ЗМІСТ

[Теоретичні відомості 4](#_Toc115643733)

[Комбінаторика 4](#_Toc115643734)

[Рекурсія і перестановки 4](#_Toc115643735)

[Жадібний алгоритм 5](#_Toc115643736)

[**Задача о расписании** 5](#_Toc115643737)

[**Когда можно быть жадным?** 5](#_Toc115643738)

[**Другие Жадные алгоритмы** 7](#_Toc115643739)

[**Задачи, в которых жадные алгоритмы не дают оптимального решения** 7](#_Toc115643740)

[**Вариант 1** 7](#_Toc115643741)

[**Вариант 2** 8](#_Toc115643742)

[**Вариант 3** 8](#_Toc115643743)

[**Вариант 4** 9](#_Toc115643744)

[**Вариант 5** 10](#_Toc115643745)

[**Вариант 6** 10](#_Toc115643746)

[**Вариант 7** 11](#_Toc115643747)

[**Вариант 8** 11](#_Toc115643748)

[**Вариант 9** 12](#_Toc115643749)

[**Вариант 10** 12](#_Toc115643750)

[**Вариант 11** 13](#_Toc115643751)

[**Вариант 12** 14](#_Toc115643752)

[**Вариант 13** 14](#_Toc115643753)

[**Вариант 14** 15](#_Toc115643754)

[**Вариант 15** 15](#_Toc115643755)

[**Вариант 16** 16](#_Toc115643756)

[**Вариант 17** 16](#_Toc115643757)

[**Вариант 18** 17](#_Toc115643758)

[**Вариант 19** 18](#_Toc115643759)

[**Вариант 20** 18](#_Toc115643760)

[**Вариант 21** 19](#_Toc115643761)

[**Вариант 22** 20](#_Toc115643762)

[**Вариант 23** 22](#_Toc115643763)

[**Вариант 24** 23](#_Toc115643764)

[**Вариант 25** 23](#_Toc115643765)

[**Вариант 26** 24](#_Toc115643766)

[**Вариант 27** 25](#_Toc115643767)

[**Вариант 28** 25](#_Toc115643768)

[**Вариант 29** 26](#_Toc115643769)

[**Вариант 30** 27](#_Toc115643770)

[**Вариант 31** 27](#_Toc115643771)

[**Вариант 32** 28](#_Toc115643772)

[**Вариант 33** 29](#_Toc115643773)

[**Вариант 34** 30](#_Toc115643774)

[**Вариант 35** 31](#_Toc115643775)

[**Вариант 36** 32](#_Toc115643776)

[**Вариант 37** 32](#_Toc115643777)

[**Вариант 38** 33](#_Toc115643778)

[**Вариант 39** 34](#_Toc115643779)

[**Вариант 40** 35](#_Toc115643780)

[**Вариант 41** 36](#_Toc115643781)

[**Вариант 42** 37](#_Toc115643782)

[**Вариант 43** 37](#_Toc115643783)

[**Вариант 44** 39](#_Toc115643784)

[**Вариант 45** 39](#_Toc115643785)

[**Вариант 46** 40](#_Toc115643786)

[**Вариант 47** 41](#_Toc115643787)

[**Вариант 48** 42](#_Toc115643788)

[**Вариант 49** 42](#_Toc115643789)

[**Вариант 50** 43](#_Toc115643790)

[**Вариант 51** 43](#_Toc115643791)

[**Вариант 52** 44](#_Toc115643792)

[**Вариант 53** 45](#_Toc115643793)

[**Вариант 54** 46](#_Toc115643794)

[**Вариант 55** 47](#_Toc115643795)

[**Вариант 56** 47](#_Toc115643796)

[**Вариант 57** 48](#_Toc115643797)

[**Вариант 58** 49](#_Toc115643798)

[**Вариант 59** 50](#_Toc115643799)

[**Вариант 60** 50](#_Toc115643800)

[**Вариант 61** 51](#_Toc115643801)

[**Вариант 62** 51](#_Toc115643802)

[**Вариант 63** 52](#_Toc115643803)

[**Вариант 64** 53](#_Toc115643804)

[**Вариант 65** 53](#_Toc115643805)

[**Вариант 66** 54](#_Toc115643806)

[**Вариант 67** 55](#_Toc115643807)

[**Вариант 68** 55](#_Toc115643808)

[**Вариант 69** 56](#_Toc115643809)

[**Вариант 70** 58](#_Toc115643810)

[**Вариант 71** 58](#_Toc115643811)

[**Вариант 72** 59](#_Toc115643812)

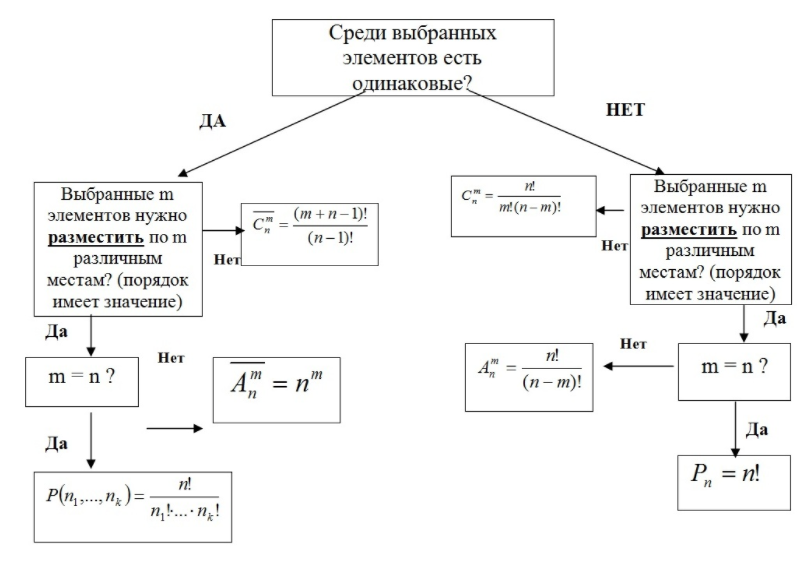
[**Вариант 73** 60](#_Toc115643813)

[**Вариант 74** 61](#_Toc115643814)

[**Вариант 75** 62](#_Toc115643815)

# Теоретичні відомості

## Комбінаторика

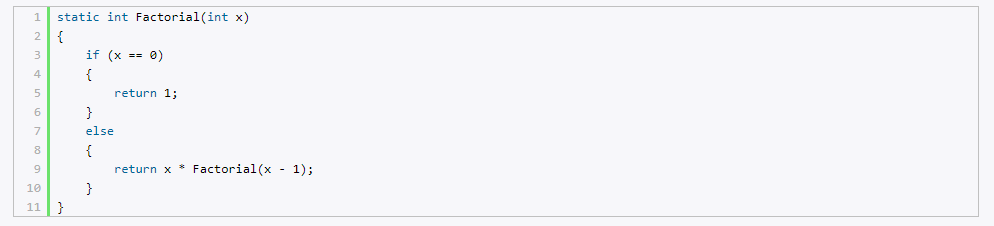


## Рекурсія і перестановки

Рекурсивная функция представляет такую конструкцию, при которой функция вызывает саму себя.

Возьмем, к примеру, вычисление факториала, которое использует формулу n! = 1 \* 2 \* … \* n. Например, факториал числа 5 равен 120 = 1 \* 2 \* 3 \* 4 \* 5.

Определим метод для нахождения факториала:



## Жадібний алгоритм

[Жадный алгоритм](http://ru.wikipedia.org/wiki/%D0%96%D0%B0%D0%B4%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) (greedy algorithm) — это алгоритм, который на каждом шагу делает локально наилучший выбор в надежде, что итоговое решение будет оптимальным.

К примеру, [алгоритм Дейкстры](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%94%D0%B5%D0%B9%D0%BA%D1%81%D1%82%D1%80%D1%8B) нахождения кратчайшего пути в графе вполне себе жадный, потому что мы на каждом шагу ищем вершину с наименьшим весом, в которой мы еще не бывали, после чего обновляем значения других вершин. При этом можно доказать, что кратчайшие пути, найденные в вершинах, являются оптимальными.

К слову, [алгоритм Флойда](http://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0_%E2%80%94_%D0%A3%D0%BE%D1%80%D1%88%D0%B5%D0%BB%D0%BB%D0%B0), который тоже ищет кратчайшие пути в графе (правда, между всеми вершинами), не является примером жадного алгоритма. Флойд демонстрирует другой метод — метод динамического программирования.

Использование жадного алгоритма довольно стандартное. Рассмотрим его на примере следующей задачи:

### **Задача о расписании**

Пусть программисту-фрилансеру Васе Пупкину дано n заданий. У каждого задания известен свой дедлайн, а также его стоимость(то есть если он не выполняет это задание, то он теряет столько-то денег). Вася настолько крут, что за один день может сделать одно задание. Выполнение задания можно начать с момента 0. Нужно максимизировать прибыль.

Классический пример применения жадины: Васе выгодно делать самые «дорогие задания», а наименее дорогие можно и не выполнять — тогда прибыль будет максимальна. Возникает вопрос: каким образом распределить задания? Будем перебирать задания в порядке убывания стоимости и заполнять расписание следующим образом: если для заказа есть еще хотя бы одно свободное место в расписании раньше его дедлайна, то поставим его на самое последнее из таких мест, в противном случае в срок мы его не можем выполнить, значит поставим в конец из свободных мест.

### **Когда можно быть жадным?**

~~Всегда~~ Иногда может возникнуть искушение использовать жадину везде, где только это возможно, но на некоторых задачах это неприемлимо. К примеру,[задача о рюкзаке](http://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BE_%D1%80%D0%B0%D0%BD%D1%86%D0%B5): вор пробрался на склад, в котором хранятся три вещи весом 10 кг, 20 кг и 30 кг и стоимостью 60, 100 и 120 деревянных вечнозеленых нанорублей соответственно. Вор максимум может унести 50 кг. Нужно максимизировать прибыль вора. Если поступать здесь жадно и выбирать самую ценную вещь(то есть, 6 нанорублей за кг первой штуки, 5 нанорублей за кг второй и 4 нанорубля за кг третьей), то вор должен взять первую вещь, потом останется место для второй вещи, однако оптимальное решение составляет вторая и третья вещь.

