

Памятка участникам заключительного тура

1. Заключительный тур проводится очно на заранее согласованных площадках проведения. Каждому участнику во время проведения заключительного тура предоставляется персональный компьютер с установленным программным обеспечением (ОС, компилятор, среда разработки) согласно предварительным заявкам. Сеть Интернет недоступна. Использование любых справочных материалов, за исключением установленных на предоставленном компьютере, а также средств связи запрещено.
2. Заключительный тур проводится в тестирующей системе Ejudge, развернутой на сервере факультета компьютерных наук – http://www.cs.vsu.ru/ejudge/cgi-bin/new-register?action=209&contest_id=10&locale_id=1 Несмотря на то, что в данный момент на сервере настроен упрощенный режим регистрации без подтверждения по электронной почте (режим регистрации может быть изменен), настоятельно рекомендуется при регистрации указывать адрес реально существующего почтового ящика участника (завести почтовый ящик в интернете совсем несложно). И обязательно запомните ваш пароль (а лучше запишите)!
3. В рамках заключительного тура участникам за ограниченное время (4 часа) предлагается решить 6 задач разного уровня сложности. Решением каждой задачи является программа на одном из языков программирования, поддерживаемых тестирующей системой (C++, Pascal, C#, Java, Python), которую участник должен отправить на тестирование (посылка).
4. **В заключительном туре засчитываются только абсолютно полные решения задач, т.е. такие решения, которые проходят все тесты** (если задача не пройдет хотя бы один тест, то решение задачи не засчитывается). Ранжирование участников осуществляется по количеству решенных задач, если же кол-во решенных задач одинаковое, то учитывается суммарное время решения задач (для решенных задач – сумма времени, прошедшего от начала олимпиады до правильного решения каждой задачи) – чем оно меньше, тем лучше. **За каждую неверную посылку к суммарному времени добавляется штраф в 10 минут (так называемые АСМ-правила проведения турнира).**
5. Кол-во тестов для каждой задачи – 50 (для отдельных задач может отличаться), тесты составлены так, чтобы учитывались всевозможные, в том числе граничные, случаи в решениях (как правило, часть тестов к каждой задаче составлена случайным образом).
6. Общее количество всех посылок по всем задачам ограничено числом 100.
7. Для организации ввода-вывода в программе можно использовать как стандартный ввод-вывод, так и работу с файлами `input.txt / output.txt`. Решение всех задач должно соответствовать ограничению по времени в 1 с и по памяти – в 64 Mb (для отдельных задач могут быть указаны индивидуальные ограничения по времени и памяти).
8. Участникам олимпиады во время проведения олимпиады доступны полные протоколы тестирования отправленных решений. В протоколе тестирования отражены входные данные нескольких первых тестов для каждой задачи (возможно, что и всех тестов, зависит от размера тестов), но правильный ответ не показан (только вердикт проверки решения также без указания правильного ответа). Этой информацией можно пользоваться для отладки своих решений.
9. Решения, в которых вместо честного решения будет подбираться / забиваться ответ под известные тесты, если такие будут обнаружены, не учитываются. Жюри также оставляет за собой право перетестировать все решения всех участников для любой задачи заключительного тура на другом наборе тестов и при подведении итогов учитывать новые результаты.

10. Жюри олимпиады оставляет за собой право проводить проверку работ на списывание, после которой вносить изменения в результаты участников олимпиады, полученные при автоматическом тестировании решений.
11. В процессе проведения олимпиады участники могут обращаться с вопросами к жюри посредством отправки сообщений в тестирующей системе. Жюри может не отвечать на вопросы участников, если сочтет, что вопрос задан некорректно или же ответ будет содержать подсказку участнику.
12. В условиях задач, приведенных ниже, присутствует Задача 0. Длина окружности. Решение данной задачи не учитывается в итоговом результате заключительного тура, данная задача приведена исключительно для ознакомления с тестирующей системой и правилами оформления решений участниками олимпиады.

Желаем вам успеха!

Задача 0. Длина окружности

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1

Примечание

Данная задача является учебной для ознакомления с правилами оформления задач и системой тестирования. Решение данной задачи не учитывается.

Условие задачи

Вам необходимо посчитать длину окружности радиуса R .

Входные данные

Во входного файла записано единственное целое число R – радиус окружности, длину которой вам необходимо посчитать.

Выходные данные

В выходной файл требуется вывести одно вещественное число L – длину окружности радиуса R .

