Министерство просвещения Российской Федерации
Министерство образования Иркутской области
Департамент образования города Иркутска
Муниципальное автономное общеобразовательное учреждение Лицей ИГУ города Иркутска
МАОУ Лицей ИГУ г. Иркутска

РАССМОТРЕНО на заседании методического объединения учителей информатики и технологии от 29.08.2023г. протокол №1. Руководитель МО Л.Н. Шеметова	УТВЕРЖДЕНО Приказ № 01-06-140 от 30.08.2023 г. Директор Е.Ю. Кузьмина
ПРИНЯТО решением педагогического совета от 30.08.2023 г., протокол №1	**ID -**

РАБОЧАЯ ПРОГРАММА

ОСНОВНОГО ОБЩЕГО ОБРАЗОВАНИЯ

ID ---
учебного курса

«РЕШЕНИЕ ОЛИМПИАДНЫХ ЗАДАЧ ПО ИНФОРМАТИКЕ»

(для 10-11 классов)

Срок освоения – 2 года

Уровень сложности программы УГЛУБЛЕННЫЙ

Количество часов по программе за весь период реализации - 136

Разработчик: Зубков О.В., учитель информатики,
высшая кв.категория
Шеметова Л.Н., учитель информатики,
высшая кв.категория

г. Иркутск, 2023 г.

АННОТАЦИЯ К РАБОЧЕЙ ПРОГРАММЕ
учебного курса «Решение олимпиадных задач по информатике»

для обучающихся 10-11 классов

Рабочая программа спецкурса «Решение олимпиадных задач по информатике» (10-11 классы) разработана в соответствии с требованиями ФГОС и ФОП основного общего образования и Положением «О рабочих программах учебных предметов, учебных курсов (в том числе внеурочной деятельности), учебных модулей в соответствии с требованиям ФГОС и ФОП основного общего образования» МАОУ Лицея ИГУ г.Иркутска, утвержденного приказом директора 01-06-130 от 30.08.2023 года и является частью основной образовательной программы основного общего образования.

Рабочая программа ориентирована на целевые приоритеты, сформулированные в федеральной рабочей программе воспитания и в рабочей программе воспитания МАОУ Лицей ИГУ г. Иркутска.

Обучение информатике направлено на достижение следующих задач:

- закрепить навыки программирования, полученные при изучении курса «Информатика и ИКТ»;
- ознакомиться с основными разделами и темами олимпиадных задач по информатике.
- рассмотреть основные подходы к и реализации известных алгоритмов.
- изучить новые, более эффективные по времени работы алгоритмы.
- узнать специфику и особенности проведения олимпиад по информатике высокого уровня.
- научиться писать программы грамотно, быстро и правильно, уметь их отлаживать и тестировать.
- получить новые сведения об объектах и процессах в смежных областях знаний, моделируемых в рассматриваемых задачах.

Содержание обучения информатике направлено на реализацию следующих целей:

- подготовка детей к современной жизни,
- развитие алгоритмического мышления, способностей к формализации реально протекающих процессов;
- освоение учащимися основных приемов технологии программирования;
- воспитание культуры решения прикладных задач с использованием программных средств и алгоритмических языков высокого уровня;
- развитие нестандартного мышления для решения прикладных и специальных задач повышенной сложности;
- формирование умения понять проблему, оценить ее сложность и наметить пути ее решения;
- развитие навыков работы в команде;
- развитие настойчивости и упорства при достижении поставленной цели и корректировки методов ее достижения.

Рабочая программа учебного курса «Решение олимпиадных задач по информатике» входит в УЗ ОРИПО предметной области «Математика и информатика»

Срок реализации программы – 2 год (10-11 классы)

Количество учебный часов, на которые рассчитана программа

	10 класс	11 класс	Всего
Кол-во учебных недель	34	34	
Кол-во часов в неделю	2	2	
Кол-во часов в год	68	68	136

В программу включены содержание, планируемые результаты (личностные, метапредметные, предметные), тематическое планирование с учетом рабочей программы воспитания, оценочные и методические материалы.

Рабочая программа обсуждена и принята решением методического объединения учителей-предметников (протокол №1 от 29.08.2023 г.), согласована с заместителем директора МАОУ Лицей ИГУ г. Иркутска, утверждена приказом директора № 01-06-130 от 30.08.2023 г.

Пояснительная записка

Актуальность программы.

Основной задачей образования является выявление и работа с одаренными учащимися по подготовке к предметным олимпиадам. Участие в олимпиадах позволяет развивать творческие способности школьников и обеспечивает высокую мотивацию к образовательной деятельности. Олимпиада по информатике – это олимпиада по программированию, которая предполагает наличие обширных познаний в математике и языках программирования.

Курса "Решение олимпиадных задач по информатике" разработан для возрастной категории 10, 11 классов и дополняет своим содержанием и методами основной курс информатики. В нем рассматриваются и осваиваются сведения из областей информатики и программирования, важные в общеобразовательном и прикладном отношении; формируются навыки и приемы решения задач с использованием готовых и собственноручно разработанных программных средств; развиваются алгоритмическое мышление, информационная культура вообще, культура программирования, в частности; вырабатывается компетентность в использовании компьютерных технологий.

Данный учебный курс рекомендован для использования в 10-11-х специализированных (профильных) классах с углубленным изучением информатики и программирования. Особое внимание уделяется алгоритмизации и программированию задач творческого типа. Предлагаемый курс предназначен для подготовки интересующихся данной тематикой учеников, предварительно уже имеющих начальные навыки работы в одном из языков программирования и желающих писать грамотные, эффективные и правильно скомпонованные программы. В курсе рассматриваются разделы и разбираются задачи самого различного уровня сложности, впрочем, метод “от простого к сложному” является основополагающим. Если сравнивать сложность рассматриваемых задач, то они охватывают достаточно широкий круг от сложных задач школьного и муниципального (городского) уровня, все типы и сложности регионального уровня и заканчивая сравнительно простыми по сложности темами всероссийского уровня олимпиад.

Таким образом, программа спецкурса “Решение олимпиадных задач по информатике” ориентирована как на подготовку к школьным олимпиадам по информатике и программированию самого различного уровня, так и к сдаче итоговой аттестации в формате ЕГЭ и составлена с учетом профессиональной ориентации учащихся.

Задача курса:

1. закрепить навыки программирования, полученные при изучении курса «Информатика и ИКТ»;
2. ознакомиться с основными разделами и темами олимпиадных задач по информатике.
3. рассмотреть основные подходы к и реализации известных алгоритмов.
4. изучить новые, более эффективные по времени работы алгоритмы.
5. узнать специфику и особенности проведения олимпиад по информатике высокого уровня.
6. научиться писать программы грамотно, быстро и правильно, уметь их отлаживать и тестировать.
7. получить новые сведения об объектах и процессах в смежных областях знаний, моделируемых в рассматриваемых задачах.

Цели курса:

- подготовка детей к современной жизни,
- развитие алгоритмического мышления, способностей к формализации реально протекающих процессов;
- освоение учащимися основных приемов технологии программирования;
- воспитание культуры решения прикладных задач с использованием программных средств и алгоритмических языков высокого уровня;
- развитие нестандартного мышления для решения прикладных и специальных задач повышенной сложности;
- формирование умения понять проблему, оценить ее сложность и наметить пути ее решения;
- развитие навыков работы в команде;
- развитие настойчивости и упорства при достижении поставленной цели и корректировки методов ее достижения.

Планируемые результаты освоения спецкурса

Личностные результаты

- формирование целостного мировоззрения, соответствующего современному уровню развития науки и общественной практики, учитывающего социальное, культурное, языковое, духовное многообразие современного мира.
- формирование собственного понимания значимость подготовки в области программирования в условиях развития информационного общества;
- формирование готовности к повышению своего образовательного уровня и продолжению обучения с использованием средств и методов информатики и программирования;
- формирование способности и готовности к общению и сотрудничеству со сверстниками и взрослыми в процессе образовательной, учебно-исследовательской, творческой деятельности.

Метапредметные образовательные результаты:

- умение решать задачи из разных сфер человеческой деятельности с применением методов программирования и средств языка программирования.
- владение информационно-логическими умениями: создавать обобщения, устанавливать аналогии, классифицировать, строить логическое рассуждение, умозаключение (индуктивное, дедуктивное и по аналогии) и делать выводы;

- владение основными универсальными умениями информационного характера, такими как: постановка и формулирование задачи; выбор наиболее эффективных способов решения задач в зависимости от конкретных условий; самостоятельное создание алгоритмов деятельности при решении задач программирования.

Предметные образовательные результаты:

- формирование представления об основных правилах программирования, тестирования и отладки;
- формирование представления о классификации и свойства базовых алгоритмических конструкций;
- формирование знаний о некоторых сложных алгоритмах для эффективного решения олимпиадных задач по информатике;
- формирование знаний о структуре и возможностях сред программирования и основных особенностях работы с ними;
- формирование знаний об основных приемах программирования на языке программирования C++;
- формирование представления об особенностях проведения современных соревнований по информатике и программированию;
- формирование представления об особенностях работы в автоматических проверяющих системах.
- формирование представления, как проводить предварительный анализ задачи;
- формирование представления, как разрабатывать алгоритмическую модель решения задачи;
- формирование представления, как реализовывать разработанный алгоритм на языке программирования C++;
- формирование представления, как проводить отладку программы с помощью и без помощи среды программирования;
- формирование представления, как сознательно выбирать структуры данных, оптимальные для решения задач;
- формирование представления, как пользоваться возможностями операционной системы, файловых менеджеров, текстовых редакторов, трансляторов, сред программирования для решения олимпиадных задач по информатике;
- формирование представления, как анализировать и интерпретировать полученные результаты;
- формирование представления, как успешно взаимодействовать с современными автоматическими системами проведения турниров и проверки сдаваемых программ.

Содержание программы

(136 часов)

Встроенные типы данных языка C++

Работа с файлами. Написание программ, производящих ввод и вывод через файлы.

Особенности проведения современных олимпиад по информатике и программированию. Современные автоматические системы проведения турниров.

Ресурсы сети Интернет для самостоятельной подготовки к олимпиадам по информатике.

Задачи для начинающих. Операторы цикла в С++. Основные конструкции в языке С++, массивы и векторы. Процедуры и функции в языке С++.

Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида. Решето Эратосфена. Понятие сложности и эффективности алгоритма. Задачи целочисленной арифметики. Быстрое возведение в степень. Быстрое вычисление чисел Фибоначчи.

Эффективные алгоритмы. Дискретный бинарный поиск. Бинарный поиск ответа на множестве вещественных чисел.

Контейнер set в С++, динамическое упорядочение.

Контейнер map в С++, индексация сложными объектами.

Сортировка объектов. Квадратичная и быстрая сортировки. Встроенная сортировка в С++, изменение методов сравнения и сортировки.

Длинная арифметика

Разбор строк. Обработка текста. Разбор строк. Решение символьных уравнений.

Жадные алгоритмы.

Геометрия на плоскости. Простейшие фигуры. Геометрия на плоскости. Прямые. Скалярное и векторное произведения векторов. Площадь произвольных (в том числе и невыпуклых) многоугольников. Формула Пика. Работа с углами.

Рекурсия, перебор.

Математическое моделирование. .

Двумерные массивы и векторы в С++

Теория графов. Определение и методы задания графов. Обходы графа.

Поиск путей и циклов.

Содержание программы
(68 часов)

Встроенные типы данных языка С++.

Работа с файлами. Написание программ, производящих ввод и вывод через файлы.

Особенности проведения современных олимпиад по информатике и программированию. Современные автоматические системы проведения турниров. Ресурсы сети Интернет для самостоятельной подготовки к олимпиадам по информатике

Задачи для начинающих. Операторы цикла в С++. Основные конструкции в языке С++, массивы и векторы. Процедуры и функции в языке С++

Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида. Решето Эратосфена. Быстрое возведение в степень. Быстрое вычисление чисел Фибоначчи.

Понятие сложности и эффективности алгоритма

Эффективные алгоритмы. Дискретный бинарный поиск. Бинарный поиск ответа на множестве вещественных чисел

Контейнер set в С++, динамическое упорядочение

Контейнер map в С++, индексация сложными объектами

ТЕМАТИЧЕСКОЕ, ПОУРОЧНОЕ ПЛАНИРОВАНИЕ
(68 часов).

№ п/п	Наименование разделов и тем программы	Количество часов			Электронные (цифровые) образовательные ресурсы	Формы контроля
		Всего	Контрольные работы	Практические работы		
10 класс						
1.	Встроенные типы данных языка С++.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
2.	Встроенные типы данных языка С++.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
3.	Работа с файлами. Написание программ, производящих ввод и вывод через файлы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
4.	Работа с файлами. Написание программ, производящих ввод и вывод через файлы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
5.	Работа с файлами. Написание программ, производящих ввод и вывод через файлы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
6.	Работа с файлами. Написание программ, производящих ввод и вывод через файлы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
7.	Особенности проведения современных олимпиад по	1		1	1.http://acmp.ru/	Текущий

	информатике и программированию				2. http://codeforces.ru	
8.	Особенности проведения современных олимпиад по информатике и программированию	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
9.	Особенности проведения современных олимпиад по информатике и программированию	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
10.	Особенности проведения современных олимпиад по информатике и программированию	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
11.	Особенности проведения современных олимпиад по информатике и программированию	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
12.	Особенности проведения современных олимпиад по информатике и программированию	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
13.	Современные автоматические системы проведения турниров.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
14.	Современные автоматические системы проведения турниров.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
15.	Ресурсы сети Интернет для самостоятельной подготовки к	1		1	1.http://acmp.ru/	Текущий

	олимпиадам по информатике				2. http://codeforces.ru	
16.	Ресурсы сети Интернет для самостоятельной подготовки к олимпиадам по информатике	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
17.	Задачи для начинающих	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
18.	Задачи для начинающих	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
19.	Задачи для начинающих	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
20.	Задачи для начинающих	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
21.	Операторы цикла в C++	1.		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
22.	Операторы цикла в C++	1.		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
23.	Операторы цикла в C++	1.		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
24.	Операторы цикла в C++	1.		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
25.	Основные конструкции в языке C++, массивы и векторы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
26.	Основные конструкции в языке C++, массивы и векторы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

27.	Основные конструкции в языке С++, массивы и векторы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
28.	Основные конструкции в языке С++, массивы и векторы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
29.	Процедуры и функции в языке С++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
30.	Процедуры и функции в языке С++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
31.	Процедуры и функции в языке С++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
32.	Процедуры и функции в языке С++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
33.	Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
34.	Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
11 класс						
35.	Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
36.	Задачи целочисленной арифметики. Нахождение НОД, алгоритм Евклида.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

37.	Задачи целочисленной арифметики. Решето Эратосфена.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
38.	Задачи целочисленной арифметики. Решето Эратосфена.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
39.	Задачи целочисленной арифметики. Решето Эратосфена.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
40.	Задачи целочисленной арифметики. Решето Эратосфена.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
41.	Понятие сложности и эффективности алгоритма	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
42.	Понятие сложности и эффективности алгоритма	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
43.	Понятие сложности и эффективности алгоритма	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
44.	Понятие сложности и эффективности алгоритма	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
45.	Задачи целочисленной арифметики. Быстрое возведение в степень.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
46.	Задачи целочисленной арифметики. Быстрое возведение в степень.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
47.	Задачи целочисленной арифметики. Быстрое возведение в степень.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

48.	Задачи целочисленной арифметики. Быстрое возведение в степень.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
49.	Задачи целочисленной арифметики. Быстрое вычисление чисел Фибоначчи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
50.	Задачи целочисленной арифметики. Быстрое вычисление чисел Фибоначчи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
51.	Задачи целочисленной арифметики. Быстрое вычисление чисел Фибоначчи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
52.	Задачи целочисленной арифметики. Быстрое вычисление чисел Фибоначчи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
53.	Эффективные алгоритмы. Дискретный бинарный поиск.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
54.	Эффективные алгоритмы. Дискретный бинарный поиск.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
55.	Эффективные алгоритмы. Дискретный бинарный поиск.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
56.	Эффективные алгоритмы. Дискретный бинарный поиск.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
57.	Эффективные алгоритмы. Бинарный поиск ответа на множестве вещественных чисел	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

58.	Эффективные алгоритмы. Бинарный поиск ответа на множестве вещественных чисел	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
59.	Эффективные алгоритмы. Бинарный поиск ответа на множестве вещественных чисел	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
60.	Эффективные алгоритмы. Бинарный поиск ответа на множестве вещественных чисел	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
61.	Контейнер set в C++, динамическое упорядочение	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
62.	Контейнер set в C++, динамическое упорядочение	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
63.	Контейнер set в C++, динамическое упорядочение	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
64.	Контейнер set в C++, динамическое упорядочение	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
65.	Контейнер map в C++, индексация сложными объектами	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
66.	Контейнер map в C++, индексация сложными объектами	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
67.	Контейнер map в C++, индексация сложными объектами	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
68.	Контейнер map в C++, индексация сложными объектами	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

ТЕМАТИЧЕСКОЕ ПЛАНИРОВАНИЕ

(136 часов).