Вывод: есть область применимости жадных алгоритмов. Общих рецептов тут нет, но есть довольно мощный инструмент, с помощью которого в большинстве случаев можно определить, даст ли жадина оптимальное решение. Называется матроид.

***В рамках выполнения практической доказывать что жадный алгоритм являеться оптимальным не нужно, сведения ниже даються в ознакомительных целях***

Как подсказывает википедия, [матроид](http://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D1%80%D0%BE%D0%B8%D0%B4)— это пара (X, I), где X — конечное множество, называемое **носителем матроида**, а I — некоторое множество подмножеств X, называемое **семейством независимых множеств**. При этом должны выполняться следующие условия:

1.   
   Множество I непусто. Даже если исходное множество X было пусто — X = Ø, то I будет состоять из одного элемента — множества, содержащего пустое. I = {{Ø}}
2.   
   Любое подмножество любого элемента множества I также будет элементом этого множества.
3.   
   Если множества A и B принадлежат множеству I, а также известно, что размер А меньше B, то существует какой-нибудь элемент x из B, не принадлежащий А, такое что объединение x и A будет принадлежать множеству I. Это свойство является не совсем тривиальным, но чаще всего наиважшейшим из всех остальных.

Матроид называется взвешенным, если на множестве X существует аддитивная весовая функция w. Вес множества будет определяться как сумма весов его элементов.

Вернемся к программисту-фрилансеру с заданиями. Здесь можно разглядеть следующий матроид: пусть носителем будет множество заданий, а независимыми множествами — успешно выполненные задания. Весом каждой заявки пусть будет его стоимость. Проверим, является ли данная пара матроидом:

1. первое свойство, очевидно, выполняется. Пустое множество выполненных заданий входит в наше множество.То что Вася не хочет зарабатывать деньги не имеет значения.
2. второе множество тоже выполняется. Почему это так: давайте отсортируем успешно выполненные задания в порядке увеличения дедлайна. В таком порядке они все равно будут успешно выполненными. В таком порядке очевидно, что любое подмножество успешно выполненных заданий будет успешно выполнено.
3. третье свойство, хоть и не очевидно, но выполняется. Пусть у нас есть два множества успешно выполненных заданий A и B, при чем известно, что |A| < |B|. Стандартно отсортируем задания в порядке увеличения дедлайна в обоих множествах. Возьмем задание из B, которого нет в A, и попробуем добавить его к множеству A. Это у нас получится, ведь если бы в А не было пробела, то данное задание должно было присутствовать.

К чему это я? Вся прелесть матроидов заключается в теореме [Радо-Эдмондса](http://ru.wikipedia.org/wiki/%D0%96%D0%B0%D0%B4%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B4%D0%BE_%E2%80%94_%D0%AD%D0%B4%D0%BC%D0%BE%D0%BD%D0%B4%D1%81%D0%B0): если доказать, что объект является матроидом, то жадный алгоритм будет работать корректно и выдавать правильный результат.

### **Другие Жадные алгоритмы**

* [Алгоритм Хаффмана](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A5%D0%B0%D1%84%D1%84%D0%BC%D0%B0%D0%BD%D0%B0) (адаптивный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью).
* [Алгоритм Крускала](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%9A%D1%80%D0%B0%D1%81%D0%BA%D0%B0%D0%BB%D0%B0) (поиск остовного дерева минимального веса в графе).
* [Алгоритм Прима](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%9F%D1%80%D0%B8%D0%BC%D0%B0) (поиск остовного дерева минимального веса в связном графе).

Обобщением жадных алгоритмов является [алгоритм Радо— Эдмондса](https://ru.wikipedia.org/wiki/%D0%96%D0%B0%D0%B4%D0%BD%D1%8B%D0%B9_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A0%D0%B0%D0%B4%D0%BE_%E2%80%94_%D0%AD%D0%B4%D0%BC%D0%BE%D0%BD%D0%B4%D1%81%D0%B0).

### **Задачи, в которых жадные алгоритмы не дают оптимального решения**

Для ряда задач, относящихся к [классу NP](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81%D1%81_NP" \o "Класс NP), жадные алгоритмы не дают оптимального решения. К ним относятся:

* [задача коммивояжера](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BA%D0%BE%D0%BC%D0%BC%D0%B8%D0%B2%D0%BE%D1%8F%D0%B6%D0%B5%D1%80%D0%B0);
* [задача минимальной раскраски графа](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BA%D1%80%D0%B0%D1%81%D0%BA%D0%B0_%D0%B3%D1%80%D0%B0%D1%84%D0%B0);
* [задача разбиения графа на подграфы](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%B1%D0%B8%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B3%D1%80%D0%B0%D1%84%D0%B0);
* [задача выделения максимальной клики](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BE_%D0%BA%D0%BB%D0%B8%D0%BA%D0%B5);
* задачи, связанные с составлением [расписаний](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D0%B5" \o "Расписание).

### **Вариант 1**

Для того, чтобы заработать огромный капитал, новым русским необходимо иметь неординарное мышление. Конечно, при такой сложной работе, должны так же присутствовать какие-то особенные механизмы для отдыха и развлечений. В этих целях в казино был придуман специальный набор домино для новых русских. Обычные кости домино представляют собой набор из различных комбинаций сочетаний двух плиток, на каждой из которых отображается от 0 до 6 точек. А этот набор представляет собой подобные сочетания плиток, но количество точек на каждой может быть от нуля до заданного значения, которое зависит от интеллектуального уровня игроков. В таком наборе костей присутствуют всевозможные сочетания плиток, но при этом ни одна из костей не повторяется (даже такие комбинации как 2-5 и 5-2 считаются одинаковыми).

Для изготовления данного набора костей перед изготовителем встала проблема вычисления суммарного количества точек на всех костях домино. Это связано с тем, что домино для новых русских украшается бриллиантами, которые представляют собой точки на плитках и при изготовлении необходимо оценить стоимость.

Помогите написать программу, которая решит эту задачу.

Входные данные

Входной файл INPUT.TXT содержит одно натуральное число N – максимальное количество точек на одной плитке домино. (N ≤ 10000)

Выходные данные

В выходной файл OUTPUT.TXT выведите количество бриллиантовых камней, которые необходимо изготовить для заданного набора костей.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 | 12 |

### **Вариант 2**

Как-то раз, придя домой со школы, Света обнаружила записку от мамы, в которой она просила сделать салат. Света знала, что салат – это смесь двух или более ингредиентов, поэтому ей не составило труда выполнить мамину просьбу.

Но Света хочет стать математиком, поэтому, для тренировки, решила посчитать, сколько различных салатов она сможет сделать из имеющихся продуктов (майонез, огурцы, помидоры). После небольших расчетов она получила ответ: 4.

Зная, что вы любите интересные задачки, и хотите стать программистами, Света попросила вас написать программу, которая определяет количество различных салатов для произвольного числа ингредиентов.

Входные данные

Входной файл INPUT.TXT содержит натуральное число N – количество имеющихся ингредиентов (N < 32).

Выходные данные

В выходной файл OUTPUT.TXT выведите количество различных салатов.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 | 4 |
| 2 | 4 | 11 |

### **Вариант 3**

Найдите количество невырожденных прямоугольников со сторонами, параллельными осям координат, вершины которых лежат в точках с целыми координатами внутри или на границе прямоугольника, противоположные углы которого находятся в точках (0, 0) и (W, Н).

Входные данные

Входной файл INPUT.TXT содержит два натуральных числа W и Н, не превосходящих 1000.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 | 1 |
| 2 | 2 1 | 3 |
| 3 | 2 2 | 9 |

### **Вариант 4**

В известном городе Кызылорда, где находятся N центров, живет некий граф - Азамат. Он желает узнать количество различных построек дорог между ними, если известно, что два центра могут быть связаны в одном из двух направлений или не связаны вообще. Например, при N=2 все получается 3 варианта:

оба центра не связаны

дорога идет из первого во второй центр

дорога идет из второго в первый центр

Входные данные

Во входном файле INPUT.TXT записано единственное натуральное число - количество центров в городе, 2 ≤ N ≤ 100.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести число всевозможных построек дорог.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 | 3 |
| 2 | 4 | 729 |

### **Вариант 5**

Шаблоном размера n назовем строку длины n, каждый из символов которой входит в множество {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, ?}. Шаблоны преобразуются в строки из цифр по следующим правилам:

• символы от 0 до 9 могут быть преобразованы только сами в себя;  
• символ a может преобразован в любой из символов 0,1, 2, 3;  
• символ b может преобразован в любой из символов 1,2,3,4;  
• символ c может преобразован в любой из символов 2,3,4,5;  
• символ d может преобразован в любой из символов 3,4,5,6;  
• символ e может преобразован в любой из символов 4,5,6,7;  
• символ f может преобразован в любой из символов 5,6,7,8;  
• символ g может преобразован в любой из символов 6,7,8,9;  
• символ ? может преобразован в любой из символов от 0 до 9;

Даны два шаблона: p1 и p2. Рассмотрим множество S1 строк, которые могут быть получены из p1 по описанным правилам, и множество S2 строк, которые могут быть получены из p2. Необходимо найти количество строк, входящих в оба этих множества.

Входные данные

Первая строка входного файла INPUT.TXT содержит шаблон p1, вторая — шаблон p2. Шаблоны имеют одинаковый положительный размер, не больше 9.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | ??? abc | 64 |
| 2 | ??? 000 | 1 |
| 3 | abc 999 | 0 |

### **Вариант 6**

Задано натуральное число N. Требуется написать программу, которая находит количество натуральных чисел, не превышающих N и не делящихся ни на одно из чисел 2, 3, 5.

Входные данные

Входной файл INPUT.TXT содержит число N (1 ≤ N ≤ 109).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 10 | 2 |
| 2 | 20 | 6 |

### **Вариант 7**

На день рождения Пете подарили набор карточек с буквами. Теперь Петя с большим интересом составляет из них разные слова. И вот, однажды, составив очередное слово, Петя заинтересовался вопросом: "А сколько различных слов можно составить из тех же карточек, что и данное?".

Помогите ему ответить на этот вопрос.

