ABAB\ldots ABA$.
5. Два вида символов, строка нечётной длины вида $BABA\ldots BAB$.
6. Два вида символов, случайные сочетания двух символов, нет двух меньших символов, идущих подряд.
7. Два вида символов, случайные сочетания двух символов, есть два меньшие символа, идущие подряд.

8–11. Случайные тесты, много видов символов, нет двух меньших символов, идущих подряд, длина строки меньше 255.

12–14. Случайные тесты, много видов символов, есть два меньшие символа, идущие подряд, длина строки меньше 255.
15–17. Случайные тесты, много видов символов, нет двух меньших символов, идущих подряд, длина строки равна 255.
18–20. Случайные тесты, много видов символов, есть два меньшие символа, идущие подряд, длина строки равна 255.

9.4. «Странное сравнение». *На математическом кружке Петя Торопыжкин прослушал лекцию на тему отношений порядка и узнал, что целые числа можно сравнивать не только так, как это принято на уроках математики. Например, можно считать, что любое чётное число меньше любого нечётного, а пару чётных чисел или пару нечётных чисел уже сравнивать по обычным правилам — и это тоже будет порядком на натуральных числах! Правда, непривычным. В портфеле у него завалялся листочек с каким-то набором целых чисел, и он заинтересовался, какое из чисел меньше (в смысле обычного порядка): k-е число в наборе, отсортированном по обычным правилам, или k-е в наборе, отсортированном в смысле порядка, описанного выше (когда любое чётное число меньше любого нечётного). Помогите ему, написав программу, которая будет отвечать на этот вопрос.*

Собственно, прямолинейное решение данной задачи подразумевает двукратную сортировку заданного массива, каждый раз с указанным порядком: один раз с обычным, другой раз с необычным. После каждой сортировки из отсортированного массива извлекается k-е число, результаты сравниваются, выдаётся их минимум.

Таким образом, имеются две сложности. Первая состоит в том, чтобы организовать процедуру сортировки, которая работает достаточно быстро. Вторая — в том, чтобы организовать сортировку в необычном порядке. Первая сложность, в принципе, вполне по силам подготовленному школьнику: нужно просто использовать библиотечную функцию сортировки. Однако, чтобы организовать сортировку в нестандартном порядке с использованием стандартной функции сортировки требуются хорошие знания библиотеки используемого языка. Если же при этом отказаться от библиотечной функции, то необходимо умение самому писать хорошие сортировки (быструю сортировку, сортировку слиянием, сортировку кучей).

В принципе, возможно решение, опирающееся в трёх четвертях случаев на однократную сортировку массива (в необычном порядке; что, впрочем, не снимает указанные сложности, а лишь делает решение чуть более эффективным). Действительно, пусть в необычном порядке на k-м месте стоит некоторое нечётное число a. При перестановке в обычном порядке ***перед*** a новых чисел возникнуть не может: все чётные уже и так стояли перед a, а все нечётные, стоящие справа, больше a, и перед a встать не могут. Стало быть, a при пересортировке a может либо остаться на месте (это значит, что все чётные числа так и остались левее a,

то есть все чётные числа меньше a), либо переместиться левее (это означает, что есть чётные числа бóльшие a).

Пусть теперь в необычном порядке на k-м месте стоит некоторое чётное число a. При пересортировке оно не может переместиться левее: влево числа могут лишь добавляться. Стало быть, в этом случае число a может либо остаться на месте (нечётные числа левее a не становились, то есть все нечётные больше a), либо переместиться правее (то есть имеются нечётные числа, меньшие a).

Таким образом, логически возможны 4 случая:

1. в необычном порядке на k-м месте стоит некоторое нечётное число a; все чётные числа меньше a; искомый результат — число a;
2. в необычном порядке на k-м месте стоит некоторое нечётное число a; имеются чётные числа, бóльшие a; искомый результат — число a;
3. в необычном порядке на k-м месте стоит некоторое чётное число a; все нечётные числа больше a; искомый результат — число a;
4. в необычном порядке на k-м месте стоит некоторое чётное число a; имеются нечётные числа, меньшие a; искомый результат — число, стоящее на k-месте в массиве, отсортированном в обычном порядке.

Таким образом, видно, что если в необычном порядке на k-м месте стоит нечётное число a, то оно же будет результатом. В случае, когда это число чётное, оно будет ответом, если нет нечётных, меньших его. Этот факт можно установить за один проход массива. И, наконец, если число a чётное и имеются нечётные, меньшие a, то требуется сортировка в смысле обычного порядка чисел.

Идеи тестов:

1. Минимальный тест: $n = k = 1$. Единственное число — чётное.
2. Минимальный тест: $n = k = 1$. Единственное число — нечётное.
3. Минимальный тест: $n = 2$, $k = 2$, ответ — большее из этих двух чисел (в обычном порядке).
4. Минимальный тест: $n = 2$, $k = 2$, ответ — меньшее из этих двух чисел (в обычном порядке).
5. Минимальный тест: $n = 2$, $k = 1$, ответ, соответственно — меньшее из этих двух чисел (в обычном порядке).
6. $n = 20$, нет нечётных чисел.

7–10. $n = 20$, есть нечётные числа, рассматриваются случаи 1–4.

11–15. То же, что в тестах 6–10, только $n = 5000$.

16–20. То же, что в тестах 6–10, только $n = 10000$.