№ п/п	Наименование разделов и тем программы	Количество часов			Электронные (цифровые) образовательные ресурсы	Формы контроля
		Всего	Контрольные работы	Практические работы		
10 класс						
1.	Сортировка объектов. Квадратичная и быстрая сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
2.	Сортировка объектов. Квадратичная и быстрая сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
3.	Сортировка объектов. Квадратичная и быстрая сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
4.	Сортировка объектов. Квадратичная и быстрая сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
5.	Всроенная сортировка в C++, изменение методов сравнения и сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
6.	Всроенная сортировка в C++, изменение методов сравнения и сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

7.	Всроенная сортировка в С++, изменение методов сравнения и сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
8.	Всроенная сортировка в С++, изменение методов сравнения и сортировки.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
9.	Длинная арифметика	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
10.	Длинная арифметика	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
11.	Длинная арифметика	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
12.	Длинная арифметика	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
13.	Разбор строк. Обработка текста	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
14.	Разбор строк. Обработка текста	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
15.	Разбор строк. Обработка текста	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
16.	Разбор строк. Обработка текста	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
17.	Разбор строк. Решение символьных уравнений.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

18.	Разбор строк. Решение символьных уравнений.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
19.	Разбор строк. Решение символьных уравнений.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
20.	Разбор строк. Решение символьных уравнений.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
21.	Жадные алгоритмы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
22.	Жадные алгоритмы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
23.	Жадные алгоритмы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
24.	Жадные алгоритмы.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
25.	Геометрия на плоскости. Простейшие фигуры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
26.	Геометрия на плоскости. Простейшие фигуры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
27.	Геометрия на плоскости. Простейшие фигуры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
28.	Геометрия на плоскости. Простейшие фигуры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
29.	Геометрия на плоскости. Прямые.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

30.	Геометрия на плоскости. Прямые.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
31.	Геометрия на плоскости. Прямые.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
32.	Геометрия на плоскости. Прямые.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
33.	Геометрия на плоскости. Скалярное и векторное произведения векторов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
34.	Геометрия на плоскости. Скалярное и векторное произведения векторов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
35.	Геометрия на плоскости. Скалярное и векторное произведения векторов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
36.	Геометрия на плоскости. Скалярное и векторное произведения векторов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
37.	Геометрия на плоскости. Площадь произвольных (в том числе и невыпуклых) многоугольников. Формула Пика.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
38.	Геометрия на плоскости. Площадь произвольных (в том числе и невыпуклых) многоугольников. Формула Пика.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
39.	Геометрия на плоскости. Площадь	1		1	1.http://acmp.ru/	Текущий

	произвольных (в том числе и невыпуклых) многоугольников. Формула Пика.				2. http://codeforces.ru	
40.	Геометрия на плоскости. Площадь произвольных (в том числе и невыпуклых) многоугольников. Формула Пика.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
41.	Геометрия на плоскости. Работа с углами.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
42.	Геометрия на плоскости. Работа с углами.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
43.	Геометрия на плоскости. Работа с углами.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
44.	Геометрия на плоскости. Работа с углами.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
45.	Рекурсия, перебор.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
46.	Рекурсия, перебор.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
47.	Рекурсия, перебор.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
48.	Рекурсия, перебор.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
49.	Математическое моделирование.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

50.	Математическое моделирование.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
51.	Математическое моделирование.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
52.	Математическое моделирование.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
53.	Двумерные массивы и векторы в C++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
54.	Двумерные массивы и векторы в C++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
55.	Двумерные массивы и векторы в C++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
56.	Двумерные массивы и векторы в C++	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
57.	Теория графов. Определение и методы задания графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
58.	Теория графов. Определение и методы задания графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
59.	Теория графов. Определение и методы задания графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
60.	Теория графов. Определение и методы задания графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
61.	Теория графов. Обходы графа.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

62.	Теория графов. Обходы графа.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
63.	Теория графов. Обходы графа.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
64.	Теория графов. Обходы графа.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
65.	Теория графов. Поиск путей и циклов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
66.	Теория графов. Поиск путей и циклов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
67.	Теория графов. Поиск путей и циклов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
68.	Теория графов. Поиск путей и циклов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
11 класс						
69.	Теория графов. Связность и двудольность.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
70.	Теория графов. Связность и двудольность.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
71.	Теория графов. Связность и двудольность.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
72.	Теория графов. Связность и двудольность.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

73.	Теория графов. Топологическая сортировка ациклических графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
74.	Теория графов. Топологическая сортировка ациклических графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
75.	Теория графов. Топологическая сортировка ациклических графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
76.	Теория графов. Топологическая сортировка ациклических графов.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
77.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
78.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
79.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
80.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
81.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
82.	Теория графов. Поиск мостов и точек сочленения.	1		1.	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
83.	Теория графов. Кратчайшие пути. Алгоритм Флойда.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
84.	Теория графов. Кратчайшие пути. Алгоритм Флойда.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

85.	Теория графов. Кратчайшие пути. Алгоритм Флойда.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
86.	Теория графов. Кратчайшие пути. Алгоритм Флойда.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
87.	Теория графов. Кратчайшие пути. Алгоритм Дейкстры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
88.	Теория графов. Кратчайшие пути. Алгоритм Дейкстры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
89.	Теория графов. Кратчайшие пути. Алгоритм Дейкстры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
90.	Теория графов. Кратчайшие пути. Алгоритм Дейкстры.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
91.	Теория графов. Кратчайшие пути. Алгоритм Форда-Беллмана.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
92.	Теория графов. Кратчайшие пути. Алгоритм Форда-Беллмана.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
93.	Теория графов. Кратчайшие пути. Алгоритм Форда-Беллмана.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
94.	Теория графов. Кратчайшие пути. Алгоритм Форда-Беллмана.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
95.	Теория графов. Минимальный каркас. Алгоритмы Прима и Краскала.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
96.	Теория графов. Минимальный	1		1	1.http://acmp.ru/	Текущий

	каркас. Алгоритмы Прима и Краскала.				2. http://codeforces.ru	
97.	Теория графов. Минимальный каркас. Алгоритмы Прима и Краскала.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
98.	Теория графов. Минимальный каркас. Алгоритмы Прима и Краскала.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
99.	Теория графов. Минимальный каркас. Алгоритмы Прима и Краскала.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
100.	Теория графов. Минимальный каркас. Алгоритмы Прима и Краскала.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
101.	Теория графов. Деревья, их обходы. Наименьший общий предок.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
102.	Теория графов. Деревья, их обходы. Наименьший общий предок.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
103.	Теория графов. Деревья, их обходы. Наименьший общий предок.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
104.	Теория графов. Деревья, их обходы. Наименьший общий предок.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
105.	Динамическое программирование. Простейшие задачи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

106.	Динамическое программирование. Простейшие задачи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
107.	Динамическое программирование. Простейшие задачи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
108.	Динамическое программирование. Простейшие задачи.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
109.	Динамическое программирование. Двумерная динамика, задача о рюкзаке.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
110.	Динамическое программирование. Двумерная динамика, задача о рюкзаке.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
111.	Динамическое программирование. Двумерная динамика, задача о рюкзаке.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
112.	Динамическое программирование. Двумерная динамика, задача о рюкзаке.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
113.	Динамическое программирование. Динамика на деревьях.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
114.	Динамическое программирование. Динамика на деревьях.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
115.	Динамическое программирование. Динамика на деревьях.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
116.	Динамическое программирование. Динамика на деревьях.	1		1	1.http://acmp.ru/	Текущий

					2. http://codeforces.ru	
117.	Динамическое программирование по подмножествам.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
118.	Динамическое программирование по подмножествам.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
119.	Динамическое программирование по подмножествам.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
120.	Динамическое программирование по подмножествам.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
121.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
122.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
123.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
124.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
125.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
126.	Динамическое программирование по профилю	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
127.	Структуры данных. Бинарная куча. Priority queue.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

128.	Структуры данных. Бинарная куча. Priority queue.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
129.	Структуры данных. Бинарная куча. Priority queue.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
130.	Структуры данных. Бинарная куча. Priority queue.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
131.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
132.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
133.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
134.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
135.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий
136.	Структуры данных. Дерево отрезков.	1		1	1.http://acmp.ru/ 2. http://codeforces.ru	Текущий

УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

ОБЯЗАТЕЛЬНЫЕ УЧЕБНЫЕ МАТЕРИАЛЫ ДЛЯ УЧЕНИКА

1. К.Ю. Поляков, Е.А. Еремин: Информатика. 10 класс. Углубленный уровень. Учебник. В 2-х частях. — М.: БИНОМ. Лаборатория знаний, 2016. — 352 с.
2. К.Ю. Поляков, Е.А. Еремин: Информатика. 11 класс. Углубленный уровень. Учебник. В 2-х частях. — М. : БИНОМ. Лаборатория знаний, 2016. — 240 с.

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ДЛЯ УЧИТЕЛЯ

1. С. А. Абрамов, Г. Г. Гнездилова, Е. Н. Капустина, М. И. Семон. Задачи по программированию. Москва, Изд-во Наука, 1988 г.
2. Л. Залогова, М. Плаксин, С. Русаков и др. Задачник – практикум. Москва, Лаборатория Базовых Знаний, 1999 г.
3. Андреева Е., Фалина И. Информатика: Систем счисления и компьютерная арифметика. М.: Лаборатория Базовых Знаний, 1999. - 256 с.
4. Ахо А., Дж. Хопкрофт, Дж. Ульман. Структуры данных и алгоритмы –М.: Издательский дом “Вильямс”, 2000
5. Антонов Ю.С. Олимпиадные задачи по информатике с математическим содержанием // Информатика и образование. - 2000. - №9. - С. 59-60.
6. Вирт Н. Алгортимы и структуры данных. –М.: Мир., 1989.
7. Казиев В.М. Развивающие задачи // Информатика и образование. - 1998. - №2. - С.79-83.
8. Кирюхин В.М. Информатика. Всероссийские олимпиады. –М.: Просвещение, -2008.
9. Кирюхин В.М., Лапунов А.В., Окулов С.М. Задачи по информатике. Международные олимпиады 1989-1996 гг. - М.: ABF, 1996. - 272 с.
10. Окулов С.М., Пестов А.А. 100 задач по информатике. Киров: Изд-во ВГПУ, 2000. - 272 с.
11. Окулов С.М., Пестов А.А., Пестов О.А. Информатика в задачах. - Киров: Изд-во ВГПУ, 1998. - 343 с.

12. Олимпиады по информатике 1999: Сб. задач/ Под ред. Н.Л. Андреевой. - Саратов: Изд-во Сарат. ун-та, 1999. - 48 с.
13. Прохоров В.В. Олимпиадные задачи по информатике // Информатика и образование. - 1991. - №3. - С. 67-75.
14. Чернов А.В. и др. Московские студенческие командные Олимпиады по программированию (сборник задач) / МГУ. - М., 1999. - 72 с.
15. Шестаков А.П. Задачи на длинную арифметику // Информатика и образование. - 6. Кнут Д.Э. Искусство программирования. Том 1. Основные алгоритмы. – М.: Вильямс, 2010. – 720 с.
16. Окулов С.М. Программирование в алгоритмах. – М.: БИНОМ. Лаборатория знаний, 2007. – 384 с.
17. Скиена С.С., Ревилла М.А. Олимпиадные задачи по программированию. Руководство по подготовке к соревнованиям. – М.: КУДИЦ-ОБРАЗ, 2005. – 416 с.
18. Долинский М.С. Решение сложных и олимпиадных задач по программированию. – СПб.: Питер, 2006. – 366 с.
19. Меньшиков Ф.В. Олимпиадные задачи по программированию. – СПб.: Питер, 2006. – 315 с.
20. Порублев И.Н., Ставровский А.Б. Алгоритмы и программы. Решение олимпиадных задач. – М.: Вильямс, 2007. – 480 с.
21. Московские олимпиады по информатике / Под ред. Е.В. Андреевой, В.М. Гуровица и В.А. Матюхина. – М.: МЦНМО, 2006. – 256 с.
22. Московские олимпиады по информатике 2002-2009 / Под ред. Е.В. Андреевой, В.М. Гуровица и В.А. Матюхина. – М.: МЦНМО, 2009. – 416 с.
23. Кирюхин В.М. Методика проведения и подготовки к участию в олимпиадах по информатике. Всероссийская олимпиада школьников. – М.: БИНОМ. Лаборатория знаний, 2012. – 280 с.

24. Ярославские олимпиады по информатике. Сборник задач с решениями. – М.: БИНОМ. Лаборатория знаний, 2010. – 408 с.
25. Русаков С.В. Олимпиады по базовому курсу информатики. – М.: БИНОМ. Лаборатория знаний, 2009. – 352 с.
26. Лебедев А.Б. Сборник задач по алгоритмизации и программированию для подготовки к ЕГЭ (с решениями). – М.: Феникс, 2010. – 448 с.
27. Брудно А.Л. Каплан Л.И. Московские олимпиады по программированию. –М.: Наука, гл. ред. физю-матю лит., 1990.
28. Кнут Д. Искусство программирования для ЭВМ. Т. 1-3. –М.: Вильямс, 2000.
29. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. –М.: МЦНМО, 1999.

30. Кристофидес Н. Теория графов. Алгоритмический подход. –М.: Мир, 1978.
31. Рейнгольд Э. Нивельгельт Ю. Део Н. Комбинаторные алгоритмы: теория и практика. –М.: Мир, 1980.
32. Бьерн Страуструп - Программирование. Принципы и практика использования C++. –М.: Вильямс, 2011. - 1248 с.
33. Стефан Р. Дэвис - C++ Для чайников. - М.: Вильямс, 2003 . - 336 с.
34. Прата С. - Язык программирования C++. 6 Издание. - М.: Вильямс, 2011 . - 1244 с.
35. Литвиненко Н. А. - Технология программирования на C++. - СПб.: БХВ-Петербург, 2010.- 281 с.

ЦИФРОВЫЕ ОБРАЗОВАТЕЛЬНЫЕ РЕСУРСЫ И РЕСУРСЫ СЕТИИНТЕРНЕТ

1. http://acmp.ru/
2. http://acm.timus.ru/
3. http://codeforces.ru
4. http://acm.mipt.ru/
5. http://habrahabr.ru/
6. http://e-maxx.ru/

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

Приложение 1

Тема 18: Геометрия на плоскости. Площадь произвольных (в том числе и невыпуклых) многоугольников. Формула Пика.

План занятия “Целые точки. Формула Пика”.

Задачи занятия:

1. Ознакомление с формулой Пика.
2. Нахождение числа целых точек на отрезке и периметре.
3. Нахождение площади произвольной фигуры.

Ход занятия

Решим задачу №209 “Целые точки”:

Время на тест 1 сек. Память 16 Мб. Сложность 64%.

Многоугольник (не обязательно выпуклый) на плоскости задан координатами своих вершин. Требуется подсчитать количество точек с целочисленными координатами, лежащих внутри него (но не на его границе).

Входные данные. В первой строке входного файла INPUT.TXT содержится N (3<=N<=1000) – число вершин многоугольника. В следующих N строках идут координаты (X_i и Y_i) вершин многоугольника в порядке обхода по часовой стрелке. X_i и Y_i – целые числа, по модулю не превосходящие 10^6.

Выходные данные. Ваша программа должна вывести в выходной файл OUTPUT.TXT одно число – ответ на задачу.

Примеры.

INPUT.TXT

```
4
-1 -1
-1 1
1 1
1 -1
```

OUTPUT.TXT

```
1
```

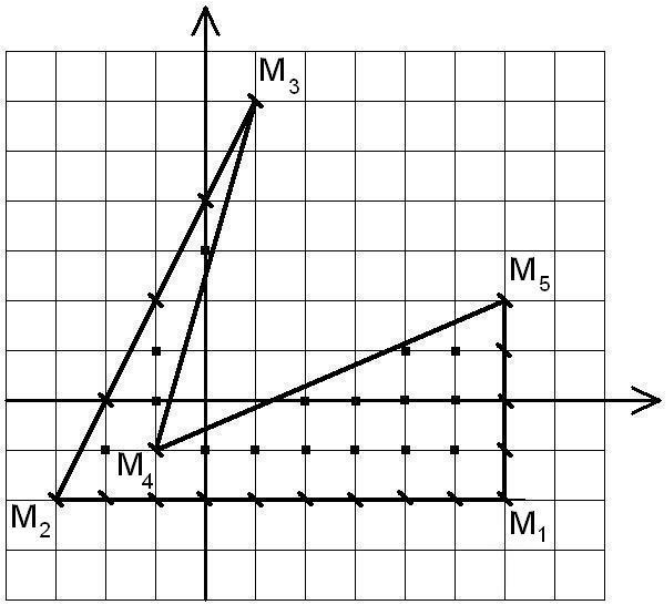
INPUT.TXT

```
5
6 -2
-3 -2
1 6
-1 -1
6 2
```

OUTPUT.TXT

```
16
```

На рис. 1 приведен чертеж для второго примера. Жирными точками указаны искомые целые точки, находящиеся внутри многоугольника. Их 16 штук, потому в файл ответа OUTPUT.TXT программа должна выдать 16. Легко видеть, что эти точки расположены внутри фигуры довольно случайно, и установить факт нахождения конкретной целой точки внутри или за пределами фигуры без чертежа достаточно сложно.



Важнейшим условием является то, что вершины многоугольника так же целочисленные. Это позволяет применить формулу Пика, которая связывает площадь фигуры S, число целых точек, содержащихся строго внутри нее N_{in} и число целых точек, расположенных на ее периметре N_{per}.

рис. 1

Формула Пика имеет вид

$$S=N_{in}+1/2N_{per} - 1.$$

Для начала проверим эту довольно неожиданную формулу на нашем примере. Для фигуры на рисунке площадь S=24.5, N_{in}=16, N_{per}=19 (эти точки выделены косыми засечками). Действительно 24.5=16 + 9.5 – 1.

Преобразуем формулу Пика так как нам нужно:

$$2N_{in} = 2S - N_{per}+2.$$

Таким образом, зная удвоенную площадь фигуры и число целых точек на ее периметре можно легко узнать удвоенное число целых точек, расположенных внутри фигуры. Отсюда вместо одной трудной задачи получаем две более простых:

1. по координатам вершин многоугольника найти число целых точек на ее периметре;
2. по координатам вершин многоугольника найти ее площадь.

Для решения задачи 1 разобьем периметр фигуры на отрезки. Пусть нам дан отрезок AB с координатами (X_A ,Y_A) и (X_B ,Y_B). Найти число целых точек на отрезке AB. Здесь нам поможет нахождение наибольшего общего делителя. Действительно, число целых точек (включая вершины) на отрезке с целочисленными вершинами есть

НОД(X,Y) +1, где X=| $X_A - X_B$ | а Y=| $Y_A - Y_B$|.

Полезно вспомнить, что такое НОД – это наибольшее натуральное число, которое делит нацело оба входящих в НОД натуральных числа.

Для отрезка M_1M_2 X=9, Y=0, НОД(9,0)=9. Число целых точек на нем 10.

Для отрезка M_2M_3 X=4, Y=8, НОД(4,8)=4. Число целых точек на нем 5.

Для отрезка M_3M_4 X=2, Y=7, НОД(2,7)=1. Число целых точек на нем 2.

Для отрезка M_4M_5 X=7, Y=3, НОД(7,3)=1. Число целых точек на нем 2.

Для отрезка M_5M_1 X=0, Y=4, НОД(0,4)=4. Число целых точек на нем 5.