Входные данные

Во входном файле INPUT.TXT задано слово, составленное Петей - строка из маленьких английских букв не длиннее 15 символов.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число - ответ на поставленную задачу.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | solo | 12 |

### **Вариант 8**

Петя играет в интересную игру. Для этой игры необходима монетка. Петя подбрасывает ее n раз и считает, сколько раз выпадает «решка». Если решка выпадает хотя бы m раз, то Петя считает, что он выиграл игру.

Однажды Петя задумался, какова вероятность того, что он выиграет игру. Для этого он хочет найти количество последовательностей результатов подбрасывания монетки, содержащих ровно n подбрасываний, при которых «решка» выпала хотя бы m раз.

Помогите Пете — найдите это число, считая, что при каждом броске монетка может выпасть либо «орлом», либо «решкой».

Входные данные

Входной файл INPUT.TXT содержит два целых числа: n и m (1 ≤ n ≤ 20, 0 ≤ m ≤ n).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

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

### **Вариант 9**

Все знают, что счастливые числа – это те числа, которые содержат только счастливые цифры 7 и 4. Вам нужно найти количество счастливых чисел не больших N.

Входные данные

В единственной строке входного файла INPUT.TXT записано натуральное число N, не превышающее 1032.

Выходные данные

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

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 56 | 4 |

### **Вариант 10**

В городском зоопарке содержатся животные n разных видов. Для участия в международной выставке «Три твари» зоопарк должен представить трех животных различных видов.

Требуется написать программу, которая вычислит число способов выбрать трех животных для участия в выставке.

Например, если в зоопарке два медведя, тигр, лев и пингвин, то есть семь способов выбрать трех животных:

первый медведь, тигр и лев;

первый медведь, тигр и пингвин;

первый медведь, лев и пингвин;

второй медведь, тигр и лев;

второй медведь, тигр и пингвин;

второй медведь, лев и пингвин;

тигр, лев и пингвин.

Входные данные

Входной текстовый файл INPUT.TXT содержит в первой строке натуральное число n – количество видов животных в городском зоопарке (1 ≤ n ≤ 1000). Во второй строке через пробел записаны n натуральных чисел – количество животных соответствующего вида. Число животных каждого вида не превышает 1000.

Выходные данные

Выходной текстовый файл OUTPUT.TXT должен содержать одно число – количество способов выбрать трех животных для международной выставки.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 4 2 1 1 1 | 7 |
| 2 | 3 100 100 100 | 1000000 |

### **Вариант 11**

Требуется найти число способов расставить на шахматной доске N×N K ладей так, чтобы они не били друг друга. Все ладьи считаются одинаковыми.

Входные данные

Во входном файле INPUT.TXT записаны натуральные числа N и K (N, K ≤ 8).

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число – ответ на задачу.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 8 8 | 40320 |

### **Вариант 12**

Перестановкой из N элементов называется упорядоченный набор из N различных чисел от 1 до N. Количество всех перестановок порядка N равно PN = N!

Требуется найти перестановку по ее номеру в лексикографическом порядке (по алфавиту). Например, для N=3 лексикографический порядок перестановок выглядит следующим образом:

(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1).

Таким образом, перестановка (2,3,1) имеет номер 4 в этой последовательности.

Входные данные

В первой строке входного файла INPUT.TXT записано число N (1 ≤ N ≤ 12) - количество элементов в перестановке, во второй - число K (1 ≤ K ≤ N!) - номер перестановки.

Выходные данные

В выходной файл OUTPUT.TXT выведите через пробел N чисел - искомую перестановку.

Примеры

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

### **Вариант 13**

Определим расстояние между равными по длине строками SA и SB (обозначим d(SA, SB)) как сумму для всех 1 ≤ i ≤ |SA| кратчайших расстояний между буквами SA(i) и SB(i) в циклически замкнутом английском алфавите (т.е. после буквы «a» идет буква «b», ..., после буквы «z» идет «a»). Например d(aba, aca) = 1, а d(aba, zbz) = 2.

Напомним, что циклическим сдвигом строки S называется строка (обозначим, как S→k) Sk+1Sk+2Sk+3 ... S|S|S1S2 ... Sk для некоторого k, где |S| – длина строки S.

Степенью циклического расстояния между строками SA и SB (|SA| = |SB|) называется сумма:

Text

Description automatically generated

Требуется посчитать степень циклического расстояния заданных строк SA и SB.

Входные данные

Входной файл INPUT.TXT содержит две строки равной длины, не превышающей 105 символов. Строки состоят только из маленьких букв английского алфавита.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | a b | 1 |
| 2 | ab ac | 8 |

### **Вариант 14**

Пусть х — натуральное число. Назовем у его делителем, если 1 ≤ у ≤ х и остаток от деления х на у равен нулю.

Задано число х. Найдите количество его делителей.

Входные данные

Входной файл INPUT.TXT содержит заданное число x (1 ≤ x ≤ 1018). Все простые делители числа x не превосходят 1000.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 12 | 6 |
| 2 | 239 | 2 |

### **Вариант 15**

Пусть х — натуральное число. Назовем у его делителем, если 1 ≤ у ≤ х и остаток от деления х на у равен нулю.

Задано число х. Найдите количество его делителей, делящихся на каждое из простых чисел, на которое делится х.

Входные данные

Входной файл INPUT.TXT содержит целое число x (1 ≤ х ≤ 1018). Все простые делители числа х не превосходят тысячу.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 12 | 2 |
| 2 | 239 | 1 |

### **Вариант 16**

Рассмотрим фигуру, аналогичную показанной на рисунке (большой равносторонний треугольник, составленный из маленьких равносторонних треугольников). На рисунке приведена фигура, состоящая из 4-х уровней треугольников.

Требуется написать программу, которая будет определять, сколько всего в ней треугольников (необходимо учитывать не только "маленькие" треугольники, а вообще все треугольники — в частности, треугольник, выделенный жирным, а также вся фигура, являются интересующими нас треугольниками).

Входные данные

Входной файл INPUT.TXT содержит одно число N — количество уровней в фигуре (1 ≤ N ≤ 105).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одно число - количество треугольников в такой фигуре.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 | 1 |
| 2 | 2 | 5 |
| 3 | 4 | 27 |

### **Вариант 17**

У вас имеется N выстроенных в ряд коробок, A красных и B синих шаров. Все красные шары (аналогично и синие) идентичны. Вы можете класть шары в коробки. Разрешается размещать в коробках шары как одного, так и двух видов одновременно. Так же разрешается оставлять некоторые из коробок пустыми. Не обязательно класть все шары в коробки.

Требуется написать программу, которая определяет количество различных способов, которыми возможно заполнить коробки шарами.

Входные данные

Входной файл INPUT.TXT содержит целые числа N, A, B. (1 ≤ N ≤ 20, 0 ≤ A, B ≤ 20)

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 1 | 4 |
| 2 | 2 1 1 | 9 |

### **Вариант 18**

Петя и Вася увлеклись азартной игрой на деньги. Игра состоит из множества раундов, в каждом из которых бросается монетка и если выпадает «Решка», то побеждает Петя и получает очко, в противном случае очко присуждается Васе. Набранные очки в разных раундах суммируются и тот, кто первым наберет N очков, выигрывает.

Вдруг, посреди игры прозвенел звонок, и ребятам пришлось закончить игру. Помогите Пете и Васе справедливо поделить поставленные на кон монеты между собой в соответствии с математическим ожиданием выигрыша.

Входные данные

Входной файл INPUT.TXT содержит в одной строке 4 целых числа N, K1, K2 и S. Все числа разделены пробелами. N – количество очков, которые необходимо набрать для победы. K1 и K2 – текущие очки Пети (K1) и Васи (K2). S – число монет, которые на кону и которые необходимо поделить между игроками.

Ограничения: 1 ≤ N ≤ 50, 0 ≤ K1, K2 < N, 1 < S < 10100.

Выходные данные

В выходной файл OUTPUT.TXT через пробел выведите два целых числа: количество монет, которые должны получить Петя и Вася. Не забудьте, что сумма этих чисел в точности должна быть равна S. Гарантируется, что однозначный ответ в целых числах существует.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1 2 16 | 4 12 |
| 2 | 4 1 1 2 | 1 1 |
| 3 | 5 1 4 32 | 2 30 |

### **Вариант 19**

Петя выписал все сочетания из N первых английских букв по K букв. В каждом сочетании он выписывал буквы в лексикографическом порядке. Сочетания он выписывал в лексикографическом порядке по одному в строке. Теперь он хочет узнать: какое слово записано в M-ой строке.

Входные данные

Во входном файле INPUT.TXT записаны целые числа N, K, M (1 ≤ N ≤ 26, 1 ≤ K ≤ N). Гарантируется, что M не превосходит количества всех выписанных сочетаний.

Выходные данные

В выходной файл OUTPUT.TXT выведите M-ое выписанное сочетание.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 4 2 3 | ad |

Пояснение

Все сочетания в порядке их записи: ab, ac, ad, bc, bd, cd. Здесь 3м по счету сочетанием является ad.

### **Вариант 20**

Партия в волейболе выигрывается командой, которая первой набирает 25 очков с преимуществом минимум в два очка. В случае равного счета 24-24, игра продолжается до достижения преимущества в 2 очка (26-24; 27-25).

Две сыгранные партии, закончившиеся с одинаковым счетом, будем считать разными, если строки, в которых выписан порядок набора очков командами, не равны.

Комитет по проведению соревнований по волейболу заинтересовался, количеством различных партий, заканчивающихся счетом 25:23. Их оказалось 16123801841550.

Определить, сколько существует различных партий, заканчивающихся заданным счетом.

Входные данные

Во входном файле INPUT.TXT указан конечный счет в партии (то есть такой, при котором победа в партии отдаётся одной из команд). Также известно, что ни одна из команд не набрала более 40 очков.

Выходные данные

В выходной файл OUTPUT.TXT выведите количество всевозможных партий, которые заканчиваются данным счетом.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 25:12 | 1251677700 |
| 2 | 20:25 | 1761039350070 |
| 3 | 25:23 | 16123801841550 |

### **Вариант 21**

Перестановкой из N элементов называется упорядоченный набор из N различных чисел от 1 до N.

Размещением порядка K называют подмножество элементов некоторой перестановки порядка N. Например, (1, 3) – размещение порядка 2 для перестановки (1, 2, 3) порядка 3.