Абсолютная погрешность ответа не должна превышать 10^{-6} .

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)
1	6.283185

Пример решения

C++ (стандартный ввод/вывод)	Pascal (стандартный ввод/вывод)
<pre>#define _USE_MATH_DEFINES #include <fstream> #include <ios> #include <iomanip> #include <cmath> using namespace std; int main() { int r; cin >> r; double l = 2 * M_PI * r; cout << fixed << setprecision(6); cout << l << endl; // другой способ: // char buf[50]; // sprintf(buf, "%.6f", l); // cout << buf << endl; }</pre>	<pre>const PI = 3.14159265359; var R: Integer; L: Double; begin ReadLn(R); L := 2 * R * PI; WriteLn(L:20:6); // никаких ReadLn в конце // быть не должно end.</pre>

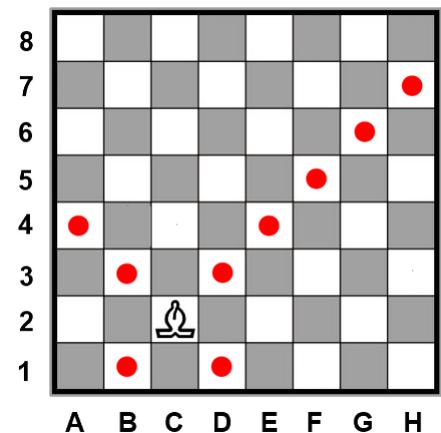
Задача 1. Ход слоном

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

Чтобы защитить свой блог от автоматического спама, Вовочка решил использовать оригинальную капчу (CAPTCHA – англ. Completely Automated Public Turing test to tell Computers and Humans Apart – полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей). Как большой любитель шахмат Вовочка в качестве теста придумал просить людей решить простую задачу, а именно ответить на вопрос, сколько различных ходов можно сделать шахматной фигурой, находящейся на пустой доске в определенной позиции. Для проверки ответа требуется составить программу, сегодня Вовочка реализует алгоритм проверки ответа для Слона и просит ему помочь.

Как известно, Слон – шахматная фигура, которая, не имея препятствий, может перемещаться на любую клетку, находящуюся на любой из двух диагоналей. Позиция фигуры (клетка) в шахматах задается в виде пары – латинской буквы от А до Н и цифры от 1 до 8. На рисунке справа показан пример доски, поясняющий правила обозначения клеток. На данном рисунке единственная фигура Слон находится в клетке С2. Точками отмечены клетки, на которые Слон может пойти с позиции С2.



Входные данные

В единственной строке входного файла в шахматной нотации (буква латинского алфавита от А до Н и цифра от 1 до 8) указана клетка, на которой находится Слон. Буквы латинского алфавита – заглавные.

Выходные данные

В выходной файл необходимо записать единственное число – количество различных клеток, на которые может за один ход переместиться Слон, находящийся в заданной позиции на пустой шахматной доске.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
C2	9	Пример соответствует рисунку в условии задачи.
D6	11	

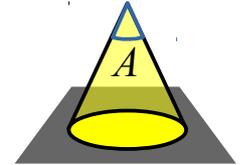
Задача 2. Освещение коридора

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

В большой новой квартире, в которую недавно переехал Вовочка, имеется длинный прямой коридор, соединяющий все помещений квартиры. Вовочка решил сделать в квартире небольшой ремонт и, в частности, натянуть натяжной потолок и переделать освещение в коридоре.

Для организации освещения Вовочка собирается использовать встраиваемые потолочные энергосберегающие светильники, обеспечивающие направленное освещение с углом рассеивания A . Для простоты будем считать, что это означает, что свет равномерно расходится от светильника в виде конуса с углом раствора (углом между образующими конуса) A . Опять же для простоты примем, что за пределы угла A свет не выходит.



Для красоты Вовочка хочет все светильники разместить по центру коридора. Другими словами все светильники должны располагаться на воображаемой линии, проходящей по потолку на равноудаленном расстоянии от длинных сторон коридора. Светильники должны быть направлены строго вертикальной вниз (по перпендикуляру к полу). При этом Вовочка считает, что необходимо разместить светильники так, чтобы пол коридора был полностью освещен прямым светом (в противном случае щенок Вовочки Ньютон будет прятать в темных углах тапки, которые нужно будет искать).

Вовочка хочет знать, какое минимальное количество светильников ему необходимо приобрести для освещения коридора согласно приведенным выше требованиям?