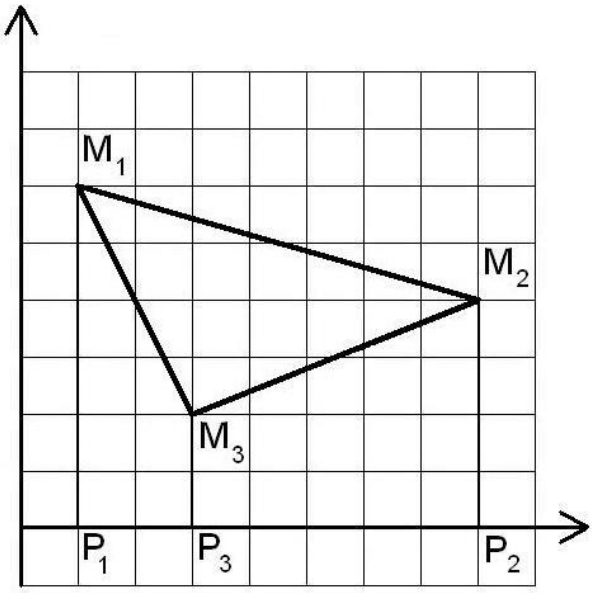
Учтем, что при суммировании каждая вершина многоугольника будет входить в два отрезка и в итоге получим N_{per}= НОД(| $X_1 – X_2$ | , | $Y_1 – Y_2$ |)+ НОД(| $X_2 – X_3$| , | $Y_2 – Y_3$ |)+ … + НОД(| $X_N – X_1$ | , | $Y_N – Y_1$ |).

Для фигуры на рис. 1: 9+4+1+1+4=19 и это совпадает с N_{per}.

Вспомним, что НОД можно найти при помощи следующей рекурсивной функции

```
int NOD (int a, int b){
 if (b ==0) return(a); else return NOD(b, a mod b);
}
```

Таким образом, мы уже можем решить задачу 1.



Для решения второй задачи воспользуемся идеей площади со знаком. Например, решим ее при помощи трапеций. На рис 2. приведен простой пример, позволяющий наглядно проиллюстрировать идею этого метода. Из каждой вершины M_i проводится перпендикуляр на ось OX и получается проекция этой вершины P_i. Далее рассматривается система трапеций $M_iM_{i+1}P_{i+1}P_i$. Это действительно трапеции, у них есть два параллельных основания M_iP_i и $M_{i+1}P_{i+1}$. Если обратиться к рис. 2, то можно заметить, что площадь

рис. 2

треугольника $M_1M_2M_3$ можно получить как площадь трапеции $M_1M_2P_2P_1$ минус площадь трапеции $M_2M_3P_3P_2$ минус площадь трапеции $M_3M_1P_1P_3$. Здесь полезно заметить, что, так как мы движемся по периметру, то для точки M_N следующей N+1-й точкой будет, очевидно, точка M_1.

В данном случае вместо того, чтобы считать площадь величиной положительной и потом ее отнимать со знаком минус значительно удобнее рассмотреть площадь со знаком и только суммировать получающиеся величины. Тогда если мы идем от точки M_i к точке M_{i+1} слева направо, площадь трапеции $M_iM_{i+1}P_{i+1}P_i$ считается положительной и если идем справа налево, то площадь трапеции считается отрицательной. Вспомним, что площадь трапеции есть полусумма оснований, умноженная на высоту, но так как нам нужна удвоенная площадь для подстановки в формулу Пика, то будем искать просто сумму оснований, умноженную на высоту. Так как основания со знаком есть величины Y_i, а высота со знаком есть $X_{i+1} - X_i$, то удвоенная площадь со знаком есть величина $(X_{i+1} - X_i)*(Y_{i+1} + Y_i)$. Осталось просуммировать эти произведения и получить искомую площадь фигуры со знаком. Если обход происходит по часовой стрелке, то эта площадь будет со знаком плюс, если против часовой стрелки, то со знаком минус. На всякий случай, чтобы не ошибиться со знаком, полезно после взятия суммы для полученной величины взять модуль и получить искомую положительную площадь.

Для фигуры на рис.1:

Трапеция $M_1M_2P_2P_1$ дает $(X_2 - X_1)*(Y_2 + Y_1)$ =(-3-6)*(-2+(-2))=36.

Трапеция $M_2M_3P_3P_2$ дает $(X_3 - X_2)*(Y_3 + Y_2)$ =(1-(-3))*(6+(-2))=16.

Трапеция $M_3M_4P_4P_3$ дает $(X_4 - X_3)*(Y_4 + Y_3)$ =(-1-1)*(-1+6)=-10.

Трапеция $M_4M_5P_5P_4$ дает $(X_5 - X_4)*(Y_5 + Y_4)$ =(6-(-1))*(2+(-1))=7.

Трапеция $M_5M_1P_1P_5$ дает $(X_1 - X_5)*(Y_1 + Y_5)$ =(6-6)*(-2+2)=0.

Суммируя, получаем 36+16+(-10)+7+0=49, то есть удвоенную площадь 2*S=2*24.5.

Осталось подставить в формулу $2S - N_{per}+2$ полученные в задачах 1 и 2 величины и получить удвоенное количество точек, расположенных строго внутри многоугольника. В файл OUTPUT.TXT нужно выдать это количество, поделенное пополам .

Для нашего примера:

(49 – 19 +2) / 2= 16, что и является правильным ответом.

Далее каждый ученик рисует на клетчатой бумаге свой собственный, достаточно нетривиальный невыпуклый многоугольник и для него находит S, N_{in} , N_{per} просто подсчетом на чертеже, а затем при помощи только что разобранного алгоритма. Результаты должны совпасть.

После этого, при помощи какой-либо оболочки (например Codeblocks) происходит программирование рассмотренного алгоритма и проверка полученной программы на сайте acmp.ru.

Приложение 2

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

Все предлагаемые ниже задания предлагались на школьных олимпиадах по информатике и программированию различных уровней. Источник заданий http://acmp.ru/.

1. Задачи для начинающих

Торт

(Время: 1 сек. Память: 16 Мб Сложность: 6%)

На свой день рождения Петя купил красивый и вкусный торт, который имел идеально круглую форму. Петя не знал, сколько гостей придет на его день рождения, поэтому вынужден был разработать алгоритм, согласно которому он сможет быстро разрезать торт на N равных частей. Следует учесть, что разрезы торта можно производить как по радиусу, так и по диаметру.

Помогите Пете решить эту задачу, определив наименьшее число разрезов торта по заданному числу гостей.

Входные данные

Входной файл INPUT.TXT содержит натуральное число N – число гостей, включая самого виновника торжества (N <= 1000).

Выходные данные

В выходной файл OUTPUT.TXT выведите минимально возможное число разрезов торта.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2	1
2	3	

Бинарные числа

(Время: 1 сек. Память: 16 Мб Сложность: 8%)

Говорят, что плохой программист – это тот, кто считает, что в одном килобайте 1000 байт, а хороший программист – это тот, кто полагает, что в одном километре 1024 метра.

Многим эта шутка понятна, так как все знают, что в процессах, связанных с информатикой и компьютерной техникой, фигурирует множество значений, выражаемых степенью двойки, то есть чисел вида 2^K, где K – некоторое неотрицательное целое число. Назовем такие числа *бинарными*. Это такие числа как 2, 4, 8, 16, 32 и т.д. Действительно, когда речь идет о размере памяти или о разрешении экрана монитора, то мы часто наталкиваемся на бинарные числа. Все это связано с принципом хранения информации в памяти ЭВМ.

Задано целое число N. Требуется определить, является ли оно бинарным.

Входные данные

Входной файл INPUT.TXT содержит единственное целое число N, не превосходящее 10000 по абсолютной величине.

Выходные данные

В выходной файл OUTPUT.TXT выведите YES, если заданное число является бинарным, и NO в противном случае.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1024	YES
2	23	NO

Разворот

(Время: 1 сек. Память: 16 Мб Сложность: 9%)

Дано натуральное число N и последовательность из N элементов. Требуется вывести эту последовательность в обратном порядке.

Входные данные

В первой строке входного файла INPUT.TXT записано натуральное число N ($N \le 10^3$). Во второй строке через пробел идут N целых чисел, по модулю не превосходящих 10^3 - элементы последовательности.

Выходные данные

В выходной файл OUTPUT.TXT выведите заданную последовательность в обратном порядке.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 1 2 3	3 2 1

Внеземные гости

(Время: 1 сек. Память: 16 Мб Сложность: 10%)

Недавно на поле фермера Джона были обнаружены следы приземления летающих тарелок. Об этом даже писала газета Mew Yorc Courier.

Поле фермера Джона имеет форму круга радиусом r_1. По сообщениям журналистов были обнаружены два следа от летающих тарелок, имевшие форму кругов. Один из них имел радиус r_2, второй - радиус r_3. Также сообщается, что они находились внутри поля фермера Джона и не пересекались (при этом, они, возможно, касались друг друга и/или границы поля).

Поскольку журналисты часто склонны преувеличивать масштабы событий, необходимо написать программу, которая будет проверять, могли ли иметь место события, описанные в газете.

Входные данные

Входной файл INPUT.TXT содержит три целых положительных числа - r_1, r_2, r_3 ($1 \le r_1, r_2, r_3 \le 10^9$).

Выходные данные

В выходной файл OUTPUT.TXT выведите слово YES, если информация, опубликованная в газете, может соответствовать правде, и слово NO - иначе.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	10 10 10	NO
2	10 3 4	YES

Золотой песок

(Время: 1 сек. Память: 16 Мб Сложность: 10%)

Сотрудники завода по производству золотого песка из воздуха решили поправить свое финансовое положение. Они пробрались на склад завода, где хранился золотой песок трех видов. Один килограмм золотого песка первого вида они смогли бы продать за A_1 рублей, второго вида – за A_2 рублей, а третьего вида – за A_3 рублей. Так получилось, что у сотрудников оказалось с собой только три емкости: первая была рассчитана на B_1 килограмм груза, вторая на B_2 килограмм, а третья на B_3 килограмм. Им надо было заполнить полностью все емкости таким образом, чтобы получить как можно больше денег за весь песок. При заполнении емкостей нельзя смешивать песок разных видов, то есть, в одну емкость помещать более одного вида песка, и заполнять емкости песком так, чтобы один вид песка находился более чем в одной емкости.

Требуется написать программу, которая определяет, за какую сумму предприимчивые сотрудники смогут продать весь песок в случае наилучшего для себя заполнения емкостей песком.

Входные данные

В единственной строке входного файла INPUT.TXT записано 6 натуральных чисел A_1, A_2, A_3, B_1, B_2, B_3, записанных в одной строке через пробел. Все числа не превосходят 100.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести единственное целое число – сумму в рублях, которую смогут сотрудники заработать в случае наилучшего для себя заполнения емкостей песком.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	1 2 3 3 2 1	14

Клавиатура

(Время: 1 сек. Память: 16 Мб Сложность: 11%)

Для данной буквы латинского алфавита нужно вывести справа стоящую букву на стандартной клавиатуре. При этом клавиатура замкнута, т.е. справа от буквы «p» стоит буква «a», от буквы «l» стоит буква «z», а от буквы «m» — буква «q».

Входные данные

Входной файл INPUT.TXT содержит один символ — маленькую букву латинского алфавита.

Выходные данные

В выходной файл OUTPUT.TXT следует вывести букву стоящую справа от заданной буквы, с учетом замкнутости клавиатуры.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	q	w
2	t	y
3	p	a
4	l	z
5	m	q

Ремонт

(Время: 1 сек. Память: 16 Мб Сложность: 11%)

Ваш любимый дядя – директор фирмы, которая делает евроремонты в офисах. В связи с финансово-экономическим кризисом, дядюшка решил оптимизировать свое предприятие.

Давно ходят слухи, что бригадир в дядюшкиной фирме покупает лишнее количество стройматериалов, а остатки использует для отделки своей новой дачи. Ваш дядя заинтересовался, сколько в действительности банок краски необходимо для покраски стены в офисе длиной L метров, шириной – W и высотой – H, если одной банки хватает на 16м^2, а размерами дверей и окон можно пренебречь? Заказов много, поэтому дядя попросил написать программу, которая будет все это считать.

Входные данные

Входной файл INPUT.TXT содержит три натуральных числа L, W, H – длину, ширину и высоту офиса в метрах соответственно, каждое из которых не превышает 1000.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число – минимальное количество банок краски, необходимых для покраски офиса.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	8 8 2	4
2	1 1 3	1

Автобусная экскурсия

(Время: 1 сек. Память: 16 Мб Сложность: 14%)

Оргкомитет Московской городской олимпиады решил организовать обзорную экскурсию по Москве для участников олимпиады. Для этого был заказан двухэтажный автобус (участников олимпиады достаточно много и в обычный они не умещаются) высотой 437 сантиметров. На экскурсионном маршруте встречаются N мостов. Жюри и оргкомитет олимпиады очень обеспокоены тем, что высокий двухэтажный автобус может не проехать под одним из них. Им удалось выяснить точную высоту каждого из мостов. Автобус может проехать под мостом тогда и только тогда, когда высота моста превосходит высоту автобуса.

Помогите организаторам узнать, закончится ли экскурсия благополучно, а если нет, то установить, где произойдет авария.

Входные данные

Во входном файле INPUT.TXT сначала содержится число N (1<=N<=1000). Далее идут N натуральных чисел, не превосходящих 10000 - высоты мостов в сантиметрах в том порядке, в котором они встречаются на пути автобуса.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести фразу "No crash", если экскурсия закончится благополучно. Если же произойдет авария, то нужно вывести сообщение "Crash k", где k - номер моста, где произойдет авария. Фразы выводить без кавычек ровно с одним пробелом внутри.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1 763	No crash
2	3 763 245 113	Crash 2
3	1 437	Crash 1

Арбузы

(Время: 1 сек. Память: 16 Мб Сложность: 14%)

Иван Васильевич пришел на рынок и решил купить два арбуза: один для себя, а другой для тещи. Понятно, что для себя нужно выбрать арбуз потяжелей, а для тещи полегче. Но вот незадача: арбузов слишком много и он

не знает как же выбрать самый легкий и самый тяжелый арбуз? Помогите ему!

Входные данные

В первой строке входного файла INPUT.TXT задано одно число N – количество арбузов. Вторая строка содержит N чисел, записанных через пробел. Здесь каждое число – это масса соответствующего арбуза. Все числа натуральные и не превышают 30000.

Выходные данные

В выходной файл OUTPUT.TXT нужно вывести два числа через пробел: массу арбуза, который Иван Васильевич купит теще и массу арбуза, который он купит себе.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	5 5 1 6 5 9	1 9

Подмассив массива

(Время: 1 сек. Память: 16 Мб Сложность: 15%)

Пусть задан массив целых чисел a_1, a_2, ..., a_n. Назовем его подмассивом f(i,j) массив, составленный из чисел массива a_i, a_{i+1},..., a_{j-1}, a_j. Напишите программу, которая будет выводить подмассивы массива a.

Входные данные

Первая строка входного файла INPUT.TXT содержит число n (1 <= n <= 1000) - количество элементов в массиве a. Во второй строке содержатся числа a_1, a_2, … , a_n разделенные пробелом. Все a_i находятся в диапазоне от -2^{31} до $2^{31} - 1$. В третьей строке находится m (1 <= m <= 100) — количество подмассивов, которые необходимо вывести. Следующие m строк содержат пары чисел i_k, j_k (1 <= i_k <= j_k <= n).

Выходные данные

В выходной файл OUTPUT.TXT для каждой пары (i_k,j_k) в отдельной строке выведите подмассив f(i_k,j_k).

Пример

№	INPUT.TXT	OUTPUT.TXT
1	6 1 2 3 4 5 6	1 2 3 4 5 6

	5 1 1 2 6 3 4 5 6 2 4	3 4 5 6 2 3 4

Метро

(Время: 1 сек. Память: 16 Мб Сложность: 16%)

Витя работает недалеко от одной из станций кольцевой линии метро, а живет рядом с другой станцией той же линии. Требуется выяснить, мимо какого наименьшего количества промежуточных станций необходимо проехать Вите по кольцу, чтобы добраться с работы домой.

Входные данные

Во входном файле INPUT.TXT заданы три числа: сначала N – общее количество станций кольцевой линии, а затем i и j – номера станции, на которой Витя садится, и станции, на которой он должен выйти. Станции пронумерованы подряд натуральными числами 1, 2, 3, …, N (1-я станция – соседняя с N-й), N не превосходит 100. Числа i и j не совпадают. Все числа разделены пробелом.

Выходные данные

В выходной файл OUTPUT.TXT требуется вывести минимальное количество промежуточных станций (не считая станции посадки и высадки), которые необходимо проехать Вите.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	100 5 6	0
2	10 1 9	1

Нули

(Время: 1 сек. Память: 16 Мб Сложность: 16%)

Требуется найти самую длинную непрерывную цепочку нулей в последовательности нулей и единиц.

Входные данные

В единственной строке входного файла INPUT.TXT записана последовательность нулей и единиц (без пробелов). Суммарное количество цифр не превышает 100.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести искомую длину цепочки нулей.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	00101110000110	4

Болты и гайки

(Время: 1 сек. Память: 16 Мб Сложность: 17%)

Вновь созданная фирма купила заброшенные склады на окраине города. Новому заведующему складами поручили произвести учёт в короткие сроки. Всё шло хорошо, пока случайно не рассыпали контейнеры с болтами и гайками на каждом складе, после чего собрали их в общие (для болтов и гаек) контейнеры, потеряв при этом несколько деталей.

Помогите оценить нанесённый ущерб на каждом складе, приняв во внимание, что, помимо потерянных деталей, болт (или гайка) считается непригодным, если он не имеет соответствующей гайки (или болта).

Входные данные

Во входном файле INPUT.TXT описано текущее положение на складе. В первой строке через пробел записаны три целых числа: k1, l1, m1 – начальное число болтов (100<=k1<=30000, k1 кратно 100), процент потерянных деталей (0<=l1<=100) и стоимость одного болта (1<=m1<=100) соответственно. Во второй строке через пробел записаны также три целых числа: k2, l2, m2 – начальное число гаек (100<=k2<=30000, k2 кратно 100), процент потерянных деталей (0<=l2<=100) и стоимость одной гайки (1<=m2<=100) соответственно.

Выходные данные

В выходной OUTPUT.TXT выведите одно целое число – размер ущерба.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1000 10 100 1200 20 90	37000

2	5000 15 23 4000 17 22	53600

Треугольник - 3

(Время: 1 сек. Память: 16 Мб Сложность: 17%)

Даны длины трех отрезков. Требуется проверить: могут ли они являться сторонами треугольника.

Входные данные

Входной файл INPUT.TXT содержит 3 натуральных числа X Y Z – длины заданных отрезков. Длины отрезков записаны в одной строке через пробел и не превышают 1000.