Требуется по заданному размещению определить его позицию в лексикографическом порядке всех возможных размещений, образованных из всевозможных перестановок порядка N.

Например, лексикографическая последовательность всевозможных размещений для K=2 и N=3 выглядит следующим образом:

(1,2), (1,3), (2,1), (2,3), (3,1), (3,2)

Таким образом, перемещение (2,3) имеет номер 4 в этой последовательности.

Входные данные

В первой строке входного файла INPUT.TXT находятся числа N и K (1 ≤ K ≤ N ≤ 12). Во второй строке записаны K чисел из диапазона от 1 до N - размещение.

Выходные данные

В выходной файл OUTPUT.TXT выведите единственное число - номер данного размещения.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 2 3 2 | 6 |
| 2 | 6 4 1 3 2 5 | 14 |

### **Вариант 22**

Ваня решил заняться музыкой. Но какой музыкальный инструмент выбрать для изучения, он пока еще не решил. Его друг Женя, будучи немного постарше и опытнее, уже хорошо освоил гитару. Он хочет продемонстрировать Ване возможности игры на гитаре.

На гитаре шесть струн, каждая из них может быть зажата пальцем на определенном ладу или оставлена открытой (не прижатой ни на каком ладу), от этого изменяется нота, которая звучит, когда струна колеблется.

Чтобы продемонстрировать возможности инструмента, Женя ведет с Ваней такую беседу. Ваня называет некоторый аккорд, а Женя говорит ему, сколькими способами можно этот аккорд взять на гитаре, то есть выбрать на каждой струне лад и зажать его (или оставить ее открытой), чтобы этот аккорд зазвучал. Женя любит громкий звук, поэтому звучать должны все струны.

Нот в музыке, конечно, семь, но помимо них существуют еще полутона. Поэтому для удобства мы будем считать, что нот всего 12. Их обозначения такие: A, Bb, B, C, C#, D, D#, E, F, F#, G, G#.

Кроме этого, C# может быть обозначен как Db, Bb как A#, D# как Eb, F# как Gb, а G# как Ab. Обозначения нот приведены в порядке их следования, то есть за A идет Bb, за Bb идет B, . . . , за G# идет снова A. Расстояние между любыми двумя соседними нотами равно одному полутону.

Струн у гитары шесть, они занумерованы с 1 по 6. Каждая струна издает определенную ноту, когда звучит и не прижата ни на каком ладу. Совокупность нот, соответствующих открытым струнам, называется строем гитары. Если открытую струну, издающую ноту номер i, зажать на j-том ладу (лады нумеруются с единицы), то этой струной будет издаваться нота i + j, то есть, от ноты i надо сместиться на j нот по циклу. Например, если открытая струна издает ноту D, то зажав ее на третьем ладу, мы получим ноту F, а на восьмом – ноту Bb.

На гитаре Вани N ладов, поэтому 1 ≤ j ≤ N. Аккордов существует большое множество, но строятся они по общей схеме. Вначале выбирается так называемая тоника аккорда – нота, от которой он будет строиться. Затем остальные ноты аккорда выписываются уже относительно нее.

Запись аккорда состоит из двух частей. Первая часть – это всегда обозначение тоники аккорда. Вторая часть описывает само звучание аккорда. Мы ограничимся несколькими частными случаями.

Мажорный аккорд

К тонике добавляются ноты, отстоящие на 4 и 7 полутонов. Вторая часть записи этого аккорда пуста.  
Пример: Eb, состоит из нот Eb, G, Bb.

Минорный аккорд

К тонике добавляются ноты, отстоящие на 3 и 7 полутонов.  
Вторая часть записи этого аккорда состоит из маленькой английской буквы m.  
Пример: Am, состоит из нот A, C, E.

Мажорный септаккорд

Образуется из мажорного аккорда путем добавления ноты, отстоящей от тоники на 10 полутонов.  
Вторая часть состоит из цифры 7.  
Пример: E7, состоит из нот E, B, G#, D.

Минорный септаккорд

Образуется из минорного аккорда путем добавления ноты, отстоящей от тоники на 10 полутонов.  
Вторая часть состоит из буквы m и цифры 7.  
Пример: Gm7, состоит из нот G, D, Bb, F.

Напишите программу, помогающую Жене отвечать на вопросы Вани. Учтите, что аккорд считается взятым на гитаре, если ни одна нота аккорда не пропущена и не взята ни одна нота, не принадлежащая аккорду. В отличие от реальной игры на гитаре, расстояние между ладами, на которых зажата хотя бы одна струна, может быть любым.

Входные данные

В первой строке входного файла INPUT.TXT содержится N – число ладов гитары (0 ≤ N ≤ 9). Во второй строке записан строй гитары – шесть нот, соответствующие шести струнам гитары. Ноты обозначены, как в тексте задачи, и отделены друг от друга пробелами. Регистр букв важен. В третьей строке записан аккорд, который надо взять на гитаре.

Выходные данные

В выходной файл OUTPUT.TXT выведите количество способов взять данный аккорд на данной гитаре.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 E B G D A E C | 4 |
| 2 | 0 A C E A C E Am | 1 |

Пояснения к примерам

Если обозначить вариант взятия аккорда шестью цифрами – номерами ладов, на которых зажаты струны с первой по шестую, считая открытую струну зажатой на нулевом ладу то в первом примере четыре способа таковы:

(0; 1; 0; 2; 3; 0)  
(3; 1; 0; 2; 3; 0)  
(0; 1; 0; 2; 3; 3)  
(3; 1; 0; 2; 3; 3)

Во втором примере можно играть только на открытых струнах, зато гитара настроена ровно на требуемый аккорд.

### **Вариант 23**

В результате очередной хитроумной комбинации у Остапа Бендера и его компаньонов - K детей лейтенанта Шмидта оказалось X рублей пятирублевыми банкнотами. И вот дело, как водится, дошло до дележа...

Шура Балаганов предложил "по справедливости", т.е. всем поровну. Паниковский порешил себе отдать половину, а остальным "по заслугам". Каждый из K детей лейтенанта предложил что-нибудь интересное. Однако, у Великого Комбинатора имелось свое мнение на этот счет...

Ваша же задача состоит в нахождении количества способов разделить имеющиеся деньги между всеми участниками этих славных событий: K детьми лейтенанта Шмидта и Остапом Бендером.

Входные данные

Во входном файле INPUT.TXT записаны целые числа X (0 ≤ X ≤ 500) и K (0 ≤ K ≤ 100). Естественно, что число X делится на 5. Да и при дележе рвать пятирублевые банкноты не разрешается.

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число - количество способов дележа.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 15 2 | 10 |

### **Вариант 24**

Назовем число гладким, если его цифры, начиная со старшего разряда, образуют неубывающую последовательность. Упорядочим все такие числа в возрастающем порядке и присвоим каждому номер.

Вам требуется по номеру N вывести N-ое гладкое число.

Входные данные

Во входном файле INPUT.TXT содержится номер N (1 ≤ N ≤ 2147483647).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать искомое N-е гладкое число.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 | 1 |
| 2 | 11 | 12 |
| 3 | 239 | 1135 |

### **Вариант 25**

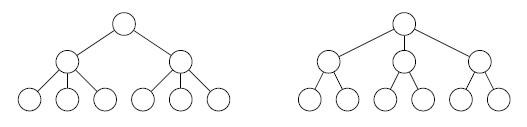
2-3 дерево — элегантная структура данных, изобретенная Джоном Хопкрофтом. Она предназначена для использования с той же целью, что и двоичное дерево поиска. 2-3 дерево представляет собой дерево с корнем, которое обладает следующими свойствами:

корень и каждая внутренняя вершина имеет либо 2 либо 3 ребенка;

глубина всех листьев одна и та же.

Единственное исключение — это когда дерево содержит ровно одну вершину. В этом случае корень дерева является и листом, и поэтому не имеет детей. Основная суть приведенных свойств в том, что дерево с L листьями имеет высоту O(log L).

Вообще говоря, может существовать несколько 2-3 деревьев с L листьями. Например, на следующем рисунке показаны два возможных дерева с 6 листьями.



По заданному числу L найдите количество различных 2-3 деревьев с L листьями. Так как ответ может быть довольно большим, выведите его по модулю R.

Входные данные

Входной файл INPUT.TXT содержит два целых числа: L и R (1 ≤ L ≤ 5 000, 1 ≤ R ≤ 109).

Выходные данные

В выходной файл OUTPUT.TXT выведите одно число – количество различных 2-3 деревьев, имеющих ровно L листьев, взятое по модулю R.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 6 1000000000 | 2 |
| 2 | 7 1000000000 | 3 |

### **Вариант 26**

Василий Афанасьев в качестве курсовой работы получил задание - построить компьютер, который бы работал не с числами, а со строками. Вася для начала фиксировал некоторый алфавит. Обозначим за K количество букв в этом алфавите. Далее, Вася фиксировал некоторый набор различных строк, длины не более N каждая, который он назвал базовым. Компьютер умеет работать только со строками, которые получаются конкатенацией (т.е. приписыванием) некоторых строк из этого набора друг к другу (одну и ту же строку при приписывании можно использовать несколько раз). Однако оказалось, что исходный базовый набор оказался чрезмерным! Это значит, что в нем была строка, при удалении которой из набора не изменяется множество строк, с которыми умеет работать компьютер.

Васю заинтересовал вопрос - как много может быть строк в нечрезмерном наборе, и сколько таких максимальных наборов существует.

Так как его компьютер еще не готов, то он попросил Вас посчитать это число.

Входные данные

Во входном файле INPUT.TXT содержится два целых числа 1 ≤ N ≤ 1 000 и 1 ≤ K ≤ 100 – соответственно максимальная длина строки и количество символов в алфавите.

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите количество строк в максимальном наборе. Во второй – количество таких наборов.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 | 1 1 |
| 2 | 3 2 | 12 1 |

### **Вариант 27**

Для заданных натуральных чисел N и K требуется вычислить количество чисел от 1 до N, имеющих в двоичной записи ровно K нулей.

Например, если N=8 и K=1, то мы можем записать все числа от 1 до 8 в двоичной системе счисления:

1, 10, 11, 100, 101, 110, 111 и 1000.

Откуда видно, что только числа 10, 101 и 110 имеют ровно один ноль в записи, т.е. правильный ответ – 3.