Входные данные

В единственной строке входного файла через пробел записаны 4-ре целых положительных числа A , L , W , H , где A – угол рассеивания светильников, заданный в градусах ($10 \leq A \leq 150$), а L , W и H – длина, ширина и высота (высота потолка) коридора соответственно, заданные в одинаковых единицах измерения ($1 \leq L, W, H \leq 10^5$, $W \leq L$). Напомним, что коридор имеет строгую перпендикулярную форму.

Выходные данные

В единственной строке выходного файла надо записать одно число N_{min} – минимальное количество светильников, которые потребуются Вовочке для освещения коридора с соблюдением всех озвученных требований. Если это невозможно, то вместо N_{min} требуется записать число 0.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
60 1000 200 300	4	
30 50 20 30	0	Каждый потолочный светильник будет давать на полу круглое пятно света диаметром менее 17 единиц. Рядом с длинными стенами коридора будет оставаться темное пространство (ширина коридора – 20).

Задача 3. Красивые букеты

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

В день 8-го Марта Вовочка решил поздравить как можно большее количество представительниц прекрасного пола (маму, сестру, одноклассниц и, конечно же, учителей). Но какое же поздравление в такой день без букета цветов! Поэтому накануне Вовочка пошел в ближайший цветочный магазин и скупил там все имеющиеся тюльпаны. Тюльпаны были разных цветов и теперь перед Вовочкой стоит задача сформировать *красивые* букеты. Вовочка считает, что *красивые* букеты состоят из нечетного количества, но не менее 3-х, тюльпанов различных цветов (т.е. цвета тюльпанов в букете не должны повторяться).

Помогите Вовочке сформировать наибольшее количество красивых букетов из имеющихся у него тюльпанов.

Входные данные

В первой строке входного файла записано целое число N – кол-во различных цветов купленных Вовочкой тюльпанов ($0 \leq N \leq 100$). Во второй строке через пробел записано N целых чисел C_1, C_2, \dots, C_N – количество тюльпанов каждого цвета ($1 \leq C_i \leq 100, 1 \leq i \leq N$).

Выходные данные

В первой строке входного файла надо вывести число M – максимальное кол-во красивых букетов, которые может составить Вовочка. Далее в M строках необходимо описать красивые букеты в виде набора чисел $F_{j1}, F_{j2}, \dots, F_{jk}$ – номеров цветов тюльпанов, составляющих j -ый букет ($1 \leq F_{jk} \leq N, 1 \leq j \leq M, 1 \leq k \leq K, 3 \leq K, K$ – нечетное).

В случае существования различных вариантов составления максимального кол-ва красивых букетов необходимо вывести любой из них.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
7 1 3 10 1 4 1 2	6 2 3 5 3 5 7 3 6 7 1 2 3 2 3 5 3 4 5	При этом у Вовочки еще останутся 4 тюльпана одного цвета – цвета 3, так как было опрометчиво куплено целых 10 тюльпанов данного цвета.
5 1 1 1 1 1	1 4 5 2	Любые 3 тюльпана образуют красивый букет.
5 2 3 2 2 2	3 1 2 3 3 4 5 1 2 5 4 2	По условию задачи букет из 5-ти различных тюльпанов также красивый :)
1 7	0	Все тюльпаны одного цвета, из них составить красивые букеты не получится.

Задача 4. Сладкая мечта

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

Вовочка – большой любитель шоколада. Он хотел бы кушать шоколад каждый день, однако плитки шоколада дарят Вовочке не так часто, а тратить свои деньги на сладости Вовочка не хочет, так как копит средства на новый компьютер.

Вовочка сильно расстраивается, если в какой-то день у него не окажется шоколада, если в предыдущий день он ел шоколад. Вовочка твердо решил сильно не расстраиваться на каникулах. Поэтому он будет откладывать подаренные шоколадки "на будущее", чтобы с какого-то дня и до конца каникул позволить себе каждый день съесть по одной плитке шоколада.

Вовочка составил список, когда и сколько шоколадок он ожидает в подарок. Теперь он просит Вас помочь определить самый ранний день, когда он, согласно составленному списку, сможет наконец осуществить свою мечту и кушать по одной плитке шоколада каждый день до конца каникул.

Плитку шоколада можно съесть уже в день подарка. Также известно, что в последний день каникул Вовочка обязательно получит шоколад в подарок.

Входные данные

В первой строке входного файла записано целое число N ($1 \leq N \leq 10^5$) – количество дней, когда Вовочке будут подарены шоколадки.