Выходные данные

В выходной файл OUTPUT.TXT выведите YES, если отрезки могут быть сторонами треугольника и NO в противном случае.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1 2 3	YES
2	1 1 5	NO

Детали

(Время: 1 сек. Память: 16 Мб Сложность: 20%)

На клеточном поле N•M расположены две жёсткие детали. Деталь A накрывает в каждой строке несколько (не ноль) первых клеток, деталь B — несколько (не ноль) последних; каждая клетка либо полностью накрыта одной из деталей, либо нет.

A	A	.	B	B	B
A	.	.	.	.	B
A	A	A	.	.	B
A	.	.	B	B	B

Деталь B начинают двигать влево, не поворачивая, пока она не упрётся в A хотя бы одной клеткой. Определите, на сколько клеток будет сдвинута деталь B.

Входные данные

В первой строке входного файла INPUT.TXT записано два числа N и M — число строк и столбцов соответственно ($1 \le N, M \le 100$). Далее следуют N строк, задающих расположение деталей. В каждой находится ровно M символов "A" (клетка, накрытая деталью A), "B" (накрытая деталью B) или "." (свободная клетка).

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно число — ответ на задачу.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 6 AA.BBB A....B AAA..B A..BBB	1

Лифт

(Время: 1 сек. Память: 16 Мб Сложность: 20%)

В доме Вилли установили скоростной лифт новой экспериментальной модели. В этом лифте кнопки с номерами этажей заменены двумя другими кнопками. При нажатии на первую кнопку лифт поднимается на один этаж вверх, а при нажатии на вторую – опускается на один этаж вниз.

Младшему брату Вилли Дилли очень нравится кататься на новом лифте. Он катается на нём до тех пор, пока не побывает на каждом из этажей хотя бы по одному разу. После этого Дилли довольный возвращается домой.

Зная порядок, в котором Дилли нажимал на кнопки лифта, попробуйте определить общее количество этажей в доме Вилли и Дилли.

Входные данные

Первая строка входного файла INPUT.TXT содержит последовательность нажатий на кнопки лифта. Символ «1» означает, что была нажата первая кнопка, а символ «2» – что была нажата вторая кнопка. Символы «1» и «2» не разделены пробелами. Количество нажатий не превосходит 100. Гарантируется, что лифт никогда не опускался ниже первого и не поднимался выше последнего этажа.

Выходные данные

В выходной файл OUTPUT.TXT следует вывести одно число – количество этажей в доме Вилли и Дилли.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	11	3
2	21212	2
3	1221221221221	6

Очередь

(Время: 1 сек. Память: 16 Мб Сложность: 20%)

Студент Василий живет в общежитии. Отделение банка, в котором он производит оплату за проживание, имеет всего две кассы, поэтому почти всегда длинная очередь к ним. Первая касса открывается в 8.00, вторая – в 8.05. Последний клиент будет принят в 20.00. Очередь единая, и очередной клиент обслуживается, как только освобождается одна из касс. На обслуживание одного клиента уходит ровно 10 минут. Василий приходит ровно в 8.00 и видит, сколько человек стоит перед ним. Требуется определить, сколько времени ему придется простоять в очереди, и вообще обслужат ли его сегодня.

Входные данные

Входной файл INPUT.TXT содержит единственное натурально число K – номер Василия в очереди (K < 250).

Выходные данные

В выходной файл OUTPUT.TXT выводится строка «NO», если Василий сегодня заплатить уже не успеет, и время его ожидания (в формате «X Y», где X – количество целых часов, которые простоит в очереди Василий, и Y – количество минут), если все же успеет заплатить.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1	0 0
2	20	1 35
3	235	NO

Автобусы - 2

(Время: 1 сек. Память: 16 Мб Сложность: 23%)

Для заезда в оздоровительный лагерь организаторы решили заказать автобусы. Известно, что в лагерь собираются поехать N детей и M взрослых. Каждый автобус вмещает K человек. В каждом автобусе, в котором поедут дети, должно быть не менее двух взрослых.

Определите, удастся ли отправить в лагерь всех детей и взрослых, и если да, то какое минимальное количество автобусов требуется для этого заказать.

Входные данные

В единственной строке входного файла INPUT.TXT записано через пробел 3 натуральных числа - N, M и K, каждое из них не превосходит 10 000.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести количество автобусов, которые нужно заказать. Если же отправить всех в лагерь невозможно, выведите 0 (ноль).

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	10 4 7	2
2	10 4 5	0

Время прибытия

(Время: 1 сек. Память: 16 Мб Сложность: 26%)

Задано время отправления поезда и время в пути до конечной станции. Требуется написать программу, которая найдет время прибытия этого поезда (возможно, в другие сутки).

Входные данные

Входной файл INPUT.TXT содержит две строки. В первой строке задано время отправления, а во второй строке – время в пути. Время отправления задается в формате «HH:MM», где HH время в часах, которое принимает значение от 00 до 23, MM – время в минутах, которое принимает значение от 00 до 59. Время в пути задается двумя неотрицательными целыми числами – количество часов и количество минут. Числа разделяются одним пробелом. Количество часов не превышает 120, минут – 59.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одну строку – время прибытия поезда на конечную станцию. Формат вывода этого времени совпадает с форматом ввода времени отправления.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	00:00 10 10	10:10
2	01:02 4 6	05:08
3	11:00 22 0	09:00

Железная дорога

(Время: 1 сек. Память: 16 Мб Сложность: 26%)

При строительстве новой железной дороги возникли проблемы. Дорога пролегает по холмистой местности, однако сами пути должны идти строго горизонтально. Поэтому руководство строительной компании приняло решение выровнять поверхность земли на этом участке. Главная проблема состоит в том, что привозить или вывозить землю на стройку стоит 10000$ за кубический метр. Поскольку бюджет железной дороги невелик, этого нельзя себе позволить.

Поэтому главный инженер принял решение выровнять поверхность, используя только землю, из которой состоят холмы. Теперь самая сложная задача состоит в том, чтобы выяснить высоту над уровнем моря, на которой будет пролегать дорога. Это ответственное задание было поручено Вам.

Через каждый метр от начала участка была измерена высота над уровнем моря. Напишите программу, которая по данным измерений рассчитывает искомую высоту.

Входные данные

Первая строка входного файла INPUT.TXT содержит количество N (1 < N <= 30000) точек, в которых была замерена высота. Вторая строка содержит результаты замеров – i-ое число строки содержит высоту над уровнем моря точки, находящейся на расстоянии (i-1) метр от начала участка. Все высоты – целые неотрицательные числа, не превосходящие 10000. Считайте, что между соседними точками измерений земная поверхность строго прямолинейна.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу с точностью, не меньшей 10^{-5}.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 0 1 1 0	0.6666666667
2	5 2 2 2 2 2	2.0000000000

Наименьшая система счисления

(Время: 1 сек. Память: 16 Мб Сложность: 26%)

Известно, что основанием позиционной системы счисления называют количество различных символов, используемых для записи чисел в данной системе счисления. Также известно, что любое число x в b-ичной системе счисления имеет вид $x=a_0 \cdot b^0+a_1 \cdot b^1+\ldots+a_n \cdot b^n$, где $b \geq 2$ и $0 \leq a_i < b$.

Для записи чисел в b-ичной системе счисления, где $b \leq 36$, могут быть использованы первые b символов из следующего списка 0,1,…, 9, A, B, …, Z. Например, для записи чисел в троичной системы используются символы 0, 1, 2, а в двенадцатеричной - 0,1,…, 9, A, B.

Требуется написать программу, которая по входной строке S определит, является ли данная строка записью числа в системе счисления, с основанием не большим 36, и, если является, определит минимальное основание этой системы счисления.

Входные данные

Входной файл INPUT.TXT содержит в единственной строке входную строку. Длина строки не превышает 255. Все символы строки имеют коды от 32 до 127.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одно число. Если строка является записью числа в некоторой системе счисления, то нужно вывести минимальное основание такой системы счисления. Иначе вывести -1.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	123	4
2	ABCDEF	16

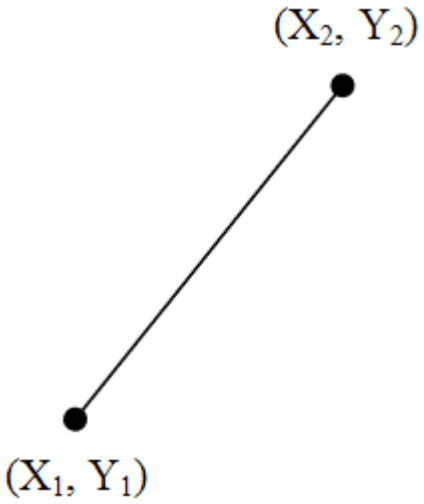
3	AD%AF	-1

2. Геометрия

Длина отрезка

(Время: 1 сек. Память: 16 Мб Сложность: 12%)

Отрезок задан координатами своих концевых точек. Требуется вычислить длину этого отрезка.



Входные данные

Входной файл INPUT.TXT содержит координаты концов отрезка в формате X_1 Y_1 X_2 Y_2 . Все координаты – целые числа, не превышающие 1000 по абсолютной величине.

Выходные данные

В выходной файл OUTPUT.TXT выведите длину отрезка с точностью 10^{-5}.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 4 8 4	5

Две окружности

(Время: 1 сек. Память: 16 Мб Сложность: 17%)

На плоскости даны две окружности. Требуется проверить, пересекаются ли они.

Входные данные

Входной файл INPUT.TXT состоит из двух строк. На каждой строке записана информация об одной окружности – координаты ее центра x и y (целые числа, по модулю не превосходящие 5000) и радиус (целое число $1 \le r \le 1000$).

Выходные данные

В выходной файл OUTPUT.TXT выведите «YES», если окружности пересекаются, и «NO» в противном случае.

Примеры

№	INPUT.TXT	OUTPUT.TXT

1	0 0 2 0 3 2	YES
2	1 1 1 4 4 1	NO

Суслик и собака

(Время: 1 сек. Память: 16 Мб Сложность: 19%)

На большом поле находятся суслик и собака. Собака хочет суслика съесть, а суслик хочет оказаться в безопасности, добежав до одной из норок, выкопанных в поле. Ни собака, ни суслик в математике не сильны, но, с другой стороны, они и не беспросветно глупы. Суслик выбирает определенную норку и бежит к ней по прямой с определенной скоростью. Собака, которая очень хорошо понимает язык телодвижений, угадывает, к какой норке бежит суслик, и устремляется к ней со скоростью вдвое большей скорости суслика. Если собака добегает до норки первой, то она съедает суслика; иначе суслик спасается.

Требуется написать программу, которая поможет суслику выбрать норку, в которой он может спастись, если таковая существует.

Входные данные

Во входном файле INPUT.TXT записано в первой строке два числа – координаты суслика. Во второй строке записаны два числа – координаты собаки. В третьей строке записано число n – число норок на поле. В следующих n строках записаны координаты норок. Все координаты являются целыми числами, по модулю не превышающими 10000, и записываются через пробел. Количество норок не превышает 1000.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести число – номер норки, если у суслика есть возможность в ней спастись. Если у суслика есть возможность спрятаться в нескольких норках, то выведите ту, которая первая шла во входных данных. Если суслик не может спастись, то выведите в выходной файл «NO» (без кавычек).

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	10 10 20 20 1 15 15	NO
2	20 20	2

	10 10 2 15 15 25 25	

Прямоугольник - 2

(Время: 1 сек. Память: 16 Мб Сложность: 27%)

Заданы координаты трех вершин прямоугольника. Необходимо определить координаты четвертой вершины.

Входные данные

Во входном файле INPUT.TXT записаны через пробел координаты трех вершин прямоугольника в произвольном порядке в формате x_1 y_1 x_2 y_2 x_3 y_3. Все числа целые, не превосходящие 1000 по абсолютной величине.

Выходные данные

В выходной файл OUTPUT.TXT нужно вывести через пробел координаты четвертой вершины прямоугольника.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	0 3 0 0 5 0	5 3
2	1 4 8 3 7 6	2 1

Стрелок

(Время: 1 сек. Память: 16 Мб Сложность: 28%)

Стрелок стоит в центре стрельбища. На стрельбище несколько мишеней. Пули стрелка пробивают мишени насквозь, не теряя скорости, и могут поразить все мишени, стоящие на одной линии.

Будем считать, что стрелок стоит в центре начала координат. Известны координаты всех мишеней (для простоты будем считать их геометрические размеры пренебрежимо малыми). Определите минимальное число выстрелов, необходимых стрелку для поражения всех мишеней.

Входные данные

Первая строка входного файла INPUT.TXT содержит натуральное число N – количество мишеней ($N \leq 20$). Далее идет N строк с информацией о координатах каждой мишени, при этом в каждой строке указывается два целых числа через пробел X и Y ($-10 <= X, Y <= 10$).

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число – наименьшее количество выстрелов, необходимых для поражения всех мишеней.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 2 2 -2 2 -2 -2 2 -2	4
2	6 2 2 -2 2 -2 -2 2 -2 1 1 -1 3	5

Грибной дождь

(Время: 1 сек. Память: 16 Мб Сложность: 30%)

На лесной опушке растет дружная семейка грибов. Местоположение каждого гриба задается координатами X, Y, а шляпка гриба имеет радиус R. Когда идет дождь, радиус шляпки каждого гриба непрерывно и равномерно увеличивается со скоростью 1 сантиметр в минуту. Когда дождь заканчивается (а он идет не более T минут), шляпки прекращают расти. Если во время дождя шляпки двух грибов соприкоснулись, то они немедленно перестают расти, чтобы не навредить друг другу. Грибы очень дружные, поэтому если перестают расти два гриба, то и все остальные тоже не растут.

Требуется посчитать, на сколько сантиметров увеличился радиус шляпки каждого гриба после завершения дождя.

Входные данные

Первая строка входного файла INPUT.TXT содержит два натуральных числа: количество грибов K (K<=10) и длительность дождя T (T<=100). Следующие K строк содержат описание грибов: целые координаты X и Y (0<=X, Y<=100) и радиус шляпки R (1<=R<=10). Координаты и радиус даны в сантиметрах.

Выходные данные

В выходной файл OUTPUT.TXT выведите величину в сантиметрах, на которую увеличится радиус всех грибов. Результат следует вывести с точностью, не меньшей, чем два знака после запятой.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2 1 0 0 1 2 2 1	0.41
2	3 2 0 0 1 5 5 1 10 10 1	2.00

Атака летающих тарелок

(Время: 1 сек. Память: 16 Мб Сложность: 36%)

Вы работаете в фирме, занимающейся разработкой компьютерных игр. Сейчас вы занимаетесь разработкой новой компьютерной игры "Атака летающих тарелок". По сюжету игры на планету Зумла приземляются летающие тарелки, и их надо уничтожать. Игрок управляет лазерной пушкой. Для того, чтобы произвести выстрел он указывает две точки на поверхности Зумлы (которая в игре считается плоской), через которые должен проходить лазерный луч (который является прямой).

Вы должны написать программу, определяющую, какие летающие тарелки были уничтожены выстрелом.

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число n ($1 \le n \le 30000$) - число приземлившихся летающих тарелок. Вторая строка содержит числа xp_1, yp_1, xp_2, yp_2 - координаты точек, через которые проходит лазерный луч. Далее идут n строк, каждая из которых содержит описание одной летающей тарелки в формате x_i y_i r_i, где x_i, y_i - координаты центра, r_i - радиус тарелки. Все числа целые и не превосходят по модулю 10000. Радиусы летающих тарелок - целые и положительные. Летающие тарелки могут касаться и пересекаться.

Выходные данные

В первую строку выходного файла OUTPUT.TXT выведите количество уничтоженных летающих тарелок. Во вторую строку выведите номера уничтоженных летающих тарелок в возрастающем порядке. Тарелка

считается уничтоженной, если она имеет, хотя бы одну общую точку с лазерным лучом.

Пример

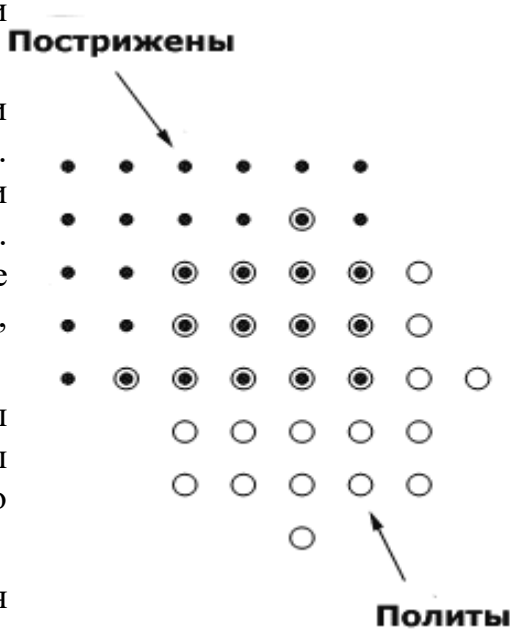
№	INPUT.TXT	OUTPUT.TXT
1	2 0 0 1 1 2 2 100 1000 1000 1	2 1 2

Газон

(Время: 1 сек. Память: 16 Мб Сложность: 36%)

Фермер Иван с юности следит за своим газоном. Газон можно считать плоскостью, на которой в каждой точке с целыми координатами растет один пучок травы.

В одно из воскресений Иван воспользовался газонокосилкой и постриг некоторый прямоугольный участок газона. Стороны этого участка параллельны осям координат, а две противоположные вершины расположены в точках (x1, y1) и (x2, y2). Следует отметить, что пучки травы, находящиеся на границе этого прямоугольника, также были пострижены.



Довольный результатом Иван купил и установил на газоне дождевальную установку. Она была размещена в точке с координатами (x3, y3) и имела радиус действия струи r. Таким образом, установка начала поливать все пучки, расстояние от которых до точки (x3, y3) не превышало r.

Все было хорошо, но Ивана заинтересовал следующий вопрос: сколько пучков травы оказалось и пострижено, и полито в это воскресенье?

Требуется написать программу, которая позволит дать ответ на вопрос Ивана.

Входные данные

Первая строка входного файла INPUT.TXT содержит четыре целых числа x1, y1, x2, y2 ($-100\,000 \le x1 < x2 \le 100\,000$; $-100\,000 \le y1 < y2 \le 100\,000$). Во второй строке записаны три целых числа x3, y3, r ($-100\,000 \le x3, y3 \le 100\,000$; $1 \le r \le 100\,000$)

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести одно целое число – число пучков травы, которые были и пострижены, и политы.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	0 0 5 4 4 0 3	14

Треугольник

(Время: 1 сек. Память: 16 Мб Сложность: 32%)

В декартовой системе координат на плоскости заданы координаты вершин треугольника и еще одной точки. Требуется написать программу, определяющую, принадлежит ли эта точка треугольнику.