Входные данные

В единственной строке входного файла INPUT.TXT записано два натуральных числа через пробел N и K, не превышающих 109.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно целое число — количество чисел от 1 до N с K нулями в двоичном представлении.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 8 1 | 3 |
| 2 | 13 2 | 4 |
| 3 | 1000 5 | 210 |

### **Вариант 28**

Иван Иванович пригласил на свой день рождения много гостей. Он написал на карточках фамилии всех гостей и разложил эти карточки на столе, полагая, что каждый гость сядет там, где обнаружит карточку со своей фамилией (фамилии у всех гостей различны). Однако гости не обратили внимания на карточки и сели за стол в произвольном порядке. При этом Иван Иванович с удивлением обнаружил, что ни один гость не сел на предназначенное ему место.

Требуется написать программу, которая найдет сколькими способами можно рассадить гостей так, чтобы ни один из них не сидел там, где лежала карточка с его фамилией.

Входные данные

Во входном файле INPUT.TXT задано целое число N – количество гостей (1 ≤ N ≤ 100).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одно целое число – количество способов рассадить гостей.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 | 0 |
| 2 | 2 | 1 |
| 3 | 5 | 44 |
| 4 | 20 | 895014631192902121 |

### **Вариант 29**

Дана последовательность натуральных чисел 7, 11, 13, 14, 19, 21, 22, 25, ….

Требуется написать программу, которая по заданному N находит N-ый член этой последовательности.

Входные данные

Входной файл INPUT.TXT содержит число N (1 ≤ N ≤ 2147483647).

Выходные данные

В выходной файл OUTPUT.TXT выведите N-ый член последовательности.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 | 7 |
| 2 | 6 | 21 |

### **Вариант 30**

В древнем Китае было придумано множество интересных игр. Все знают очень известную игру «Majong». В данной задаче мы рассмотрим другую интересную китайскую игру «Цзяньшидзы». Кратко сформулируем правила игры:

Из двух куч каменей, двое играющих поочередно могут брать:

Произвольное ненулевое число камней из одной кучи.

Одновременно по одинаковому произвольному ненулевому числу камней из обеих куч.

Выигрывает тот, кто возьмет своим ходом последний камень.

Не сложно определить в каких случаях выигрывает первый, а в каких случаях выигрывает второй игрок. Задача сводится к нахождению так называемых проигрышных пар (ai, bi), означающих, что при кучах камней содержащих ai и bi камней проигрывает тот, кто в данный момент совершает свой ход. Вам предстоит решить эту задачу, но немного в другом виде.

Обозначим проигрышную, для ходящего позицию, когда в кучках a и b камней (a, b). Так как порядок куч не играет роли, то всегда будем считать, что a ≤ b. Упорядочим все проигрышные позиции в лексикографическом порядке, иначе говоря:

(a1, b1) < (a2, b2), если (a1< a2) или ((a1= a2) и (b1< b2)).

Занумеруем проигрышные пары, начиная с 0. Ваша задача: найти k-ю пару проигрышных куч.

Входные данные

В первой строке входного файла INTPUT.TXT находится число N, (1 ≤ N ≤ 1000) - количество тестов в файле. В следующих N строках содержатся числа ki, (0 ≤ ki ≤ 109) порядковый номер пары проигрышных куч, которую требуется найти.

Выходные данные

В выходной файл OUTPUT.TXT выведите N пар чисел (aki, bki) по одной в каждой строке.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 0 1 2 | 0 0 1 2 3 5 |

### **Вариант 31**

В современной биологии ученым часто приходится иметь дело с последовательностями ДНК. Эти последовательности зачастую являются очень длинными, и их ручная обработка требует большого количества времени и сил. Поэтому возникает идея автоматизировать этот процесс.

Для этого можно применять компьютерные методы обработки данных, например, весьма полезными оказываются алгоритмы на строках. В этой задаче последовательность ДНК будет представляться в виде непустой строки, все символы которой входят в множество {A, G, С, T}.

Пусть даны две последовательности ДНК: s = s1s2 … sn и t = t1t2 … tm. Будем говорить, что t может получится в результате эволюции из s, если s является подпоследовательностью t, то есть существует такая последовательность индексов 1 ≤ i1 < i2 < … < in ≤ m, что s1=ti1, s2=ti2, … sn=tin. Необходимо выяснить, может ли последовательность t получится в результате эволюции из s.

Входные данные

Первая строка входного файла INPUT.TXT содержит последовательность s, вторая — последовательность t. Размер входного файла не превосходит 256 килобайт.

Выходные данные

В выходной файл OUTPUT.TXT выведите слово YES, если последовательность t могла получиться в результате эволюции из s, и слово NO — иначе.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | GTA AGCTA | YES |
| 2 | AAAG GAAAAAT | NO |

### **Вариант 32**

Одного неформала выгнали с работы, и теперь ему надо как-то зарабатывать себе на жизнь. Поразмыслив, он решил, что сможет иметь очень неплохие деньги на продаже собственных волос. Известно, что пункты приема покупают волосы произвольной длины стоимостью С 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 ≤ 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 |

### **Вариант 33**

Петя разгадывает головоломку, которая устроена следующим образом. Дана квадратная таблица размера N×N, в каждой клетке которой записана какая-нибудь английская буква. Кроме того, дан список ключевых слов. Пете нужно, взяв очередное ключевое слово, найти его в таблице. То есть найти в таблице все буквы этого слова, причем они должны быть расположены так, чтобы клетка, в которой расположена каждая последующая буква слова, была соседней с клеткой, в которой записана предыдущая буква (клетки называются соседними, если они имеют общую сторону — то есть соседствуют по вертикали или по горизонтали). Например, на рисунке показано, как может быть расположено в таблице слово olympiad.

Когда Петя находит слово, он вычеркивает его из таблицы. Использовать уже вычеркнутые буквы в других ключевых словах нельзя. После того, как найдены и вычеркнуты все ключевые слова, в таблице остаются еще несколько букв, из которых Петя должен составить слово, зашифрованное в головоломке.

Помогите Пете в решении этой головоломки, написав программу, которая по данной таблице и списку ключевых слов выпишет, из каких букв Петя должен сложить слово, то есть какие буквы останутся в таблице после вычеркивания ключевых слов.

Входные данные

Во входном файле INPUT.TXT записаны два числа N (1 ≤ N ≤ 10) и M (0 ≤ M ≤ 100). Следующие N строк по N заглавных английских букв описывают ребус. Следующие M строк содержат слова. Слова состоят только из заглавных английских букв, каждое слово имеет длину от 1 до 100 символов. Гарантируется, что в таблице можно найти и вычеркнуть по описанным выше правилам все ключевые слова.

Выходные данные

В выходной файл OUTPUT.TXT выведите в алфавитном порядке оставшиеся в таблице буквы.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 3 POLTE RWYMS OAIPT BDANR LEMES OLYMPIAD PROBLEM TEST | AENRSW |
| 2 | 3 2 ISQ ABC IQW I IS | ABCQQW |

### **Вариант 34**

Как Вам уже стало известно, Петя очень любит программировать. Недавно он решил реализовать популярную карточную игру «Дурак». Но у Пети пока маловато опыта, ему срочно нужна Ваша помощь.

Как известно, в «Дурака» играют колодой из 36 карт. В Петиной программе каждая карта представляется в виде строки из двух символов, где первый символ означает ранг (‘6’, ‘7’, ‘8’, ‘9’, ‘T’, ‘J’, ‘Q’, ‘K’, ‘A’) карты, а второй символ означает масть (‘S’, ‘C’, ‘D’, ‘H’). Ранги перечислены в порядке возрастания старшинства.

Пете необходимо решить следующую задачу: сможет ли игрок, обладая набором из N карт, отбить M карт, которыми под него сделан ход? Для того чтобы отбиться, игроку нужно покрыть каждую из карт, которыми под него сделан ход, картой из своей колоды. Карту можно покрыть либо старшей картой той же масти, либо картой козырной масти. Если кроющаяся карта сама является козырной, то её можно покрыть только старшим козырем. Одной картой можно покрыть только одну карту.

Входные данные

В первой строке входного файла INPUT.TXT находятся два натуральных числа N и M (N ≤ 35, M ≤ 4, M ≤ N), а также символ R, означающий козырную масть. Во второй строке перечислены N карт, находящихся на руках у игрока. В третьей строке перечислены M карт, которые необходимо отбить. Все карты отделены друг от друга одним пробелом.

Выходные данные

В выходной файл OUTPUT.TXT выведите «YES» в случае, если отбиться можно, либо «NO» , если нельзя.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 6 2 C KD KC AD 7C AH 9C 6D 6C | YES |
| 2 | 4 1 D 9S KC AH 7D 8D | NO |

### **Вариант 35**

В наши дни в космосе находятся сотни спутников, и все они обмениваются данными. При этом система распознавания сигналов работает по схеме «Свой-Чужой». Один из спутников отправляет запрос другому спутнику в формате двух целых чисел, а второй спутник отвечает первому так же двумя целыми числами. Первые два числа первого спутника представляют собой сумму цифр и количество цифр тех двух чисел, которыми должен ответить второй спутник. При этом в качестве ответа должны получиться числа, представляющие наибольшее и наименьшее возможные значения, которые могут быть сформированы по описанному выше методу.

Вам предстоит написать программу, формирующую ответ для второго спутника по известным числам, полученным от первого спутника.

Входные данные

Во входном файле INPUT.TXT записаны 2 натуральных числа S и K, представляющих сумму и количество цифр соответственно (K ≤ 100). При этом гарантируется, что возможно составить хотя бы одно K-значное число, сумма цифр которого равна S.

Выходные данные

В выходной файл OUTPUT.TXT выведите два числа – ответ второго спутника. При этом следует помнить, что все числа не имеют лидирующих нулей.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 3 | 100 100 |
| 2 | 2 3 | 200 101 |
| 3 | 3 4 | 3000 1002 |

### **Вариант 36**

Васе уже надоели задачи на строки! А Вам? А что делать? Что ж, приступим. Дана строка из маленьких букв английского алфавита. Разрешается любой ее символ сдвинуть не более, чем на K позиций в любую сторону так, чтобы в конечном счете они все встали на разные позиции (кроме случая, когда K=0). Например, если строка - aababac, а K = 2, то таким образом можно получить строки abaaabc или aaaabbc, но нельзя - aaacbab или aaaacbb.