На каждой из последующих N строк записаны через пробел по два целых числа T_i и C_i ($1 \leq C_i, T_i \leq 10^9$) – номера дней и количество плиток шоколада, который подарят Вовочке в этот день. Все дни T_i различны и идут в порядке возрастания.

Выходные данные

В выходной файл требуется записать единственное число T_{min} ($1 \leq T_{min} \leq 10^9$) – минимальный день, с которого Вовочка может начать открывать по одной в день накопленные и поступающие шоколадки, не пропуская ни одного дня до конца каникул.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
3 2 3 3 1 7 4	3	Если начинать есть шоколад по одной плитке со 2-го дня, то на 6-ой день у Вовочки не останется ни одной шоколадки.
4 1 1 2 1 3 1 4 1	1	Вовочка каждый день получает в подарок одну плитку шоколада, которую можно тут же открывать и съесть.

Задача 5. Номера домов

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

Последние несколько лет улица, на которой живет Вовочка, массово, но хаотично застраивалась новыми домами. В итоге поиск по адресу номера дома превратился в сущий кошмар. Поэтому мэрия решила навести порядок и переназначить номера некоторым домам.

Требования к новой нумерации домов следующие:

- Номера домов – целые положительные числа $\leq 2 * 10^9$, не повторяются.
- На одной стороне улицы должны находиться четные номера домов, на другой – нечетные. Четной можно сделать произвольную сторону улицы.
- При движении вдоль улицы номера домов должны либо монотонно возрастать с обеих сторон, либо монотонно убывать (но не обязательно идти последовательно).

Перенумерация домов доставит массу неудобств их жителям (переоформление документов и т.д.), поэтому мэрия в настоящий момент разрабатывает план новой нумерации, которая будет удовлетворять озвученным условиям, но затронет по возможности наименьшее кол-во домов.

Вовочка хочет предложить мэрии свой проект новой нумерации домов. Однако, чтобы оценить оптимальность своего проекта, ему необходимо знать, какое количество домов необходимо перенумеровать в самом оптимальном с точки зрения минимизации количества перенумеруемых домов проекте. Помогите Вовочке найти это минимальное количество.

Входные данные

В первой строке входного файла через пробел записаны два числа N и M ($1 \leq N, M \leq 3 * 10^3$).

Во второй строке, разделенные пробелами, следуют N различных чисел A_i ($1 \leq A_i \leq 10^9$), представляющие последовательность номеров домов по одной из сторон при движении вдоль по улице.

В третьей строке расположены M различных чисел B_j ($1 \leq B_j \leq 10^9$), представляющие последовательность номеров домов по другой стороне улицы, если двигаться в том же направлении, при котором были получены номера A_i .

Выходные данные

В единственной строке выходного файла надо записать число K – минимальное количество домов, которое необходимо перенумеровать, чтобы удовлетворить требований к новой нумерации.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
4 4 5 12 3 6 9 7 4 1	3	Возможный вариант нумерации: 18 12 8 6 9 7 3 1
5 4 1 3 7 9 15 2 6 8 10	0	Нумерация домов уже удовлетворяет требованиям.

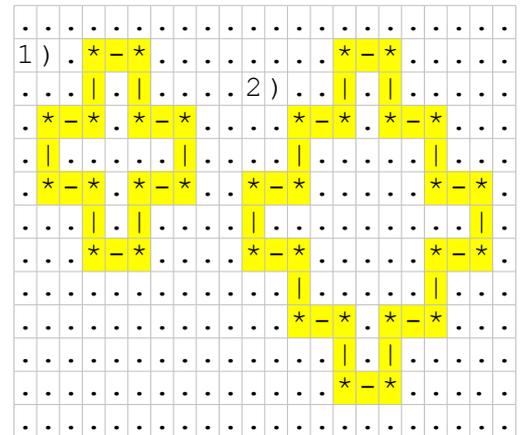
Задача 6. Снежинки

Входной файл	стандартный ввод / input.txt
Выходной файл	стандартный вывод / output.txt
Ограничение времени (сек/тест)	1
Ограничение памяти (Мб)	64

Условие задачи

На новогодних каникулах Вовочка, запустив непроверенную программу, скаченную из Интернета, заразил свой компьютер вирусом «Дед Мороз». С вирусом он справился, однако остались последствия. «Дед Мороз» оказался большим шутником и зашифровал весь исходный код проектов, с которыми Вовочка работал. Зашифрованные файлы остались текстовыми, но их содержимое на первый взгляд превратилось в случайный набор символов.