Входные данные

В четырех строках входного файла INPUT.TXT находятся пары целых чисел - координаты точек. Числа в первых трех строках - это координаты вершин треугольника (x_1,y_1), (x_2,y_2), (x_3,y_3), в четвертой строке - координаты тестируемой точки (x_4,y_4). Все координаты не превышают 10000 по абсолютной величине.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести слово «In», если точка находится внутри треугольника и «Out» в противном случае.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	0 0 100 0 0 100 100 100	Out
2	0 0 100 0 0 100 10 10	In
3	0 0 100 0 0 100 50 50	In
4	0 0	In

	100 0 0 100 0 0	

Площадь многоугольника

(Время: 1 сек. Память: 16 Мб Сложность: 48%)

Многоугольник на плоскости задан целочисленными координатами своих N вершин в декартовой системе координат. Требуется найти площадь многоугольника. Стороны многоугольника не соприкасаются (за исключением соседних - в вершинах) и не пересекаются.

Входные данные

В первой строке входного файла INPUT.TXT находится число N. В следующих N строках находятся пары чисел (Xi,Yi) - координаты точек. Если соединить точки в данном порядке, а также первую и последнюю точки, получится заданный многоугольник. (3 <= N <= 50 000, -20 000 <= Xi,Yi <= 20 000)

Выходные данные

В выходной файл OUTPUT.TXT выведите одно число - площадь многоугольника. Его следует округлить до ближайшего числа с одной цифрой после запятой.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 5 0 0 5 -5 0 0 -5	50.0
2	4 0 4 0 0 3 0 1 1	3.5

Целые точки

(Время: 1 сек. Память: 16 Мб Сложность: 64%)

Многоугольник (не обязательно выпуклый) на плоскости задан координатами своих вершин. Требуется подсчитать количество точек с целочисленными координатами, лежащих внутри него (но не на его границе).

Входные данные

В первой строке входного файла INPUT.TXT содержится N ($3 \le N \le 10^3$) – число вершин многоугольника. В последующих N строках идут координаты (X_i, Y_i) вершин многоугольника в порядке обхода по часовой стрелке. X_i и Y_i - целые числа, по модулю не превосходящие 10^6.

Выходные данные

Ваша программа должна вывести в выходной файл OUTPUT.TXT одно целое число - ответ на задачу.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 -1 -1 -1 1 1 1 1 -1	1
2	3 0 0 0 2 2 0	0

3. Теория графов

Светофорчики

(Время: 1 сек. Память: 16 Мб Сложность: 25%)

В подземелье M тоннелей и N перекрестков, каждый тоннель соединяет какие-то два перекрестка. Мышиный король решил поставить по светофору в каждом тоннеле перед каждым перекрестком. Напишите программу, которая посчитает, сколько светофоров должно быть установлено на каждом из перекрестков. Перекрестки пронумерованы числами от 1 до N.

Входные данные

Во входном файле INPUT.TXT записано два числа N и M (0 < N <= 100, 0 <= M <= N*(N-1)/2). В следующих M строках записаны по два числа i и j (1

<= i,j <= N), которые означают, что перекрестки i и j соединены тоннелем. Можно считать, что любые два перекрестка соединены не более, чем одним тоннелем. Нет тоннелей от перекрестка i до него самого.

Выходные данные

В выходной файл OUTPUT.TXT вывести N чисел: k-ое число означает количество светофоров на k-ом перекрестке.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	7 10 5 1 3 2 7 1 5 2 7 4 6 5 6 4 7 5 2 1 5 3	3 3 2 2 5 2 3

Скачки

(Время: 1 сек. Память: 16 Мб Сложность: 32%)

Иван Иванович любит ходить на скачки, надеясь на них заработать кругленькую сумму. Ему приглянулась лошадь с номером K, и он решил проверить, сможет ли она выиграть у всех остальных лошадей. Иван Иванович раздобыл информацию, в которой для некоторых пар лошадей сообщается, какая из этих лошадей быстрее. Также он узнал, что у всех лошадей разные скорости.

Требуется написать программу, которая поможет Ивану Ивановичу точно определить может ли выиграть выбранная им лошадь.

Входные данные

Входной файл INPUT.TXT содержит в первой строке два целых числа N (1 <= N <= 100) и K (1 <= K <= N), где N – количество лошадей, принимающих участие в скачках, K – номер лошади, на которую хочет сделать ставку Иван Иванович. Следующие строки содержат по два числа X и Y (1 <= X, Y <= N), обозначающие, что лошадь с номером X быстрее лошади с номером Y. Пары X и Y не повторяются. Набор данных завершается строкой, содержащей единственный ноль. Эту строку обрабатывать не надо.

Гарантируется, что информация, раздобытая Иваном Ивановичем, корректна.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать слово «Yes», если Иван Иванович уверен в своем выигрыше и «No» в противном случае.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 1 1 2 1 3 0	Yes
2	3 2 2 3 0	No
3	4 2 3 1 2 3 0	No

Алгоритм Флойда

(Время: 1 сек. Память: 16 Мб Сложность: 36%)

Полный ориентированный взвешенный граф задан матрицей смежности. Постройте матрицу кратчайших путей между его вершинами. Гарантируется, что в графе нет циклов отрицательного веса.

Входные данные

В первой строке входного файла INPUT.TXT записано единственное число N (1 <= N <= 100) - количество вершин графа. В следующих N строках по N чисел - матрица смежности графа (j-ое число в i-ой строке соответствует весу ребра из вершины i в вершину j). Все числа по модулю не превышают 100. На главной диагонали матрицы - всегда нули.

Выходные данные

В выходной файл OUTPUT.TXT выведите N строк по N чисел - матрицу кратчайших расстояний между парами вершин. j-ое число в i-ой строке должно быть равно весу кратчайшего пути из вершины i в вершину j.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 0 5 9 100 100 0 2 8	0 5 7 13 12 0 2 8 11 16 0 7

	100 100 0 7 4 100 100 0	4 9 11 0

Домой на электричках

(Время: 1 сек. Память: 16 Мб Сложность: 40%)

Одна из команд-участниц олимпиады решила вернуться домой на электричках. При этом ребята хотят попасть домой как можно раньше. К сожалению, не все электрички идут от города, где проводится олимпиада, до станции, на которой живут ребята. И, что еще более обидно, не все электрички, которые идут мимо их станции, останавливаются на ней (равно как вообще, электрички останавливаются далеко не на всех станциях, мимо которых они идут).

Все станции на линии пронумерованы числами от 1 до N. При этом станция номер 1 находится в городе, где проводится олимпиада, и в момент времени 0 ребята приходят на станцию. Станция, на которую нужно попасть ребятам, имеет номер E.

Напишите программу, которая по данному расписанию движения электричек вычисляет минимальное время, когда ребята могут оказаться дома.

Входные данные

Во входном файле INPUT.TXT записаны сначала числа N (2 <= N <= 100) и E (2 <= E <= N). Затем записано число M (0 <= M <= 100), обозначающее число рейсов электричек. Далее идет описание M рейсов электричек. Описание каждого рейса электрички начинается с числа K_i (2 <= K_i <= N) — количества станций, на которых она останавливается, а далее следует Ki пар чисел, первое число каждой пары задает номер станции, второе — время, когда электричка останавливается на этой станции (время выражается целым числом из диапазона от 0 до 10^9). Станции внутри одного рейса упорядочены в порядке возрастания времени. В течение одного рейса электричка все время движется в одном направлении — либо от города, либо к городу.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно число — минимальное время, когда ребята смогут оказаться на своей станции. Если существующими рейсами электричек они добраться не смогут, выведите –1.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	5 3 4	20

	2 1 5 2 10 2 2 10 4 15 4 5 0 4 17 3 20 2 35 3 1 2 3 40 4 45	

Нить Ариадны

(Время: 1 сек. Память: 16 Мб Сложность: 40%)

Тезею из лабиринта Минотавра помог выйти клубок ниток. Вы можете вместо клубка использовать персональный компьютер.

Требуется написать программу, которая вводит маршрут Тезея в лабиринте и находит кратчайший обратный путь, по которому Тезей сможет выйти из лабиринта, не заходя в тупики и не делая петель.

Входные данные

Входной файл INPUT.TXT содержит маршрут Тезея, который представлен строкой, состоящей из букв: N, S, W, E и длиной не более 200.

Буквы означают:

N - один "шаг" на север,
S - один "шаг" на юг,
W - один "шаг" на запад,
E - один "шаг" на восток.

Выходные данные

В выходной файл OUTPUT.TXT записывается аналогично входному файлу найденный обратный путь. Если маршрут неоднозначен, то следует выбирать согласно следующему приоритету: N, E, S, W.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	EENNESWSSWE	NWW

Друзья

(Время: 1 сек. Память: 16 Мб Сложность: 41%)

В клубе N человек. Многие из них - друзья. Так же известно, что друзья друзей так же являются друзьями. Требуется выяснить, сколько всего друзей у конкретного человека в клубе.

Входные данные

В первой строке входного файла INPUT.TXT заданы два числа: N и S (1 <= N <= 100; 1 <= S <= N), где N - количество человек в клубе, а S – номер конкретного человека. В следующих N строках записано по N чисел - матрица смежности, состоящая из единиц и нулей. Причем единица, стоящая

в i-й строке и j-м столбце гарантирует, что люди с номерами i и j – друзья, а 0 – выражает неопределенность.

Выходные данные

В выходной файл OUTPUT.TXT выведите количество гарантированных друзей у человека с номером S, помня о транзитивности дружбы.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 1 0 1 0 1 0 1 0 1 0	2

Lines

(Время: 1 сек. Память: 16 Мб Сложность: 44%)

В таблице из N строк и N столбцов некоторые клетки заняты шариками, другие свободны. Выбран шарик, который нужно переместить, и место, куда его нужно переместить. Выбранный шарик за один шаг перемещается в соседнюю по горизонтали или вертикали свободную клетку. Требуется выяснить, возможно ли переместить шарик из начальной клетки в заданную, и, если возможно, то найти путь из наименьшего количества шагов.

Входные данные

В первой строке входного файла INPUT.TXT находится число N, в следующих N строках - по N символов. Символом точки обозначена свободная клетка, латинской заглавной O - шарик, @ - исходное положение шарика, который должен двигаться, латинской заглавной X - конечное положение шарика. (2 <= N <= 50)

Выходные данные

В выходной файл OUTPUT.TXT выведите в первой строке Yes, если движение возможно, или No, если нет. Если движение возможно, то далее следует вывести N строк по N символов - как и на вводе, но букву X, а также все точки по пути следует заменить плюсами. Если решений несколько, выведите любое.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	5 ....X .OOOO ..... OOOO.	Yes +++++ +OOOO +++++ OOOO+

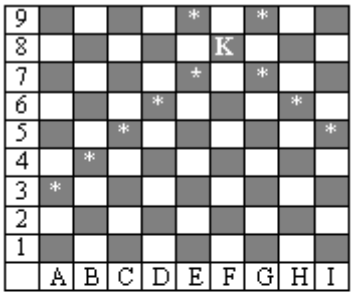
	@.... 	@++++
2	5 ..X.. ..... OOOOO ....@..	No

Мифические шахматы

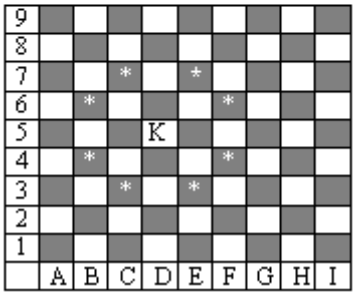
(Время: 1 сек. Память: 16 Мб Сложность: 45%)

Ваш друг Вася занимается разработкой компьютерной игры «Мифические шахматы». Он не укладывается в установленные сроки сдачи проекта.

Вася обратился к друзьям за помощью. Ему необходим модуль, вычисляющий оптимальные пути перемещения фигур из одной клетки в другую. Так как друзей у Васи много, то каждому досталась маленькая подзадача. Вам требуется написать программу, определяющую минимальное количество ходов, необходимое кентавру, чтобы добраться из одной клетки в другую.



В мифические шахматы играют на шахматной доске размером 9x9, угловые клетки которой окрашены в черный цвет. Кентавр – фигура мифических шахмат, объединяющая в себе свойства коня и слона. Когда кентавр стоит на белой клетке, он может ходить только как конь, а когда на черной – только как слон. На рисунках приведены варианты ходов для двух кентавров (буквой «K» отмечено местоположение кентавра, а звездочками – клетки, куда кентавр может сделать ход).



Входные данные

В первой строке входного файла INPUT.TXT содержатся координаты (большая латинская буква и цифра) двух клеток доски для мифических шахмат, разделенные пробелом.

Выходные данные

В выходной файл OUTPUT.TXT выведите минимальное число ходов, необходимое кентавру, чтобы добраться из первой клетки во вторую. Если добраться невозможно, то следует вывести число «-1» (без кавычек).

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	H6 E5	2

2	A6 F6	3

Алгоритм Дейкстры

(Время: 1 сек. Память: 16 Мб Сложность: 47%)

Дан ориентированный взвешенный граф. Для него вам необходимо найти кратчайшее расстояние от вершины S до вершины F.

Входные данные

В первой строке входного файла INPUT.TXT записаны три числа: N, S и F (1 <= N <= 100; 1 <= S, F <= N), где N - количество вершин графа. В следующих N строках записаны по N чисел - матрица смежности графа, где число в i-ой строке j-ом столбце соответствует ребру из i в j: -1 означает отсутствие ребра между вершинами, а любое неотрицательное целое число (от 0 до 100) - наличие ребра данного веса . На главной диагонали матрицы всегда записаны нули.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести искомое расстояние или -1, если пути между указанными вершинами не существует.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 1 2 0 -1 2 3 0 -1 -1 4 0	6

Грядки

(Время: 1 сек. Память: 16 Мб Сложность: 47%)

Прямоугольный садовый участок шириной N и длиной M метров разбит на квадраты со стороной 1 метр. На этом участке вскопаны грядки. Грядкой называется совокупность квадратов, удовлетворяющая таким условиям:

- из любого квадрата этой грядки можно попасть в любой другой квадрат этой же грядки, последовательно переходя по грядке из квадрата в квадрат через их общую сторону;
- никакие две грядки не пересекаются и не касаются друг друга ни по вертикальной, ни по горизонтальной сторонам квадратов (касание грядок углами квадратов допускается).

Подсчитайте количество грядок на садовом участке.

Входные данные

В первой строке входного файла INPUT.TXT находятся числа N и M через пробел, далее идут N строк по M символов. Символ # обозначает территорию грядки, точка соответствует незанятой территории. Других символов в исходном файле нет. (1 <= N, M <= 200)

Выходные данные

В выходной файл OUTPUT.TXT выведите количество грядок на садовом участке.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	5 10 ##.....#. .#..#...#. .###....#. ..##....#. ....#.	3
2	5 10 ##..#####. .#.#.#.... ###..##.#. ..##.....# .###.#####	5

Заправки

(Время: 1 сек. Память: 16 Мб Сложность: 49%)

В стране N городов, некоторые из которых соединены между собой дорогами. Для того, чтобы проехать по одной дороге требуется один бак бензина. В каждом городе бак бензина имеет разную стоимость. Вам требуется добраться из первого города в N-ый, потратив как можно меньшее количество денег.

Входные данные

Во входном файле INPUT.TXT записано сначала число N (1<=N<=100), затем идет N чисел, i-ое из которых задает стоимость бензина в i-ом городе (все числа целые из диапазона от 0 до 100). Далее идет число M - количество дорог в стране, далее идет описание самих дорог. Каждая дорога задается двумя числами - номерами городов, которые она соединяет. Все дороги двухсторонние (то есть по ним можно ездить как в одну, так и в другую сторону); между двумя городами всегда существует не более одной дороги; не существует дорог, ведущих из города в себя.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно число - суммарную стоимость маршрута или -1, если добраться невозможно.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 1 10 2 15 4 1 2 1 3 4 2 4 3	3

Лоскутки

(Время: 1 сек. Память: 16 Мб Сложность: 50%)

Вася Пупкин взял листочек в клетку и начал его резать по определённым линиям. На запасном листке такого же размера он закрасил клетки, по которым проходили линии. Василий Васильевич так увлёкся этим занятием, что запутался, сколько частей от листа у него осталось. Ваша задача найти это число.

Входные данные

Во входном файле INPUT.TXT в первой строке записаны N и M (0 < N, M <= 100) – размерность матрицы. Далее записана матрица из N строк, каждая из которых содержит M нулей и единиц. 0 обозначает не закрашенную клетку и 1 – закрашенную (линию разреза).

Выходные данные

В выходной файл OUTPUT.TXT следует вывести количество оставшихся частей листа.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 4 0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 1	2

Сон

(Время: 1 сек. Память: 16 Мб Сложность: 50%)

Любознательный школьник Петя очень любит программировать. Однажды он настолько увлекся этим делом, что уснул прямо за компьютером! Пете приснилось, что он попал в альтернативную реальность. Альтернативная реальность представляет собой прямоугольный лабиринт, который можно изобразить в виде таблицы размером R × C клеток. Чтобы не опоздать школу, Пете нужно найти выход из лабиринта. Начальная позиция Пети обозначается символом ‘S’, выход из альтернативной реальности – ‘E’. За один ход Петя перемещается в одну из четырех смежных клеток (влево, вправо, вниз, вверх). Если клетка занята стеной (символ ‘X’), то Петя пройти в нее не может. В некоторых клетках расположены двери с замками одного из четырех цветов (‘R’, ‘G’, ‘B’, ‘Y’). Для прохода в эту клетку, необходимо обладать ключом определенного цвета. Так как ключи многоразовые, то одним ключом можно открыть сколь угодно много соответствующих ему замков.

Властелин альтернативной реальности предлагает Пете купить ключи, чтобы пройти к выходу. У нашего героя совсем немного денег с собой, поэтому ему хочется потратить как можно меньше денег и при этом пройти к выходу. Помогите ему определить минимальную сумму денег, которую нужно потратить на ключи.

Входные данные

Первая строка входного файла INPUT.TXT содержит натуральные числа R и C (R,C <=50). Вторая строка теста содержит 4 целых числа P_i, стоимости покупки ключей ‘R’, ‘G’, ‘B’ и ‘Y’ соответственно (P_i <= 100). Далее следуют R строк, в каждой из которых C символов, описывающих лабиринт. Каждый лабиринт содержит только один символ ‘S’ и один символ ‘E’.