Вася хочет сделать так, чтобы получившаяся с помощью такой операции строка была минимально возможной лексикографически (т.е. расположена раньше всех по правилам упорядочивания слов в словаре). Как же ему быть?

Входные данные

В первой строке входного файла INPUT.TXT задано число K (K ≥ 0). Во второй строке задана сама исходная непустая строка, длиной не более 100 000 маленьких английских букв. Гарантируется, что K не превосходит длины строки.

Выходные данные

В выходной файл OUTPUT.TXT выведите лексикографически минимальный из возможных результатов.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 aababac | aaaabbc |

### **Вариант 37**

Петя, изучая, как меняется курс рубля по отношению к доллару и евро, вывел закон, по которому происходят эти изменения (или думает, что вывел :) ). По этому закону Петя рассчитал, каков будет курс рубля по отношению к доллару и евро в ближайшие N дней.

У Пети есть 100 рублей. В каждый из дней он может обменивать валюты друг на друга по текущему курсу без ограничения количества (при этом курс доллара по отношению к евро соответствует величине, которую можно получить, обменяв доллар на рубли, а потом эти рубли — на евро). Поскольку Петя будет оперировать не с наличной валютой, а со счетом в банке, то он может совершать операции обмена с любым (в том числе и нецелым) количеством единиц любой валюты.

Напишите программу, которая вычисляет, какое наибольшее количество рублей сможет получить Петя к исходу N-го дня.

Законы изменения курсов устроены так, что в течение указанного периода рублевый эквивалент той суммы, которая может оказаться у Пети, не превысит 108 рублей.

Входные данные

Первая строка входного файла INPUT.TXT содержит одно число N (1 ≤ N ≤ 5000). В каждой из следующих N строк записано по 2 числа, вычисленных по Петиным законам для соответствующего дня — сколько рублей будет стоить 1 доллар, и сколько рублей будет стоить 1 евро. Все эти значения не меньше 0.01 и не больше 10000. Значения заданы точно и выражаются вещественными числами не более, чем с двумя знаками после десятичной точки.

Выходные данные

В выходной файл OUTPUT.TXT выведите искомую величину с двумя знаками после десятичной точки.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 4 1 10 10 5.53 5.53 1.25 6 5 | 4000.00 |

### **Вариант 38**

Дано N целых чисел. Каждое из них можно один раз изменить не более чем на целую величину L как в сторону увеличения, так и в сторону уменьшения или оставить без изменения. Если после такой операции некоторые из чисел оказываются равными, то они засчитываются за одно. С данными числами произвели указанную операцию таким образом, что осталось минимально возможное количество чисел. Требуется написать программу для определения этого количества.

Входные данные

Входной файл INPUT.TXT содержит в первой строке натуральные числа L и N (N ≤ 100, L ≤ 3200), во второй строке N чисел (в диапазоне от -32768 до 32767), записанных через пробел.

Выходные данные

В выходной файл OUTPUT.TXT выведите единственное число – ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 10 3 11 21 27 | 1 |
| 2 | 5 3 6 10 27 | 2 |

### **Вариант 39**

Ну не гномы, а наказание какое-то! Подумала Белоснежка, в очередной раз пытаясь уложить гномов спать. Одного уложишь, другой уже проснулся! И так всю ночь. У Белоснежки n гномов, и все они очень разные. Она знает, что для того, чтобы уложить спать i-го гнома нужно ai минут, и после этого он будет спать ровно bi минут. Помогите Белоснежке узнать, может ли она получить хотя бы минутку отдыха, когда все гномы будут спать, и если да, то в каком порядке для этого нужно укладывать гномов спать.

Например, пусть есть всего два гнома, a1=1, b1=10, a2=10, b2=20. Если Белоснежка сначала начнет укладывать первого гнома, то потом ей потребуется целых 10 минут, чтобы уложить второго, а за это время проснется первый. Если же она начнет со второго гнома, то затем она успеет уложить первого и получит целых 10 минут отдыха.

Входные данные

Первая строка входного файла INPUT.TXT содержит число n (1 ≤ n ≤ 105), вторая строка содержит числа a1, a2, . . . an, третья - числа b1, b2, . . . bn (1 ≤ ai, bi ≤ 109).

Выходные данные

В выходной файл OUTPUT.TXT выведите n чисел – порядок, в котором нужно укладывать гномов спать. Если Белоснежке отдохнуть не удастся, выведите число −1.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 1 10 10 20 | 2 1 |
| 2 | 2 10 10 10 10 | -1 |

### **Вариант 40**

Рассмотрим следующую интересную игру для двух игроков. Для этой игры необходима таблица из 2-х строк и N столбцов, в клетках которой записаны натуральные числа, следующего вида:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A1 | A2 | A3 | ... | AN |
| B1 | B2 | B3 | ... | BN |

Игроки делают ходы по очереди. Начинает игру 1-й игрок.

За один ход 1-й игрок выполняет следующие два действия:

выбирает произвольный столбец (к примеру, j-й), который еще ни разу не был выбран одним из игроков на предыдущих ходах;

прибавляет к своим очкам число Aj.

За один ход 2-й игрок выполняет следующие два действия:

выбирает произвольный столбец (к примеру, j-й), который еще ни разу не был выбран одним из игроков на предыдущих ходах;

прибавляет к своим очкам число Bj.

Игра заканчивается, когда какой-либо из игроков не сможет сделать ход (по той причине, что все столбцы уже были выбраны). Изначально, у каждого из игроков есть 0 очков.

После того, как игра закончилась, происходит взаиморасчет между игроками. К примеру, 1-й игрок набрал S1 очков, а 2-й игрок - S2 очков. В случае, когда S1 > S2, 2-й игрок отдает 1-му игроку S1-S2 УДЕ (условных денежных единиц). В противном случае, 1-й игрок отдает 2-му игроку S2-S1 УДЕ. С этих позиций, целью 1-го игрока является максимизация величины S1-S2, а целью 2-го игрока - максимизация S2-S1.

Назовем стоимостью игры величину S1-S2 при оптимальной игре обоих игроков. Напишите программу, которая определяет стоимость игры.

Входные данные

В первой строке входного файла INPUT.TXT записано натуральное число N - количество столбцов в таблице (1 ≤ N ≤ 300000). Следующие N строк описывают числа в столбцах таблицы. i-я из этих строк содержит два натуральных числа Ai и Bi, разделенные одним пробелом (1 ≤ Ai, Bi ≤ 3000).

Выходные данные

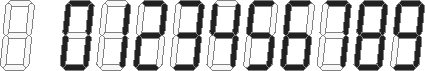
В выходной файл OUTPUT.TXT выведите одно целое число - стоимость игры.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 1 | 1 |
| 2 | 2 1 1 1 1 | 0 |
| 3 | 3 1 2 3 4 5 6 | 2 |

### **Вариант 41**

Недавно Вася приобрел настольный калькулятор с жидкокристаллическим индикатором. Этот индикатор отображает N цифр с помощью N одинаковых элементов.



Отметим, что каждый элемент содержит семь полосок, каждая из которых может быть либо белой, либо черной. В частности, при отображении цифры «1» черными являются две полоски.

Вася – очень любознательный мальчик, поэтому он хочет узнать, какое максимальное и минимальное N-значное число могут быть отображены на индикаторе его нового калькулятора так, чтобы черными были ровно K полосок.

Напишите программу, которая найдет ответ на Васин вопрос. Учитывайте при этом, что числа не могут содержать ведущие нули.

Входные данные

Входной файл INPUT.TXT содержит два натуральных числа N и K (1 ≤ N ≤ 100, 1 ≤ K ≤ 700).

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите минимальное число, во второй строке выходного файла выведите максимальное число. Если указанным образом не может быть представлено ни одно число, выходной файл должен содержать одну строку NO SOLUTION.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 15 | 10117 97111 |
| 2 | 10 1 | NO SOLUTION |

### **Вариант 42**

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 — ai и жидкостью B — bi. 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 ai и bi, разделенные пробелом (числа не превышают 106 и состоят не более чем из 11 значащих цифр).

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 0.5 1.5 7 3.5 | 6.000 4 2 1 3 |

### **Вариант 43**

Во многих фирмах, занимающихся торговлей, существует должность менеджера по закупкам. Как известно, они занимаются тем, что по торговому плану, представляющему собой список наименований товаров, для каждого из которых указано необходимое количество, закупает указанные в нем товары на оптовых базах. Торговый план при этом составляется руководством компании. Иногда у менеджеров по закупкам возникает желание принести выгоду не только своей фирме, но и себе.

Только что, как раз после подписания очередного торгового плана на заказ n наименований товаров, открылась новая оптовая база. Как это часто бывает сразу после открытия, ее цены на многие товары ниже заложенных в план. Наверное, этим можно воспользоваться.

На закупку товаров были выделены деньги из расчета того, что все товары будут закупаться на старой оптовой базе. Менеджер хочет, воспользовавшись возможностью покупать товары на новой базе, потратить как можно меньше денег на закупку требуемого количества товаров (не потраченные деньги он, конечно, сможет забрать себе).

Чтобы не вызывать сильных подозрений, производить на новой базе закупки, суммарная запланированная стоимость которых была больше, чем d денежных единиц, не следует. Осталось только рассчитать, какие товары и в каком количестве следует закупать на новой базе, чтобы осталось как можно больше не потраченных денег.

Входные данные

Первая строка содержит четыре числа: n (1 ≤ n ≤ 1000), d, а так же k1 и k2 (1 ≤ k1, k2 ≤ 1000) - количества наименований товаров, имеющихся на открытых ранее и новой базе соответственно. После этого идут n строк, каждая из которых содержит название товара в плане и его количество (положительное вещественное число). За ними следуют два блока из k1 и k2 строк соответственно, отделенные от предыдущего и разделенные между собой переводом строки - наименования товаров на базах и цены за единицу товара соответственно. Все цены являются положительными числами, даже на новой базе.

Названия товаров состоят из не более, чем 100 английских букв и символов подчеркивания, при этом регистр букв не учитывается. Вещественные числа заданы не более чем с двумя знаками после десятичной точки и по величине не превосходят 106. Гарантируется, что все товары из плана можно купить на старой базе. Считайте, что любой товар на любой базе можно покупать в любом дробном количестве, а также что при оплате можно использовать любое дробное количество денежных единиц.