Кроме того, Вовочка заметил, что в этих файлах часто повторяется шаблон, похожий на нарисованную псевдографикой снежинку. «Снежинки» встречаются разного размера. Граница «снежинки» формируются тремя символами: '|' – вертикальная черта, '-' – знак «минус», '*' – «звездочка», тело «снежинки» состоит из символов '.' – точка (в зашифрованных файлах вообще все пробелы заменены точками). На рисунке справа показаны, как выглядят две «снежинки» – размера 1 и размера 2. (формально размер «снежинки» можно определить как $M/4$, где M – кол-во «звездочек» во внутренних углах фигуры). «Снежинки» большего размера формируются аналогично. Для наглядности на рисунке в виде сетки обозначены границы символов, т.е. каждая клеточка – один символ. Выделенные цветом клеточки – символы, образующие границу «снежинки».



В интернете Вовочка прочитал, что зашифрованные вирусом «Дед Мороз» файлы можно расшифровать по специальному алгоритму, ключом к которому являются положение и размер всех «снежинок», встречающихся в файле. Помогите Вовочке найти этот ключ.

Входные данные

Входной файл состоит из произвольного количества строк, но не более 100. Строки могут содержать буквы латинского алфавита, цифры, любые знаки препинания и спецсимволы (а более конкретно символы из набора "abcdefghijklmnopqrstuvwxyzAB CDEFGHIJKLMNOPQRSTUVWXYZ1234567890.%!@#\$%^&*(){}[].,;:"). Длина каждой строки не превышает 100 символов, строки в одном файле могут быть разной длины, но не пустые. Последняя строка файла содержит единственный символ – точку.

Выходные данные

В первой строке входного файла нужно записать единственное число N – количество найденных «снежинок».

В последующих N строках надо записать информацию о каждой найденной «снежинке» в виде разделенных пробелами трех чисел R_i, C_i, S_i , где R_i, C_i – позиция центра «снежинки» (R_i – номер строки, C_i – номер символа в строке, нумеруются с 1-цы, см. пример входных-выходных данных), а S_i – размер i -той «снежинки» (см. определение выше).

Две «снежинки» могут иметь общую границу (часть границы). Также напомним, что если внутри тела «снежинки» встречаются любые символы, кроме точек, то такая фигура «снежинкой» не является. Также не существует «снежинок» размера 0.

Список обнаруженных «снежинок» можно выводить в произвольном порядке.

Пример входного файла (stdin / input.txt)	Пример выходного файла (stdout / output.txt)	Комментарий
<pre> 1) *-*.....*-*.....2)... . . *-*.*-*.....*-*.*-* *-*.*-*.....*-* *-*.....*-**-*..... </pre>	<pre> 2 5 5 1 7 16 2 </pre>	<p>Пример соответствует рисунку в условии задачи.</p>
<pre> ...*-*...*-*..... .av . .web . .eihdik.. *-*.*-*.*-*.....we *-*.*-*.....*-*2.*-*.*-*.....*-*22 dwe*-*.*-*.*-*.*-*.. *-*.....*-*.*-* 337greh% -*.....*-*.*-*...34rr3 -*.....*-* wed *-*.....*-*.....we%1() 5..23wed....^781 278*-*.*-*dghw23*-* weghd*-*.....45...23..67.....*-*.*-* bvgh.%^.we.....3.*-*..not.*-* HB@bshk4\$.snowflake ..wellkwef.34.*-*..!..*-* 3457856.....5345.45.65.....*-*.*-* 5.45.345.....34587.9879.....*-* 42re.hegjhb werkjkwernkijn3urn3untiunu3 # # end. . </pre>	<pre> 3 4 5 1 10 15 1 6 11 2 </pre>	<p>«Снежинка» размером 3 с центром (14, 7) своим левым краем чуть-чуть «вылезает» за границу файла, поэтому формально снежинкой не считается.</p> <p>Фигура с центром (25, 20) в нижнем правом углу файла также не является «снежинкой» так содержит внутри себя символы, отличные от точек.</p> <p>Таким образом остаются три «правильных» «снежинки».</p> <p>При этом большая из 3-х «снежинок» граничит (имеет общую границы) с двумя меньшими.</p>
<pre> class.NewClass.{ } . </pre>	<pre> 0 </pre>	<p>«Снежинок» нет.</p>