Выходные данные

В выходной файл OUTPUT.TXT выведите минимальную сумму денег, необходимую для покупки набора ключей, позволяющего пройти к выходу. Если пути к выходу из альтернативной реальности не существует, и Петя обречен проспать уроки, выведите «Sleep».

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 7 1 1 1 1 XXXXXXX XS.X.EX XXXXXXX	Sleep
2	6 6 1 5 3 1 XXXXXX XS.X.X X..R.X	4

	X.XXBX X.G.EX XXXXXX	

Побег с космической станции

(Время: 1 сек. Память: 16 Мб Сложность: 52%)

Представьте, что вы состоите на службе во внешней разведке Межгалактического Альянса Республиканских Сил (МАРС). Одному из агентов разведки крупно не повезло, и он был захвачен на засекреченной космической базе. К счастью, внешней разведке МАРС удалось заполучить план этой базы. И вот теперь вам поручено разработать план побега.

База представляет собой прямоугольник размером NxM, со всех сторон окружённый стенами, и состоящий из квадратных отсеков единичной площади. База снабжена K выходами, до одного из которых агенту необходимо добраться. В некоторых отсеках базы находятся стены. Ваш агент может перемещаться из отсека в любой из четырех соседних с ним, если в том отсеке, куда он хочет переместиться, нет стены.

Кроме того, база снабжена системой гипертуннелей, способных перемещать агента из одного отсека базы (вход в гипертуннель) в другой (выход из гипертуннеля). Когда агент находится в отсеке, где есть вход в гипертуннель, он может (но не обязан) им воспользоваться.

Начальное положение вашего агента известно. Вам необходимо найти кратчайший путь побега (то есть путь, проходящий через минимальное количество отсеков).

Входные данные

В первой строке входного файла INPUT.TXT записаны числа N и M (2<=N<=100, 2<=M<=100), задающие размеры базы: N — количество строк в плане базы, M — количество столбцов. Во второй строке записаны начальные координаты агента X_A,Y_A (1<=X_A<=N, 1<=Y_A<=M). Первая координата задает номер строки, вторая — номер столбца. Строки нумеруются сверху вниз, столбцы слева направо. Далее следуют N строк по M чисел, задающих описание стен внутри базы: 1 соответствует стенке, 0 — её отсутствию. Далее в отдельной строке записано число H (0<=H<=1000) — количество гипертуннелей. В последующих H строках идут описания гипертуннелей. Каждый гипертуннель задается 4 числами: X_1, Y_1, X_2, Y_2 (1<=X_1,X_2<=N; 1<=Y_1,Y_2<=M) — координатами входа и выхода гипертуннеля. Никакие два гипертуннеля не имеют общего входа. После этого в отдельной строке следует число K (1<=K<=10) — количество выходов с базы. В последующих K строках идут описания выходов с базы. Каждый выход задается двумя координатами X и Y (1<=X<=N; 1<=Y<=M).

Гарантируется, что начальные координаты агента не совпадают ни с одним из выходов и он не стоит в отсеке, занятом стеной. Никакие входы и

выходы гипертуннелей, а также выходы с базы не находятся в отсеках, занятых стенами. Никакой вход в гипертуннель не совпадает с выходом с базы.

Выходные данные

Если побег невозможен, выведите в выходной файл OUTPUT.TXT "Impossible". В противном случае следует вывести количество отсеков в кратчайшем пути побега.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 5 2 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 2 1 2 1 4 3 1 1 4 1 2 4	4

4. Динамическое программирование

Волосатый бизнес

(Время: 1 сек. Память: 16 Мб Сложность: 32%)

Одного неформала выгнали с работы, и теперь ему надо как-то зарабатывать себе на пиво и сигареты. Поразмыслив, он решил, что сможет иметь очень неплохие деньги на продаже собственных волос. Известно, что пункты приема покупают волосы произвольной длины стоимостью C у.е. за каждый сантиметр. Так как волосяной рынок является очень динамичным, то цена одного сантиметра волос меняется каждый день как и курс валют. Неформал является очень хорошим бизнес-аналитиком. Он смог вычислить, какой будет цена одного сантиметра волос в каждый из ближайших N дней (для удобства пронумеруем дни в хронологическом порядке от 0 до N-1). Теперь он хочет определить, в какие из этих дней ему следует продавать волосы, чтобы по истечению всех N дней заработать максимальное количество денег. Заметим, что волосы у неформала растут только ночью и

вырастают на 1 сантиметр за ночь. Следует также учесть, что до 0-го дня неформал с горя подстригся наголо и к 0-му дню длина его волос составляла 1 сантиметр.

Входные данные

В первой строке входного файла INPUT.TXT записано целое число N ($0 < N \le 100$). Во второй строке через пробел заданы N натуральных чисел, не превосходящих 100, соответствующие стоимости C[i] 1 сантиметра волос за каждый i-й день.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести максимальную денежную сумму, которую может заработать неформал за N дней.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	5 73 31 96 24 46	380
2	10 1 2 3 4 5 6 7 8 9 10	100
3	10 10 9 8 7 6 5 4 3 2 1	55

Только вправо или вниз

(Время: 1 сек. Память: 16 Мб Сложность: 32%)

Игровое поле N x M заполняется целыми числами, одно неотрицательное целое число в каждой клетке. Цель игры состоит в том, чтобы пройти по любому разрешенному пути от верхнего левого угла до правого нижнего. Целое число в каждой клетке указывает, какой длины шаг должен быть из текущей клетки. Все шаги могут быть или направо или вниз. Если в результате какого-либо шага игрок покидает пределы поля, такой шаг запрещается.

На рис. 1 приведен пример игрового поля 3 x 4, где сплошная окружность показывает положение начала, а пунктирная окружность – цель. Рис. 2 показывает три возможных пути от начала до цели для рассматриваемого примера игрового поля, с удаленными промежуточными числами.

(2)	1	1	2
3	2	1	44
3	1	1	(0)

Рис. 1.

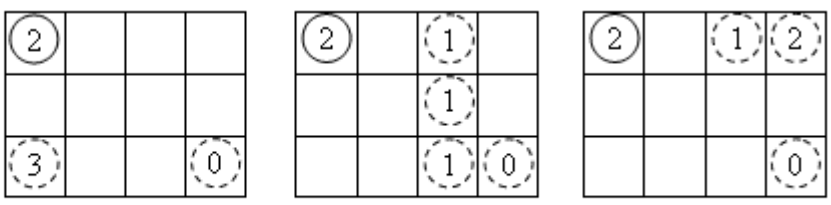


Рис. 2.

Требуется написать программу, которая определит число различных вариантов путей от верхнего левого угла до правого нижнего.

Входные данные

Входной файл INPUT.TXT содержит в первой строке размеры поля N ($1 \le N \le 70$) и M ($1 \le M \le 70$). В последующих N строках входного файла, каждая из которых описывает отдельную строку игрового поля, записаны через пробел по M целых чисел – длины шагов из клеток данной строки.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одно число - число различных вариантов путей от верхнего левого угла до правого нижнего. Для каждого поля будет менее чем 2^{31} различных путей.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 4 2 1 1 2 3 2 1 44 3 1 1 0	3

Гвоздики

(Время: 1 сек. Память: 16 Мб Сложность: 34%)

На прямой дощечке вбиты гвоздики. Любые два гвоздика можно соединить ниточкой. Требуется соединить некоторые пары гвоздиков ниточками так, чтобы к каждому гвоздику была привязана хотя бы одна ниточка, а суммарная длина всех ниточек была минимальна.

Входные данные

В первой строке входного файла INPUT.TXT записано число N - количество гвоздиков (2 <= N <= 100). В следующей строке записано N чисел - координаты всех гвоздиков (неотрицательные целые числа, не превосходящие 10000).

Выходные данные

В выходной файл OUTPUT.TXT нужно вывести единственное число - минимальную суммарную длину всех ниточек.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	6 3 4 12 6 14 13	5

Без двух нулей подряд

(Время: 1 сек. Память: 16 Мб Сложность: 37%)

Требуется вычислить количество N-значных чисел в системе счисления с основанием K, таких что их запись не содержит двух подряд идущих нулей.

Входные данные

Во входном файле INPUT.TXT записаны два натуральных числа N и K в десятичной системе счисления ($2 \le K \le 10$; $2 \le N$; $4 \le N+K \le 18$).

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести целое число в десятичной записи – ответ на задачу.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2 10	90
2	4 2	5
3	6 3	328

Есть ли жизнь на Марсе?

(Время: 1 сек. Память: 16 Мб Сложность: 37%)

- Ты врешь, Коля! На Марсе жизни нет! Кто тебе такую чушь сказал?

- Петя. А ему сказал Саша.

- Да от Пети я в жизни правдивого слова не слышал! Ему что ни скажут, он все переврет. Да и Саше откуда знать?

- А Саше рассказал про это Владимир Алексеевич, наш учитель биологии.

- Ну, Владимиру Алексеевичу-то можно верить... Только вряд ли он так сказал, это либо Саша, либо Петя придумал. А может, это ты меня разыгрываешь?..

- Минуточку, ребята, - вмешался подошедший к спорящим учитель математики, Глеб Тимофеевич, - давайте подойдем к проблеме формально. Допустим, что все диалоги - Владимира Алексеевича с Сашей, Саши с Петей и Пети с Колей - действительно имели место. Пронумеруем ребят числами 1, 2 и 3. Предположим также, что каждый из ребят независимо друг от друга передал полученную информацию относительно жизни на Марсе верно с вероятностью p_i, а соврал с вероятностью $q_i = 1 - p_i$ для $i = 1, 2, 3$. Вероятности – это вещественные числа от нуля до единицы; событие, имеющее вероятность 0, никогда не произойдет, событие же с вероятностью 1 произойдет без всякого сомнения. Зная, что Коля после этого объявил, что жизнь на Марсе все-таки есть, найдите по данным p_i вероятность того, что так действительно сказал Владимир Алексеевич.

- А как искать эту вероятность? И что значит независимо друг от друга? – растерялись ребята.

- Независимость означает, что действие одного из ребят никак не отражается на том, как поступят другие. К примеру, Пете неважно, соврал ли Саша - в любом случае он передаст сказанное Сашей правильно с вероятностью ровно p_2. Задача несложная, и можно рассмотреть все восемь возможных случаев. Первый случай - все ребята говорили правду, и вероятность этого случая равна $p_1 \cdot p_2 \cdot p_3$. В этом случае жизнь на Марсе, без сомнения, есть - Владимиру Алексеевичу мы верим, а ребята передали его слова правильно. Второй случай, когда соврал только Саша, имеет место с вероятностью $q_1 \cdot p_2 \cdot p_3$, и в этом случае жизни на Марсе нет. Далее переберем остальные шесть случаев, каждый раз перемножая соответствующие вероятности, а потом просуммируем вероятности тех случаев, в которых слова учителя переданы правильно. То, что вероятности для отдельных ребят в каждом случае надо перемножить - это и есть формальное определение независимости. Ну, в скольких случаях будет передано именно то, что говорил Владимир Алексеевич?

- В одном …

- А вот и нет. Например, если Петя и Коля соврали, а Саша сказал правду, то истина, дважды исказившись, дойдет до нас в неизменном виде. И вообще, четное количество отрицаний, примененных к утверждению, дает само утверждение. В нашей задаче случаев с четным количеством отрицаний - четыре, и итоговая вероятность равна $p_1 \cdot p_2 \cdot p_3 + q_1 \cdot q_2 \cdot p_3 + q_1 \cdot p_2 \cdot q_3 + p_1 \cdot q_2 \cdot q_3$.

- То есть если Петя и Коля точно соврут, а Саша точно скажет правду, то от Коли мы услышим в точности то, что говорил учитель?

- Совершенно верно. А теперь решите-ка задачу для общего случая, когда ребят не трое, а n. Первому, кто решит - пятерка на следующей контрольной!

Входные данные

Входной файл INPUT.TXT содержит целое число n ($1 <= n <= 100$). Во второй строке через пробел записаны n вещественных чисел - это числа p_1, p_2, ..., p_n ($0 <= p_i <= 1$). Числа даны с не более чем шестью десятичными знаками после запятой.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно вещественное число, округленное до шести знаков после запятой - вероятность существования жизни на Марсе.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3	0.18

	1 0.1 0.9	

Компьютерная игра

(Время: 1 сек. Память: 16 Мб Сложность: 38%)

Вы можете вспомнить хоть одного своего знакомого до двадцатилетнего возраста, который в детстве не играл в компьютерные игры? Если да, то может быть вы и сами не знакомы с этим развлечением? Впрочем, трудностей при решении этой задачи это создать не должно.

Во многих старых играх с двумерной графикой можно столкнуться с подобной ситуацией. Какой-нибудь герой прыгает по платформам (или островкам), которые висят в воздухе. Он должен перебраться от одного края экрана до другого. При этом при прыжке с одной платформы на соседнюю, у героя уходит |y2-y1| единиц энергии, где y1 и y2 – высоты, на которых расположены эти платформы. Кроме того, у героя есть суперприем, который позволяет перескочить через платформу, но на это затрачивается 3*|y3-y1| единиц энергии. Конечно же, энергию следует расходовать максимально экономно.

Предположим, что вам известны координаты всех платформ в порядке от левого края до правого. Сможете ли вы найти, какое минимальное количество энергии потребуется герою, чтобы добраться с первой платформы до последней?

Входные данные

В первой строке входного файла INPUT.TXT записано количество платформ n ($1 \le n \le 30000$). Вторая строка содержит n натуральных чисел, не превосходящих 30000 – высоты, на которых располагаются платформы.

Выходные данные

В выходной файл OUTPUT.TXT запишите единственное число – минимальное количество энергии, которую должен потратить игрок на преодоление платформ (конечно же в предположении, что cheat-коды использовать нельзя).

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 1 5 10	9
2	3 1 5 2	3

Максимальная подпоследовательность

(Время: 1 сек. Память: 16 Мб Сложность: 38%)

Дана числовая последовательность, требуется найти длину наибольшей возрастающей подпоследовательности.

Входные данные

В первой строке входного файла INPUT.TXT записано число N - длина последовательности (1 <= N <= 1000). Во второй строке записана сама последовательность (через пробел). Числа последовательности - целые числа, не превосходящие 10000 по модулю.

Выходные данные

В выходной файл OUTPUT.TXT требуется вывести наибольшую длину возрастающей подпоследовательности.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	6 3 29 5 5 28 6	3

Лягушонок

(Время: 1 сек. Память: 16 Мб Сложность: 39%)

Многие, вероятно, слышали песни про приключения лягушонка Crazy Frog. На этот раз неугомонное милое создание решило подкрепиться, но даже такое простое действие решило выполнить в виде игры. Итак, в каждой клетке квадратного игрового поля, разбитого на N*N (N<=50) клеток, находится один комар весом a_{ij} (вес комара – натуральное число <= 50), i - номер строки, j - номер столбца. Лягушонок, прыгая с клетки на клетку, ест комаров. Правила игры таковы - в каждом столбце можно съесть не более одного комара. Всякий раз при съедании комара запоминаем номер строки, откуда съеден комар, и сумма номеров строк, в которых были съедены комары, в конце игры должна быть в точности равна N. Учтите, если из какой-то строки съедено несколько комаров, то номер данной строки участвует в суммировании более одного раза.

Определите максимальный вес комаров, который можно съесть при следовании приведённым правилам.

Входные данные

Первая строка входного файла INPUT.TXT содержит число N. Следующие N строк содержат по N чисел a_{ij}, разделенных пробелами.

Выходные данные

В выходной файл OUTPUT.TXT выведите целое число – вес съеденных комаров.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 8 2 1 1 2 6 2 7 2	14
2	5 8 2 1 2 3 1 2 6 2 4 2 7 2 3 4 1 3 2 4 4 1 3 4 3 1	19

Энты

(Время: 1 сек. Память: 16 Мб Сложность: 39%)

Энты были созданы в Первоначальную эпоху вместе с другими обитателями Средиземья. Эльфийские легенды гласят, что когда Варда зажгла звёзды и пробудились Эльфы, вместе с ними пробудились и Энты в Великих Лесах Арды.

Когда Энты пришли в Арду, они ещё не умели говорить — этому искусству их обучали Эльфы, и Энтам это ужасно нравилось. Им доставляло удовольствие изучать разные языки, даже щебетание Людей.

Эльфы выработали хорошую технику обучения энтов своему языку. Первый энт, которого обучили эльфы, выучил всего два слова — «tancave» (да) и «la» (нет). Обученный энт выбрал одного старого и одного молодого энта, не умеющих говорить, и обучил их всем словам, которые знал сам. Затем обучение этих двух энтов продолжили сами эльфы. Каждый обучившийся у эльфов энт снова выбирал из неговорящих сородичей одного старого и одного молодого, обучал их всем словам, которые знал, передавал эльфам и так далее.

Выяснилось, что более молодые энты выучивали у эльфов ещё ровно столько же слов, сколько они узнали от обучавшего их энта. А вот более старые, уже склонные к одеревенению энты, пополняли свой запас всего лишь одним словом. После обучения у эльфов энты до конца света уже не могли выучить ни одного нового слова.

Общее число энтов в Средиземье больше, чем вы думаете. Интересно, а сколько из них знают ровно 150 квенийских слов? Похожую задачу вам предстоит решить.

Входные данные

Входной файл INPUT.TXT содержит натуральные числа K и P ($K <= 10^6$; $1 <= P <= 10^9$), записанные через пробел.

Выходные данные

Мы понимаем, что число энтов, знающих в точности K слов, может быть слишком велико, поэтому просим вывести в выходной файл OUTPUT.TXT лишь количество энтов, знающих ровно K слов, по модулю P.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 10	2
2	8 10	5
3	360 1000	179

Китайские часы

(Время: 1 сек. Память: 16 Мб Сложность: 41%)

Русский бизнесмен Иван Петров закупил в Китае большую партию наручных часов, чтобы продать их на родине за полцены (т.е. в 5 раз дороже, чем они стоили в Китае). Иван столкнулся с проблемой: китайские часы оказались некачественными. Мало того, что часы работали на протяжении всего нескольких часов, пока их не стукнешь, так еще и время подводить неудобно: вращать можно не минутную, а только секундную стрелку, причем, что самое ужасное, только в одну сторону в направлении увеличении времени. Например, для того, чтобы подвести часы на секунду назад, необходимо было сделать более 700 полных оборотов секундной стрелки, на что Иван бы потратил более 10 минут.

Чтобы продать эти часы оптом Ивану необходимо на момент сделки создать видимость того, что часы исправны. Для этого он собирается остановить все часы, установить их на одно и то же время. А перед сделкой ударить по чемодану с часами, чтобы они все дружно пошли.