Выходные данные

В выходной файл OUTPUT.TXT выведите n вещественных чисел, по одному на строке, задающих количество соответствующего товара, закупаемого на новой базе. На i-ой строке выведите количество товара, идущего i-ым в плане. Ошибки менее 0.01 будут игнорироваться.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 4 11.00 5 4 Salt 3 suGar 0.5 Powdered\_MilK 7 fLoUR 8  flOuR 1 CINNAmOn 8 saLt 0.5 SuGAr 20 poWdErED\_MILk 3  FLOur 2 SALT 0.4 liQUID\_HydRoGen 10000 sUgAR 10 | 2.0000 0.5000 0.0000 0.0000 |

### **Вариант 44**

Антон работает курьером. У него много заказов. На выполнение одного заказа у Антона уходит ровно один день. Для каждого заказа определена стоимость и срок его выполнения (количество дней, оставшихся до запланированного дня выполнения заказа). Однажды проснувшись, Антон изучил свой график и понял, что возможно он не сможет выполнить все заказы, и его могут уволить. Поэтому он решил выполнить лишь некоторые из них так, чтобы при этом получить максимальный доход.

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число N (1 ≤ N ≤ 1000) – количество заказов. Затем в N строках описаны данные каждого заказа Ti и Ci (натуральные числа, не превосходящие 105). Где Ti – последний день, в который еще можно выполнить заказ, Ci – вознаграждение за выполнение заказа.

Выходные данные

В выходной файл OUTPUT.TXT выведите максимальное вознаграждение, которое можно получить, выполняя заказы.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 1 10 2 12 | 22 |
| 2 | 3 1 10 1 20 3 24 | 44 |

### **Вариант 45**

Тетя Люба только что постирала все белье и теперь перед ней стоит непростая задача - как его высушить, чтобы ни одна вещь не успела испортиться. Сразу после стирки, i-я постиранная вещь имеет влажность wi. Если она сушится на веревке, то за минуту ее влажность уменьшается на 1, а если на батарее - то на r (если влажность была меньше r, то она становится равной 0). Причем веревок у тети Любы много (хватает для одновременной сушки всех вещей), а батарея только одна, причем такая маленькая, что на ней нельзя сушить две вещи одновременно. i-я вещь испортится, если не высохнет за время di. Помогите тете Любе составить план, когда какую вещь повесить на батарею.

Входные данные

Первая строка входного файла INPUT.TXT содержит целые числа n (1 ≤ n ≤ 105) - количество мокрых вещей, и r (1 ≤ r ≤ 109). Следующие n строк содержат описания постиранных вещей – пары чисел wi и di (1 ≤ wi, di ≤ 109).

Выходные данные

В выходной файл OUTPUT.TXT выведите план сушки в виде пар целых чисел ti и ki, где ti - время в минутах от начала сушки, а ki - номер вещи, которую нужно повесить на батарею в этот момент. Выводите пары в порядке увеличения ti. Пар не должно быть больше 105. Не выводите числа больше 109. Если высушить все вещи невозможно, выведите слово «Impossible».

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 3 2000 1000 2000 2000 2500 1500 | 0 3 500 1 1000 3 |
| 2 | 3 3 2000 1000 2000 1000 2000 1000 | Impossible |

### **Вариант 46**

В связи с проведением межпланетного шашечного турнира было принято решение о строительстве орбитальной гостиницы. Она должна была представлять собой большой куб из N×N×N блоков – маленьких кубиков 1×1×1, и каждый блок должен был быть окрашен снаружи со всех сторон в какой-то один цвет. При этом некоторые блоки могли быть покрашены в один и тот же цвет.

Через год были сделаны фотографии гостиницы с каждой из 6 сторон: спереди, слева, сзади, справа, сверху, снизу. За год эксплуатации могло случиться так, что из-за непрочного крепления некоторые блоки, из которых была построена гостиница, оторвались и улетели в открытый космос. Комиссия по восстановлению гостиницы хочет по сделанным снимкам установить максимальное возможное количество оставшихся блоков.

Итак, вам необходимо по видам гостиницы (куба N×N×N, из которого, возможно, выкинуты некоторые кубики 1×1×1) с 6 сторон определить, из какого максимального количества блоков 1×1×1 она может состоять. Может оказаться так, что гостиница состоит из двух или более не связанных между собой частей.

Входные данные

В первой строке входного файла INPUT.TXT находится число N — размер гостиницы (1≤N≤10). На следующих N строках записаны виды гостиницы с 6 сторон (в следующем порядке: спереди, слева, сзади, справа, сверху, снизу). Каждый такой вид представляет собой таблицу N×N, в которой различными заглавными английскими буквами обозначены различные цвета, а символом «.» (точка) — то, что в этом месте можно будет смотреть прямо сквозь гостиницу. Два последовательных вида отделяются друг от друга ровно одним пробелом в каждой из N строк.

Нижняя граница вида сверху соответствует верхней границе вида спереди, а верхняя граница вида снизу — нижней границе вида спереди. Для видов спереди, сзади и с боков верх и низ вида соответствуют верху и низу гостиницы.

Входные данные корректны, то есть во входном файле описано состояние, которое может получиться.

Выходные данные

Выведите в выходной файл OUTPUT.TXT одно число — искомое максимальное количество оставшихся блоков в гостинице.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 .R. YYR .Y. RYY .Y. .R. GRB YGR BYG RBY GYB GRB .R. YRR .Y. RRY .R. .Y. | 11 |
| 2 | 2 ZZ ZZ ZZ ZZ ZZ ZZ ZZ ZZ ZZ ZZ ZZ ZZ | 8 |

### **Вариант 47**

Толик придумал новую технологию программирования. Он хочет уговорить друзей использовать ее. Однако все не так просто. i-й друг согласится использовать технологию Толика, если его авторитет будет не меньше ai (авторитет выражается целым числом). Как только он начнет ее использовать, к авторитету Толика прибавится число bi (попадаются люди, у которых bi < 0). Помогите Толику наставить на путь истинный как можно больше своих друзей.

Входные данные

На первой строке входного файла INPUT.TXT содержатся два числа: n (1 ≤ n ≤ 1000) – количество друзей у Толика, и первоначальный авторитет Толика . Следующие n строк содержат пары чисел ai и bi. Все числа целые, по модулю не больше 106.

Выходные данные

В выходной файл OUTPUT.TXT выведите число m - максимальное число друзей, которых может увлечь Толик, и затем m чисел - номера друзей в том порядке, в котором их нужно агитировать.

Примеры

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

### **Вариант 48**

По кругу стоит N коробок. Каждая коробка имеет одного правого и одного левого соседа. В i-ой коробке находится Ai шаров. Известно, что общее количество шаров во всех коробках не превосходит N. За один ход разрешается переложить один шар из коробки в соседнюю. Какое наименьшее количество ходов придется совершить, чтобы в каждой коробке находилось не более одного шара?

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число N (1 ≤ N ≤ 1000). Во второй строке определена последовательность N целых чисел A1, A2, ... , AN (0 ≤ Ai ≤ N). Сумма всех значений Ai не превосходит N.

Выходные данные

В выходной файл OUTPUT.TXT выведите искомое минимальное количество ходов.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 7 1 0 0 0 2 3 1 | 7 |

### **Вариант 49**

Задано натуральное число N. Требуется написать программу, вычисляющую количество различных трехзначных чисел получающихся из N вычеркиванием цифр из его десятичной записи.

Входные данные

Входной текстовый файл INPUT.TXT содержит одно натуральное число N (1 ≤ N ≤ 10100).

Выходные данные

Выходной текстовый файл OUTPUT.TXT должен содержать одно целое число - найденное количество трехзначных чисел.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 12 | 0 |
| 2 | 111111111110011111111 | 4 |

### **Вариант 50**

Задано натуральное число x. Найдите число способов представить его в виде суммы четырех натуральных чисел: x = a + b + c + d, где a ≤ b ≤ c ≤ d.

Входные данные

Входной файл INPUT.TXT содержит целое число x (1 ≤ x ≤ 1500).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 | 0 |
| 2 | 5 | 1 |

### **Вариант 51**

Заданы две клетки шахматной доски. Требуется определить, возможно ли попасть из одной клетки в другую одним ходом шахматного коня, а если нет, то следует выяснить, возможно ли попасть с помощью двух ходов.

Входные данные

Входной файл INPUT.TXT содержит координаты двух клеток в общепринятом формате: каждая координата записывается как английская строчная буква и цифра, координаты отделены друг от друга запятой и пробелом.

Выходные данные

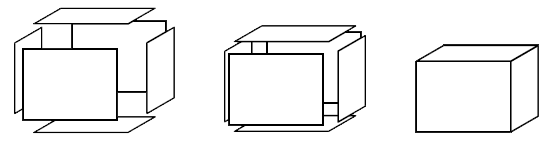
Выходной файл OUTPUT.TXT должен содержать цифру «1», если возможно из одной клетки в другую попасть за 1 ход, либо цифру «2», если попасть можно за 2 хода, либо «NO», если одна клетка недостижима из другой ни за 1 ни за 2 хода.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | a1, h8 | NO |
| 2 | a1, b3 | 1 |
| 3 | a1, d4 | 2 |

### **Вариант 52**

Иван работает на заводе, который производит тяжелую технику. Его работа очень проста – он собирает коробки и упаковывает в них технику для заказчиков. Каждая такая коробка представляет собой параллелепипед. Для сборки коробки Иван использует шесть прямоугольных деревянных плиток. Каждая плита представляет собой одну из сторон коробки.



Петр подбирает плитки для Ивана. Петр недостаточно умен и поэтому часто допускает ошибки – он приносит Ивану такие плитки, из которых невозможно собрать коробку. Но Иван не доверяет Петру. Поэтому он всегда тратит массу времени на то, чтобы объяснить Петру то, где он допустил ошибку.

К счастью, Петр обожает все, что связано с компьютерами и верит в то, что компьютеры никогда не ошибаются. Иван решил, что можно использовать это в их работе. Иван попросил Вас написать программу, которая по заданным размерам шести плиток скажет: возможно ли построить из них коробку.