21–25. То же, что в тестах 6–10, только $n = 100000$.

9.5. «Кратные делители». *Петя Торопыжкин изучил программирование и стал решать разные математические задачи на компьютере. В частности,*

он вспомнил задачку из 6-го класса на понятие делимости: пусть имеется набор натуральных чисел $\{n_i\}$ и натуральное число m. Вопрос: делителем какой степени является число m для совокупного НОК чисел n_i?

Совокупное НОК (наименьшее общее кратное) набора натуральных чисел — это наименьшее натуральное число, для которого любое число из имеющегося набора будет делителем.

Помогите Пете, напишите соответствующую программу.

Задача идейно несложна, однако полное её решение не вполне прямолинейно, а также для своей реализации требует определённого уровня программирования, что может вызвать трудности у девятиклассников.

Далее в тексте $\sigma(b, a)$ — степень делителя b у числа a. Алгоритм вычисления этой величины достаточно прост:

```
σ := 0;
while a mod b = 0 do begin
  inc(σ);
  a := a mod b;
end;
```

Единственно, при этом разрушается значение a, поэтому этот код разумно оформить как функцию.

Прямолинейное решение состоит в нахождении НОК всех предложенных чисел a_i и в последующем подсчете степени указанного делителя у найденного НОК. Недостаток предложенного метода заключается в том, что НОК может оказаться настолько большим, что не войдёт в тип int. Те, кто пишут на Java или на Python, могут привлечь библиотеки длинной арифметики. Но на самом деле, этого не требуется.

Решение основано на элементарном наблюдении, что степень того или иного простого делителя у НОК не может быть больше степени этого делителя у чисел, НОК которых подсчитывается. Однако это, вообще говоря, неверно для составных делителей. Действительно, 63 не является делителем ни 7, ни 9, но является делителем их НОК, равного как раз 63. То есть сомножители, обеспечивающие тот факт, что m является делителем НОК, могут «приходить» из разных чисел, у которых это НОК вычисляется.

Стало быть, надо вычислить степени простых делителей m, встречающихся чисел a_i. Идея оптимального алгоритма состоит в следующем:

1) раскладываем m на простые множители, запоминая значение каждого множителя p_j и его степень $d_j = \sigma(p_j, m)$; отметим, что при ограничении $m \leqslant 10^6$ число m не может иметь более 8 различных простых делителей: 2, 3, 5, 7, 11, 13, 17, 19 — их произведение уже больше 10^6; если брать следующие простые числа, то миллион превысит произведение меньшего их количества;

2) последовательно считываем предложенные числа a_i, для каждого простого делителя p_j числа m подсчитываем его степень $g_{i,j} = \sigma(p_j, a_i)$ у каждого из

чисел a_i и находим максимум $g_j^* = \max\limits_i g_{i,j}$ из величин $g_{i,j}$ для каждого j;

3) понятно, что если m является делителем степени k у какого-то числа l, то каждый простой делитель p_j числа m должен быть делителем числа l степени не ниже $k \cdot d_j$, то есть $k \leqslant \sigma(p_j, l) \text{ div } d_j$, и при этом хотя бы один делитель $p_{\bar{j}}$ является делителем степени ниже $(k+1) \cdot d_{\bar{j}}$. Отсюда вытекают действия заключительного этапа: ищем минимум $k^* = \min\limits_j (g_j^* \text{ div } d_j)$ — это и будет ответ.

Идеи тестов:

1. Минимальный тест: $n = 1$, заданное число взаимно просто с m.
2. Минимальный тест: $n = 1$, заданное число имеет m делителем степени 1.
3. Минимальный тест: $n = 1$, заданное число имеет m делителем степени больше 1.
4. $n = 25$, $m = p_1 \cdot p_2$ есть произведение двух простых сомножителей, m взаимно просто с НОК n_i.
5. $n = 25$, $m = p_1 \cdot p_2$, степень делителя m у НОК есть максимум степеней делителя m у чисел n_i.
6. $n = 25$, $m = p_1 \cdot p_2$, m не является делителем ни одного n_i, одно из чисел из набора содержит в качестве делителя один из простых сомножителей m, другое — другой.

7–9. То же, что в тестах 4–6, только НОК не входит в int.

10. $n = 25$, $m = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3}$, m взаимно просто с НОК n_i.
11. $n = 25$, $m = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3}$, степень делителя m у НОК есть максимум степеней делителя m у чисел n_i.
12. $n = 25$, $m = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3}$, m не является делителем ни одного n_i, n_i «приносят» в НОК два делителя m.
13. $n = 25$, $m = p_1^{k_1} \cdot p_2^{k_2} \cdot p_3^{k_3}$, m не является делителем ни одного n_i, n_i «приносят» в НОК по одному делителю m.

14–17. То же, что в тестах 10–13, только $n = 50$ и НОК не входит в int.

18–21. То же, что в тестах 10–13, только $n = 50$ и НОК не входит в int64.

22–33. То же, что в тестах 10–21, только $n = 50$, $m = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17$.

34–41. Случайные тесты, НОК входит в тип int. Есть один максимальный тест $n = 10^5$.

42–46. Случайные тесты, НОК не входит в тип int. Есть один максимальный тест $n = 10^5$.

47–50. Случайные тесты, НОК не входит в тип int64. Есть один максимальный тест $n = 10^5$.