Помогите Ивану выяснить: какое время на часах лучше установить для того, чтобы Иван потратил как можно меньше времени для того, чтобы подвести все часы.

Входные данные

В первой строке входного файла INPUT.TXT содержится натуральное число N – количество часов ($N \le 50000$). В последующих N строках располагаются показания всех часов в формате h:mm:ss, где h – показывает который час, mm – минуты, ss - секунды ($1 \le h \le 12$, $0 \le mm \le 59$, $0 \le ss \le 59$).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать время, которое нужно установить на всех часах, в формате, указанном выше. В случае неоднозначного ответа выведите наименьшее время.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 8:19:16 2:05:11 12:50:07	2:05:11

Фермер

(Время: 1 сек. Память: 16 Мб Сложность: 46%)

Фермер решил на своем квадратном участке земли вспахать пашню квадратной формы максимальной площади, т.к. он посчитал, что именно квадратная форма пашни наиболее удобна для обработки. Но на его участке присутствуют деревья и хозяйственные постройки, которые он никуда не хочет переносить, а так же иные места, не пригодные для пашни. Для удобства он составил квадратную карту местности NxN в форме матрицы и пометил нулями непригодные для пашни зоны, в остальные зоны он поставил единицу.

Необходимо помочь фермеру определить максимальную площадь пашни.

Входные данные

В первой строке входного файла INPUT.TXT записано единственное натуральное число N ($1 \le N \le 1000$) – длина стороны квадратного участка фермы. Далее, следует N строк, в каждой из которых находится последовательность (без пробелов) нулей и единиц, описывающих ферму.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести максимально возможную площадь пашни.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	7 1101101 1111110 1011100 0011100 1000010 1100111 1001110	9

Автомобильные пробки

(Время: 1 сек. Память: 16 Мб Сложность: 48%)

Автомобильные пробки случаются везде, даже в нашем небольшом городе. Дороги у нас имеют по две полосы в одном направлении, а автомобили только двух видов: легковые (в пробке занимают квадратное место 1x1 от ширины одной полосы) и грузовые (занимают прямоугольное место 1x2). Автомобилисты очень дисциплинированы: не становятся поперек полосы, не занимают чужую площадь, но и не оставляют свободных мест.

Требуется написать программу, которая определит количество различных автомобильных пробок длины N.

Входные данные

Входной файл INPUT.TXT содержит одно натуральное число N (N <= 1000).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать найденное количество автомобильных пробок.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2	4
2	3	9

Атлеты

(Время: 3 сек. Память: 16 Мб Сложность: 50%)

Художественная гимнастика – это вид спорта, где всё познаётся в сравнении, здесь нельзя, как в беге или плавании, измерить результат спортсмена с точностью до сотой доли секунды. Поэтому на выступлениях оценки выступлениям дают судьи. При выставлении оценок судьи ориентируются не только на текущее выступление, но, безусловно, сравнивают текущее выступление с выступлениями, показанными ранее. Кроме того, учитывается сложность показанного упражнения и рейтинг спортсмена.

Каждый выход спортсмена описывается двумя числами: номер спортсмена в рейтинге (наиболее профессиональные спортсмены имеют наибольший номер), и номер исполненного упражнения (упражнения нумеруются, начиная с самых простых). Судья сравнивает каждый выход спортсмена с каждым из выполненных ранее выходов. Если в результате сравнения получается, что спортсмен с большим номером показал более простое упражнение, чем спортсмен с меньшим номером, судья удивляется. Следует учитывать, что один выход может удивить судью несколько раз. Один спортсмен может выполнить несколько выходов, так же, как и одно упражнение может быть показано несколькими спортсменами – но такие выходы в сравнении судью не удивляют. Спортсмен не исполняет уже

показанное упражнение повторно. Требуется подсчитать, сколько раз за время выступлений будет удивлён судья.

Входные данные

Первая строка входного файла INPUT.TXT содержит количество спортсменов N ($0 < N \le 250$), количество упражнений M ($0 < M \le 250$) и количество выходов P. Следующие P строк содержат по два числа, описывающие выход спортсмена – номер спортсмена и номер упражнения.

Выходные данные

В выходной файл OUTPUT.TXT выведите, сколько раз был удивлен судья.

Примеры

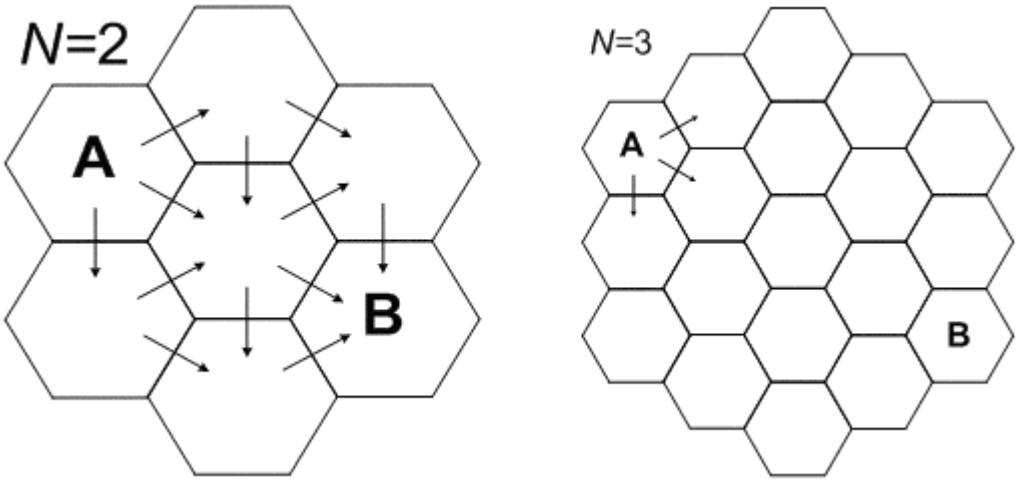
№	INPUT.TXT	OUTPUT.TXT
1	3 3 4 3 1 1 2 1 3 3 2	3
2	2 2 2 1 1 2 2	0

Пчелка

(Время: 1 сек. Память: 16 Мб Сложность: 53%)

Представьте себе пчелиные соты – поле из шестиугольных клеток со стороной N. В верхней левой клетке A находится пчелка. За один ход она может переползти на клетку вниз, на клетку вниз-вправо или на клетку вверх-вправо (вверх и влево пчелка не ползает).

Требуется написать программу, которая найдет количество способов, которыми пчелка может доползти из клетки A в противоположную клетку B.



Входные данные

Входной файл INPUT.TXT содержит единственное число N – размеры шестиугольного поля (2 <= N <= 12).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать единственное целое число – количество способов.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2	11
2	3	291

Фотограф-зануда

(Время: 1 сек. Память: 16 Мб Сложность: 55%)

Однажды глава семейства заказал фотографию своей большой семьи, состоящей из N человек, возраст которых 1 год, 2 года, …, N-1 лет и N лет. На фотографии должны присутствовать все родственники, и для этого они должны расположиться в один ряд. Сначала было решено расположить родственников по старшинству, начиная с самого младшего. Но фотограф сказал, что, возможно, на фото это будет выглядеть неестественно. Тогда было решено использовать следующее размещение:

1. слева сидит ребенок возрастом в 1 год
2. разность возрастов двух соседних родственников не превышает 2 года

Действительно, на фотографии, таким образом, все будут все равно выглядеть, будто расположенные по старшинству (ведь среди людей возрастом, к примеру, 25 и 27 лет не так легко определить старшего). Способов такой посадки существует, понятно, несколько. Фотограф снял все такие способы. Сколько же фотографий получилось в итоге?

Входные данные

Во входном файле INPUT.TXT содержится число N (1 <= N <= 55) – количество членов большой семьи.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать искомое число фотографий.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4	4

2	7	14

Фермер - 2

(Время: 1 сек. Память: 16 Мб Сложность: 60%)

После решения задачи с пашней земли, фермер хочет построить на этой земле как можно больший по площади сарай прямоугольной формы. Но на его участке есть деревья и хозяйственные постройки, которые он не хочет никуда переносить. Для простоты представим ферму прямоугольной сеткой размера MxN. Каждое из деревьев и построек размещается в одном или нескольких узлах сетки. Сарай должен быть построен на свободных узлах сетки.

Помогите фермеру определить максимально возможную площадь сарая.

Входные данные

В первой строке входного файла INPUT.TXT записаны два натуральных числа N и M ($1 \le N,M \le 1000$) – размеры фермы. Далее, следует N строк, в каждой из которых находится последовательность (без пробелов) из M нулей и единиц, описывающих ферму. Единицы соответствуют свободным для постройки участкам.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести максимально возможную площадь сарая, который может построить фермер на своем участке.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	5 10 1011011111 0111111110 1111111111 1011111111 1101110111	21

Школа танцев

(Время: 1 сек. Память: 64 Мб Сложность: 62%)

В школу бальных танцев профессора Падеграса записались n учеников — мальчиков и девочек. Профессор построил их в один ряд, и хочет отобрать из них для первого занятия группу стоящих подряд учеников, в которой количество мальчиков и девочек одинаково. Сколько вариантов выбора есть у профессора?

Входные данные

В первой строке входного файла INPUT.TXT задано число n ($1 \le N \le 10^6$). Во второй строке задается описание построенного ряда из мальчиков и девочек — строка из n символов a и b (символ a соответствует девочке, а символ b — мальчику).

Выходные данные

В единственной строке выходного файла OUTPUT.TXT должно содержаться единственное число — количество вариантов выбора требуемой группы.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 bab	2
2	8 abbababa	13

Симпатичные узоры

(Время: 1 сек. Память: 16 Мб Сложность: 66%)

Компания BrokenTiles планирует заняться выкладыванием во дворах у состоятельных клиентов узор из черных и белых плиток, каждая из которых имеет размер 1x1 метр. Известно, что дворы всех состоятельных людей имеют наиболее модную на сегодня форму прямоугольника MxN метров.

Однако при составлении финансового плана у директора этой организации появилось целых две серьезных проблемы: во первых, каждый новый клиент очевидно захочет, чтобы узор, выложенный у него во дворе, отличался от узоров всех остальных клиентов этой фирмы, а во вторых, этот узор должен быть симпатичным. Как показало исследование, узор является симпатичным, если в нем нигде не встречается квадрата 2x2 метра, полностью покрытого плитками одного цвета. На рисунке 1 показаны примеры различных симпатичных узоров, а на рисунке 2 – несимпатичных.

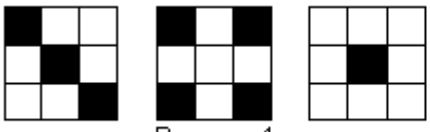


Рисунок 1

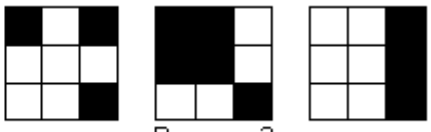


Рисунок 2

Для составления финансового плана директору необходимо узнать, сколько клиентов он сможет обслужить, прежде чем симпатичные узоры данного размера закончатся. Помогите ему!

Входные данные

В первой строке входного файла INPUT.TXT находятся два положительных целых числа, разделенные пробелом – M и N ($1 \le M \cdot N \le 30$).

Выходные данные

Выведите в выходной файл OUTPUT.TXT единственное число – количество различных симпатичных узоров, которые можно выложить во дворе размера MxN. Узоры, получающиеся друг из друга сдвигом, поворотом или отражением считаются различными.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2 2	14
2	3 3	322

5. Математическое моделирование

Задача Иосифа Флавия

(Время: 1 сек. Память: 16 Мб Сложность: 19%)

Существует легенда, что Иосиф Флавий - известный историк первого века - выжил и стал известным благодаря математической одаренности. В ходе иудейской войны он в составе отряда из 41 иудейского воина был загнан римлянами в пещеру. Предпочитая самоубийство плену, воины решили выстроиться в круг и последовательно убивать каждого третьего из живых до тех пор, пока не останется ни одного человека. Однако Иосиф наряду с одним из своих единомышленников счел подобный конец бессмысленным - он быстро вычислил спасительные места в порочном круге, на которые поставил себя и своего товарища. И лишь поэтому мы знаем его историю…

В нашем варианте мы начнем с того, что выстроим в круг N человек, пронумерованных числами от 1 до N, и будем исключать каждого k-ого до тех пор, пока не уцелеет только один человек.

Например, если N=10, K=3, то сначала умрет 3-й, потом 6-й, затем 9-й, затем 2-й, затем 7-й, потом 1-й, потом 8-й, за ним - 5-й, и потом 10-й. Таким образом, уцелеет 4-й.

Требуется написать программу, которая по заданным N и K будет определять номер уцелевшего человека.

Входные данные

Входной файл INPUT.TXT содержит два натуральных числа N и K. Ограничения: N<=500, K<=100.

Выходные данные

В выходной файл OUTPUT.TXT нужно вывести номер уцелевшего человека.

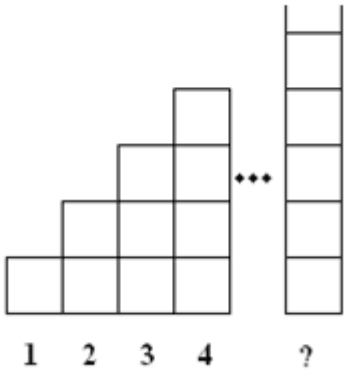
Пример

№	INPUT.TXT	OUTPUT.TXT
1	10 3	4

Лестница

(Время: 1 сек. Память: 16 Мб Сложность: 21%)

Мальчик Петя строит из кубиков лестницу. Лестница представляет собой несколько строящихся рядом башенок из кубиков, каждая из которых ровно на один кубик выше предыдущей. Требуется по имеющемуся у мальчика Пети числу кубиков определить, какой в кубиках будет высота последней ступеньки.



Входные данные

Входной файл INPUT.TXT содержит число K – количество кубиков у мальчика Пети ($1<=K<=10^6$).

Выходные данные

В выходной файл OUTPUT.TXT выведите количество кубиков в последней ступеньке у максимально высокой лестницы, которую можно построить из K кубиков.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	1	1
2	4	2
3	6	3

Мёд

(Время: 1 сек. Память: 16 Мб Сложность: 23%)

Все любят сладости и, в частности, мед. Винни-Пух тоже его любит. Каждый день он шел лакомиться медом, а по дороге домой заходил в гости к Кролику. Но приближалась зима, и Винни-Пух начал задумываться о запасах. Он решил в течении N дней не лакомиться медом, а собирать полный горшочек объемом V горстей и перекладывать его в бочку. В первый день

своего собирательства он так и сделал. Терпения хватило на один день. А на следующий день он не смог устоять и по дороге домой съел горсть меда из горшочка. В каждый следующий день из полного горшочка он съедал на одну горсть больше.

Необходимо определить объем меда, собранного Винни-Пухом на зиму.

Входные данные

Входной файл INPUT.TXT содержит три натуральных числа N (N<=300), V (V<=10^7) и K (K<=100). K – объем, на который Винни-Пух с каждым днем съедал больше меда.

Выходные данные

В выходной файл OUTPUT.TXT выведите два значения через пробел. Сначала идет строка «NO», если случилось, что Винни-Пух пришел к Кролику с пустым горшочком и «YES» в противном случае. Второе значение – объем меда, заготовленного Винни-Пухом на зиму.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	15 100 10	NO 550
2	10 10 1	YES 55

Часы с боем

(Время: 1 сек. Память: 16 Мб Сложность: 25%)

Старинные часы бьют каждые полчаса. Причем в начале каждого часа они бьют столько раз, сколько сейчас часов (по 1 разу – в час ночи и в час дня, по 2 раза – в два часа ночи в два часа дня и т.д., в полночь и в полдень они бьют, соответственно, по 12 раз). И еще 1 раз они бьют в середине каждого часа.

Дан промежуток времени (известно, что прошло строго меньше 24 часов). Напишите программу, определяющую, сколько ударов сделали часы за это время.

Входные данные

В первой строке входного файла INPUT.TXT записан начальный момент времени, во второй строке — конечный. Моменты времени задаются двумя целыми числами, разделяющимися пробелом. Первое число задает часы (от 0 до 23), второе — минуты (от 1 до 59, при этом оно не равно 30).

Выходные данные

В выходной файл OUTPUT.TXT выведите одно число — сколько ударов сделали часы за этот отрезок времени.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	5 20 10 25	45
2	10 25 5 20	135
3	5 2 5 21	0

Дни рождения

(Время: 1 сек. Память: 16 Мб Сложность: 27%)

Два одноклассника Петя и Вася родились не ранее 1993 и не позднее 1994 года, причем, Петя старше Васи.

Напишите программу, которая по заданным дням рождения определяет: на сколько дней Петя старше Васи.

Заметим, что 1993 и 1994 года не являются високосными, т.е. в феврале в них ровно 28 дней.

Входные данные

Входной файл INPUT.TXT содержит дату рождения Пети в первой строке и дату рождения Васи во второй. Даты заданы в формате «ДД.ММ.ГГ», например, строка 06.02.93 означает дату рождения 6 февраля 1993 года.

Выходные данные

В выходной файл OUTPUT.TXT выведите единственное число – искомое количество дней.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	01.01.93 02.01.93	1
2	05.02.94 05.03.94	28

Жук

(Время: 1 сек. Память: 16 Мб Сложность: 30%)

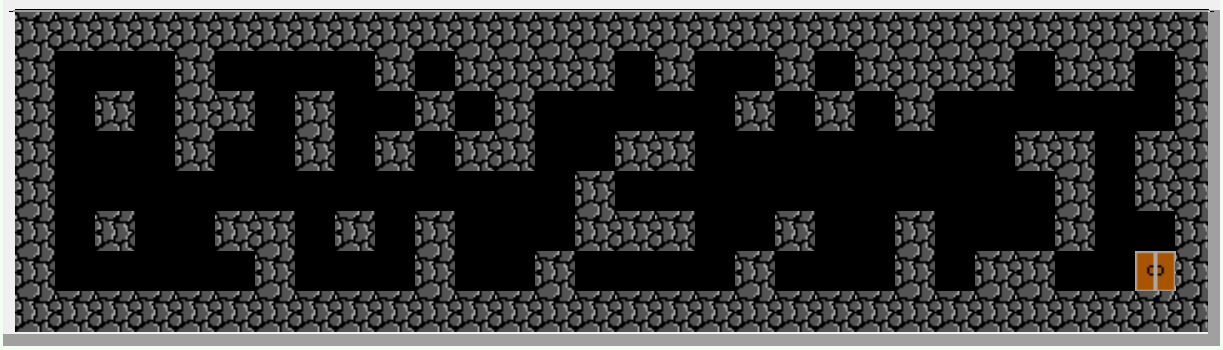
Петя нашел в Интернете по адресу **http://buglab.ru** игру-головоломку "Жук", в которой от участников требуется построить для жука лабиринт таким образом, чтобы жук как можно дольше искал выход.

Жук всегда начинает свое движение с левого верхнего угла, а выход всегда находится в правом нижнем. Жук движется не оптимально, а следующим образом: он идет туда, где еще не был, либо был там реже. Т.е.

проходя каждую клетку лабиринта, жук запоминает: сколько раз он был в этой клетке и при обдумывании направления своего движения в какой то конкретный момент он смотрит: сколько раз он был в клетке снизу, сколько справа, сколько слева и сколько сверху и движется туда, где он был меньше раз. Если таких направлений несколько и одно из них совпадает с текущим направлением движения, то он не меняет направления, иначе он движется согласно следующим приоритетам: вниз, направо, вверх, налево. Т.е. если минимальное число посещений сразу справа и слева (а двигался он при этом вверх или вниз), то жук идет направо, т.к. у "направо" приоритет выше. Следует заметить, что двигаясь по данному алгоритму жук всегда достигнет выхода в том случае, когда выход существует.

Изучив алгоритм движения жука Петя хочет написать программу, которая по заданному лабиринту определит количество перемещений жука прежде, чем он достигнет выхода. Помогите Пете с реализацией данной программы!

Конструктор лабиринта



Входные данные

Входной файл INPUT.TXT в первой строке содержит разделенные пробелом целые числа N и M - количество строк и столбцов в лабиринте (4 <= N, M <= 100). Далее следует N строк, содержащих данные лабиринта построчно. Каждая строка содержит M символов - клетки лабиринта текущей строки, где символ "@" обозначает присутствие стены, а символ пробела - пустое пространство. Гарантируется, что граница лабиринта окружена стеной. Предполагается, что жук начинает свое движение из координаты (2, 2) и заканчивает в координате (M-1, N-1), подразумевается, что в этих координатах нет стен.

Выходные данные

В выходной файл OUTPUT.TXT выведите количество движений жука, если спасительный маршрут для жука существует, и -1 в противном случае.

Примеры

1	INPUT.TXT	OUTPUT.TXT
1	6 6 @@@@@@ @ @ @ @ @ @ @@ @ @ @ @@@@@@	20
2	8 30 @@@@@@@@@@@@@@@@@@@@@@@@@@ @@@@@@ @ @ @ @@@@ @ @ @@@ @ @@ @ @ @ @@ @ @ @ @ @ @ @ @ @ @ @@ @@ @@ @@ @ @ @ @@ @ @ @@ @ @ @@@ @ @ @ @ @ @ @ @ @ @ @@ @ @@@@@@@@@@@@@@@@@@@@@@@@@@ @@@@@@	630
3	4 4 @@@@ @ @@ @@ @ @@@@	-1

Сообщество роботов

(Время: 1 сек. Память: 16 Мб Сложность: 30%)

Сообщество роботов живет по следующим законам:

- один раз в начале года они объединяются в группы по три или пять роботов;
- за год группа из трех роботов собирает 5 новых, а группа из 5 роботов – 9 новых;
- роботы объединяются так, чтобы собрать за год наибольшее количество новых роботов;
- каждый робот живет ровно три года после сборки.

В начале первого года было K роботов и все они были только что собраны.

Требуется написать программу, которая найдет количество роботов в начале N-го года.

Входные данные

Входной файл INPUT.TXT содержит записанные через пробел числа K (1 <= K <= 500) и N (1 <= N <= 100).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одно число - количество роботов в начале N-го года. Количество роботов меньше, чем 2^{31}.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3 2	8
2	8 2	22

Прыжки с шестом

(Время: 1 сек. Память: 16 Мб Сложность: 38%)

В соревнованиях по прыжкам с шестом было замечено одно интересное явление: на очередном этапе соревнований успешные и неуспешные попытки прыжков чередовались: успешный, неуспешный, успешный, неуспешный и т.д. (первый был успешным). Спортсменам разрешалась только одна попытка. Тот, кто преодолевал планку, переходил в следующий тур (этап), а тот, кто делал неудачную попытку – выбывал из соревнований. Таким образом, первым выбывал всегда спортсмен с номером 2, а последним – победитель с номером 1.

Требуется написать программу, которая по количеству участников и номеру спортсмена вычислит, каким по счету данный спортсмен выбыл из соревнований.

Входные данные

В единственной строке входного файла INPUT.TXT содержатся два натуральных числа: общее число спортсменов N и порядковый номер спортсмена в стартовом списке M. Числа разделены пробелом ($1 \le M,N \le 10^9$).

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести каким по счету спортсмен M выбыл из соревнований. Если это победитель состязания, то выводится число N.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	4 2	1
2	4 1	4
3	9 5	7

Танец

(Время: 1 сек. Память: 16 Мб Сложность: 45%)

На городском празднике танцуют девушки в красных и синих юбках. Они двигаются цепочкой и выполняют сложный рисунок танца. Из цепочки девушки выделяются по одной. Первая становится на левом краю сцены, вторая уходит в конец исходной цепочки, третья – на левый край сцены (справа от первой), четвертая – в конец исходной цепочки и т.д., пока все девушки не выстроятся на краю сцены.

Помогите постановщику танца определить, каким должно быть исходное расположение девушек, если на краю сцены, они выстроены так, что их юбки чередуются по цвету (слева направо): синяя, красная, синяя, красная и т.д.

Входные данные

Во входном файле INPUT.TXT записано натуральное число N – количество танцующих девушек (N <= 1000).

Выходные данные

В выходной файл OUTPUT.TXT выведите строку, содержащую цепочку из N символов, состоящую из заглавных букв B и R, соответствующих цветам юбок – синему и красному.

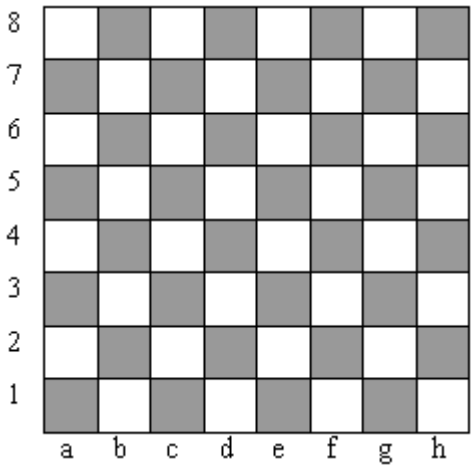
Примеры

№	INPUT.TXT	OUTPUT.TXT
1	2	BR
2	3	BBR
3	4	BBRR

Шашки

(Время: 1 сек. Память: 16 Мб Сложность: 45%)

Петя и Вася играли в шашки по описанным ниже правилам. В какой-то момент забежавший в комнату кот перевернул доску, на которой играли Петя и Вася. К счастью, у них осталась запись сделанных ходов, используя которую можно



восстановить расположение шашек к моменту, когда забежал кот.

Напишите программу, которая выведет положение шашек на доске после выполнения описанных ходов.

Игра происходит на стандартной доске (8x8), которая располагается так, что у игрока, играющего белыми, левая нижняя клетка является черной, и с нее начинается нумерация как строк, так и столбцов. Строки доски нумеруются числами от 1 до 8, столбцы — латинскими буквами от a до h.

В начале игры каждый из двух игроков имеет по 12 шашек своего цвета (белого или черного соответственно). Белые шашки располагаются на клетках a1, a3, b2, c1, c3, d2, e1, e3, f2, g1, g3, h2. Черные шашки располагаются на клетках a7, b6, b8, c7, d6, d8, e7, f6, f8, g7, h6, h8.

Игроки совершают ходы по очереди. Игрок, играющий белыми, ходит первым.

Шашки в процессе игры бывают двух видов: обычная шашка и дамка. В начале игры все шашки обычные. Белая шашка становится дамкой, если она оказывается в строке 8. Соответственно, черная шашка становится дамкой, если она оказывается в строке 1.

Шашка может совершать ходы двух типов:

1. *Простой ход* заключается в перемещении одной из шашек на одну клетку вперед по диагонали. Например, белая шашка с e3 может сходить на d4 или f4 (если соответствующая клетка свободна). А черная шашка с e3 может сходить на d2 или f2.

2. *Рубка* заключается в том, что шашка перепрыгивает через шашку (или дамку) противника, находящуюся в диагонально соседней с ней клетке при условии, что следующая клетка этой диагонали свободна. Шашка противника, которую срубили, убирается с доски. Если сразу после окончания рубки та же самая шашка может продолжить рубку, она ее продолжает этим же ходом. Рубка возможна в любом из 4-х диагональных направлений. Если в процессе рубки шашка оказывается в 1-й строке (для черных) или в 8-й (для белых), она становится дамкой.

Дамка может совершать следующие ходы:

3. *Простой ход* заключается в перемещении дамки по диагонали на любое число клеток (при этом все клетки, через которые происходит перемещение, должны быть свободны).

4. *Рубка* заключается в том, что шашка перепрыгивает через шашку (или дамку) противника, находящуюся на той же диагонали, что и рубящая дамка. Это можно делать при условии, что все клетки между рубящей дамкой и шашкой, которую рубят, а также клетка, следующая за шашкой, которую рубят, свободны. После рубки дамка может встать на любую клетку данной диагонали за клеткой, на которой стояла срубленная шашка (при условии, что

все клетки на ее пути свободны). Если из своего нового положения дамка может совершить рубку, она должна ее совершить этим же ходом.

Входные данные

В первой строке входного файла INPUT.TXT записано одно число N — количество ходов, которое было сделано с начала партии. Это количество не превышает 1000.

В каждой из следующих N строк записаны описания ходов (нечетные ходы совершались белыми, четные — черными). Описание каждого хода занимает ровно одну строку и записывается в следующем виде.

Простой ход записывается в виде <начальная клетка>–<конечная клетка>. Например, ход с c3 на d4 записывается как c3-d4.

Рубка записывается в виде <начальная клетка>:<клетка после рубки>. Если рубка продолжается, то ставится еще одно двоеточие, и записывается клетка, где окажется шашка после следующей рубки и т.д. Например, e3:c5:e7 (шашка, изначально расположенная на e3, рубит шашку на d4 и оказывается на c5, после чего рубит шашку на d6 и оказывается на e7).

Рубка a1:h8 может быть осуществлена только дамкой (например, дамка с a1 рубит шашку, стоящую в c3, и заканчивает ход в h8). Рубка дамкой может быть и такой a1:d4:f6:h4 (дамка рубит шашку, стоящую на b2, затем продолжает рубку и рубит шашку на e5, и, наконец, рубит шашку на g5). При этом после каждой рубки указывается клетка, на которой останавливается дамка перед следующей рубкой.

Строки с описанием ходов не содержат пробельных символов.

Выходные данные

В выходной файл OUTPUT.TXT выведите изображение доски с расположенными на ней шашками. В первой строке выходного файла должна быть выведена 8-я строка доски, во второй — 7-я и т.д. В каждой строке должно быть ровно 8 символов, описывающих клетки столбцов с a по h.

Белая клетка должна быть изображена символом “.” (точка), пустая черная клетка — символом “–“ (минус). Черная клетка, в которой стоит белая шашка — символом “w” (маленькая латинская буква w), а клетка с белой дамкой — символом “W” (заглавная латинская буква W). Аналогично клетка с черной шашкой должна быть изображена символом “b” (маленькая латинская буква b), а клетка с черной дамкой — символом “B” (большая латинская буква B).

Пример

№	INPUT.TXT	OUTPUT.TXT
1	3 c3-d4 f6-e5	.b.b.b.b b.b.b.b. .b.b.w.b

	d4:f6	-.-.-.-. .-.-.-.- w.-.w.w. .w.w.w.w w.w.w.w.

Двухтуровая олимпиада

(Время: 1 сек. Память: 16 Мб Сложность: 46%)

Как известно, личная олимпиада по информатике проходит в два тура. На каждом из туров участники получают какие-то баллы, при этом итоговый балл определяется как сумма полученных баллов. Известны баллы, которые каждый участник получил на каждом из туров. Жюри хочет фальсифицировать итоги олимпиады так, чтобы победил «нужный» участник.

При этом жюри может делать следующие «подтасовки» (можно делать несколько «подтасовок» применительно как к одному и тому же, так и к разным турам):

1. Прибавить к результатам всех участников по одному из туров одно и то же положительное число.
2. Умножить результаты участников по одному из туров на некоторый коэффициент, больший 1.

При этом должна сохраниться правдоподобность результатов, которая заключается в том, что никто из участников не должен получить больше 100 баллов за каждый из туров.

Определите список участников, которые в результате таких фальсификаций могут оказаться победителями олимпиады (то есть в сумме за два тура иметь не меньше баллов, чем каждый из остальных участников).

Входные данные

Во входном файле INPUT.TXT записано сначала число участников N (1<=N<=1000), затем N пар чисел — результаты каждого участника за 1-й и за 2-й туры (результат участника за тур — это вещественное число от 0 до 100) не более, чем с 3 знаками после десятичной точки.

Выходные данные

В выходной файл OUTPUT.TXT выведите сначала количество участников, которые смогут стать победителями олимпиады, а затем в возрастающем порядке их номера.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4	2

	45 90 70 80 0 0 75 75	2 4

Игра «Пуговицы»

(Время: 0,5 сек. Память: 16 Мб Сложность: 48%)

Правила игры очень просты. Перед двумя играющими находится кучка из K пуговиц. Играющие по очереди берут пуговицы из кучки, причем за один ход каждый из них может взять от 1 до L пуговиц. Выигрывает тот из спортсменов, которому удастся взять последнюю пуговицу.

Тот из игроков, которому по жребию выпадает делать первый ход, получает возможность собственноручно назначить число K. Тот из игроков, который будет ходить вторым, выбирает, в свою очередь, число L.

Вам необходимо определить наилучшую стратегию для участника, который ходит вторым.

Входные данные

Во входном файле INPUT.TXT записано одно натуральное число K ($1 \le K \le 10^8$) – общее количество пуговиц.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести целое число L ($2 \le L < K$) — максимальное количество пуговиц, которое можно взять за один ход, обеспечивающее победу второму игроку. Если таких чисел несколько, то следует вывести наименьшее из них. Если таких чисел нет, то следует вывести число 0.

Примеры

№	INPUT.TXT	OUTPUT.TXT
1	3	2
2	26	12
3	31	30

Игра "Баше"

(Время: 1 сек. Память: 16 Мб Сложность: 49%)

Студент и профессор играют в игру «Баше» по следующим правилам.

На столе лежат N экзаменационных билетов. Игроки делают ходы поочередно, и в свой ход каждый из игроков может взять от 1 до K билетов. Выигрывает тот игрок, который возьмет билет, оставшийся последним. Т.е. если его возьмет студент, то он получит «зачет», иначе он получит «незачет».

Будем называть сделанный ход ошибочным, если в этой ситуации можно было сходить иначе, гарантируя себе в дальнейшем выигрыш независимо от игры соперника. Будем называть ход правильным (или допустимым), если он не является ошибочным.

Ваша задача – проанализировать уже сыгранную партию и указать для каждого хода, был ли он правильным или ошибочным.

Входные данные

В первой строке входного файла INPUT.TXT записаны три целых числа: N, K, P ($2 \le N \le 10000$, $2 \le K \le 100$, $2 \le P$). Здесь P – количество ходов, которые сделали студент и профессор. В последующих P строках записаны числа (по одному числу на строке) в диапазоне от 1 до K.

Выходные данные

Выходной файл OUTPUT.TXT должен содержать P строк по одному символу на строке: «T» (правильный или допустимый ход – от слова True) или «F» (неверный ход – от слова False).

Пример

№	INPUT.TXT	OUTPUT.TXT
1	10 5 3 3 3 4	F F T

К коду Грея

(Время: 1 сек. Память: 16 Мб Сложность: 50%)

Рассмотрим циклическую последовательность попарно различных чисел $\{a_0, a_1, \dots, a_{2^n-1}\}$, $0 <= a_i <= 2^n-1$. Назовем эту последовательность кодом Грея, если любой a_i отличается от левого соседа a_{i-1} и правого соседа a_{i+1} только в одной цифре в двоичной записи этих чисел. Для a_0 левым соседом считается a_{2^n-1}, а для a_{2^n-1} правым соседом считается a_0.

Вася хочет запрограммировать игру-головоломку, которая будет позволять пользователю менять местами два любых числа a_i и a_j. Задача игрока – получить код Грея. Модуль, отвечающий за перестановку чисел, Вася берет на себя. А вот Ваша задача – написать программу, которая будет определять после каждой перестановки – является ли последовательность кодом Грея.

Входные данные

В первой строке входного файла INPUT.TXT содержится число n. В следующей строке перечислены попарно различные числа a_i. В третьей строке записано число m – количество перестановок, сделанных пользователем. В следующих m строках перечислены числа (i, j) – индексы

переставляемых элементов. Ограничения: $1 <= n <= 16$; $1 <= m <= 10^5$; $i \neq j$, $0 <= i, j < 2^n$.

Выходные данные

В выходной файл OUTPUT.TXT запишите m строчек – в i-той строке запишите «Yes», если после i-той перестановки последовательность стала кодом Грея и «No» в противном случае.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	2 0 1 3 2 2 1 2 2 1	No Yes

Коррозия металла

(Время: 1 сек. Память: 16 Мб Сложность: 55%)

Для хранения двух агрессивных жидкостей A и B используется емкость с многослойной перегородкой, которая изготавливается из имеющихся N листов. Для каждого листа i (i = 1, …, N) известно время его растворения жидкостью A — a_i и жидкостью B — b_i. Растворение перегородки каждой из жидкостей происходит последовательно лист за листом, с постоянной скоростью по толщине листа.

Требуется написать программу проектирования такой перегородки, время растворения которой было бы максимальным.

Входные данные

В первой строке входного файла INPUT.TXT записано число N ($1 <= N <= 256$). В каждой из последующих N строк содержатся два положительных вещественных числа a_i и b_i, разделенные пробелом.

Выходные данные

В первую строку выходного файла OUTPUT.TXT записать время растворения перегородки с точностью, не меньшей 10^{-3}. В следующую строку файла записать номера листов в порядке их расположения от жидкости A к жидкости B, разделяя числа пробелами.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	4 1 2 1 2	6.000 4 2 1 3

	0.5 1.5 7 3.5	