Входные данные

Входной файл INPUT.TXT содержит шесть строк, каждая из которых содержит два натуральных числа w и h (1 ≤ w, h ≤ 10 000) – ширина и высота плиты в миллиметрах.

Выходные данные

В выходной файл OUTPUT.TXT выведите «POSSIBLE», если возможно собрать коробку из данных плит, и «IMPOSSIBLE» в противном случае.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1345 2584 2584 683 2584 1345 683 1345 683 1345 2584 683 | POSSIBLE |
| 2 | 1234 4567 1234 4567 4567 4321 4322 4567 4321 1234 4321 1234 | IMPOSSIBLE |

### **Вариант 53**

Chart, sunburst chart

Description automatically generated with medium confidence

Игра в дартс очень популярна в Великобритании и Голландии. В игре принимают участие несколько игроков. Они по очереди бросают в мишень по три дротика.

В начале игры каждый игрок имеет некоторое количество очков, обычно 501. За каждое попадание дротика в мишень сумма игрока уменьшается на некоторое число, в зависимости от того, в какую часть мишени он попал. Первый, кто достигает нуля очков, считается победителем.

Внешний вид мишени показан на рисунке справа. Она разделена на 20 секторов, расположенных вокруг небольшого центрального круга. Этот круг, в свою очередь, делится на внутреннюю и внешнюю часть (иногда внутренняя часть называется «яблочко»). Попадание во внешнюю часть центрального круга оценивается 25 очков, а в «яблочко» - вдвое больше, то есть в 50 очков. Стоимость сектора равняется числу, которое на нем написано. Кроме того на мишени выделены два кольца - внешнее и внутреннее. Попадание в них оценивается соответственно в два и в три раза больше, чем в оставшуюся часть соответствующего сектора.

Существуют дополнительные правила для последней серии бросков, в которой игрок должен достичь нуля очков. В этой серии игроку придется бросить в мишень от одного до трех дротиков. Игрок должен достичь в точности нуля очков, получение отрицательной суммы считается ошибкой. Последний дротик должен быть «двойным», то есть попасть во внешнее кольцо какого-либо сектора или в «яблочко» - (оно считается удвоением внешней часть центрального круга).

Например, один из правильных способов закончить игру, имея 50 очков - бросить дротики в «18» и «D16».

Способы «D20», «10», или «20», «T10» не подходят: последний бросок не является удвоенным. Еще один возможный способ победить в этом случае - просто попасть в «яблочко» («Bull»). По количеству оставшихся очков, найдите все способы правильно закончить игру.

Входные данные

Входной файл INPUT.TXT содержит число n - количество оставшихся очков (1 ≤ n ≤ 200).

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите k - количество способов правильно завершить партию. Каждая из следующих k строк должна содержать описание одного правильного способа. При этом число от 1 до 20 отвечает попаданию в соответствующий сектор. Буква «D» перед числом обозначает попадание во внешнее (удваивающее) кольцо, а «T» - во внутреннее (утраивающее). Внешняя часть центрального круга обозначается как «25», а «яблочко» (bull eye) - словом «Bull».

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 | 7 1 D1 D1 1 2 D1 1 D2 D1 1 D1 T1 D1 2 1 D1 3 D1 |

### **Вариант 54**

Описана рекурсивная функции с тремя параметрами F(a, b, c):

если a ≤ 0 или b ≤ 0 или c ≤ 0, то F(a, b, c) = 1

если a > 20 или b > 20 или c > 20, то F(a, b, c) = F(20, 20, 20)

если a < b и b < c, то F(a, b, c) = F(a, b, c-1) + F(a, b-1, c-1) - F(a, b-1, c)

иначе F(a, b, c) = F(a-1, b, c) + F(a-1, b-1, c) + F(a-1, b, c-1) - F(a-1, b-1, c-1)

Однако, если указанную функцию реализовать напрямую, то даже для небольших значений a, b и c (например, a = 15, b = 15, c = 15), программа будет работать несколько часов! Необходимо реализовать эффективный алгоритм вычисления функции F, который успеет найти любое ее значение менее чем за одну секунду!

Входные данные

Входной файл INPUT.TXT содержит три целых числа a, b, c - параметры функции F (-104 ≤ a,b,c ≤ 104).

Выходные данные

В выходной файл OUTPUT.TXT выведите значение функции F(a, b, c).

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 1 1 | 2 |
| 2 | 2 2 2 | 4 |
| 3 | 10 4 6 | 523 |
| 4 | 50 50 50 | 1048576 |

### **Вариант 55**

Вася учится в третьем классе и сейчас он проходит тему «Простые дроби с натуральными числителем и знаменателем». Оказывается, что дробь называется правильной, если ее числитель меньше знаменателя, и несократимой, если числитель и знаменатель являются взаимно простыми. Вася очень любит математику и поэтому дома он решает много задач. В данный момент Вася ищет наибольшую правильную несократимую дробь, у которой сумма числителя и знаменателя равна N.

Требуется написать программу, которая поможет Васе решить эту задачу.

Входные данные

Входной файл INPUT.TXT содержит одно целое число N (3 ≤ N ≤ 2∙109).

Выходные данные

Выходной файл OUTPUT.TXT должен содержать два числа – числитель и знаменатель найденной дроби, разделенные пробелом.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 | 1 2 |
| 2 | 10 | 3 7 |

### **Вариант 56**

Во многих книгах по занимательной математике приводится такая задача. Расставить по периметру треугольной комнаты 3 стула так, чтобы у каждой стены стояло по 2. Ее решение - поставить по стулу в каждый из углов комнаты.

Эта задача легко обобщается. Пусть комната представляет собой треугольник ABC. Даны: общее количество стульев n, количество стульев nAB, которое должно стоять у стены AB, количество стульев nBC, которое должно стоять у стены BC, количество стульев nAC, которое должно стоять у стены AC. Необходимо найти соответствующую расстановку стульев или установить, что ее не существует. При этом стулья можно ставить только в углы комнаты и вдоль стен, в центр комнаты стулья ставить нельзя. В любой из углов можно поставить произвольное количество стульев.

Входные данные

Входной файл INPUT.TXT содержит целые числа n, nAB, nBC, nAC (0 ≤ n, nAB, nBC, nAC ≤ 1000).

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите NO, если стулья указанным способом расставить невозможно. В противном случае выведите YES в первой строке выходного файла, а во второй выведите 6 целых неотрицательных чисел: kA, kAB, kB, kBC, kC, kAC - соответственно количество стульев, которые необходимо поставить в угол A, вдоль стены AB, в угол B, вдоль стены BC, в угол C и вдоль стены AC.

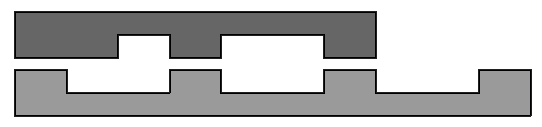
Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 2 2 2 | YES 1 0 1 0 1 0 |
| 2 | 3 3 2 2 | NO |

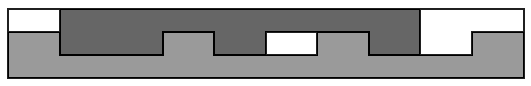
### **Вариант 57**

Исследовательская лаборатория одной известной автомобильной компании разработала специальный механизм, позволяющий повысить эффективность тормозов путем равномерной нагрузки деталей, используемых в тормозах.

Одним из основных компонентов механизма являются 2 прокладки, которые в процессе взаимодействия накладываются друг на друга. Каждая прокладка длины n разделена на n разделов, каждый из которых имеет высоту h или 2h. Таким образом, прокладки имеют зубчатую форму без закруглений.



В процессе взаимодействия прокладок важно, чтобы они накладывались друг на друга и при этом общая длина получившегося соединения была наименьшей.



По заданной конфигурации прокладок требуется определить наименьшую длину их возможного соединения, при котором общая высота конструкции не превышает значения 3h. При этом вращать прокладки и удалять зубцы запрещено.

Входные данные

Входной файл INPUT.TXT содержит 2 строки с описанием конфигурации 2х прокладок. Каждая конфигурация определяется последовательностью цифр 1 и 2, соответствующих высоте каждого зубца прокладки. Каждая из строк не пуста и имеет длину, не превышающую 100.

Выходные данные

В выходной файл OUTPUT.TXT требуется вывести наименьшую длину конструкции из заданных прокладок.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2112112112 2212112 | 10 |
| 2 | 12121212 21212121 | 8 |
| 3 | 2211221122 21212 | 15 |

### **Вариант 58**

У Вас есть N камней с массами W1, W2 , … WN. Требуется разложить камни на 2 кучки так, чтобы разница масс этих кучек была минимальной.

Входные данные

В первой строке входного файла INPUT.TXT записано число N – количество камней (1 ≤ N ≤ 18). Во второй строке через пробел перечислены массы камней W1, W2 , … WN (1 ≤ Wi ≤ 105).

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно неотрицательное целое число – минимально возможную разницу между массами двух кучек.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 5 8 13 27 14 | 3 |

### **Вариант 59**

Назовем перестановку из N чисел 1…N K-перестановкой, если любые два соседних в ней элемента отличаются не более, чем на K.

Требуется найти число K-перестановок из N чисел от 1 до N.

Входные данные

Входной файл INPUT.TXT содержит натуральные числа N и K (K ≤ N ≤ 9).

Выходные данные

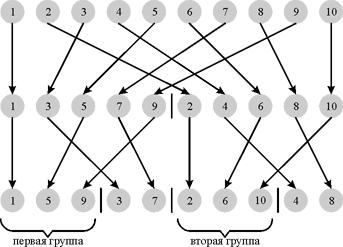
В выходной файл OUTPUT.TXT выведите целое число - ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1 | 2 |
| 2 | 4 2 | 12 |

### **Вариант 60**

Легион – основная организационная единица в армии Древнего Рима. В разное время легионы имели разную численность и различное построение. Самым простым построением была шеренга. Чтобы из N солдат легиона, выстроенных в шеренгу, отобрать троих в разведку, выполнялись следующие операции: если солдат в шеренге больше трех, то шеренга разбивалась на две, одна из которых состоит из солдат, стоящих на четных позициях, а вторая – из стоящих на нечетных позициях. Для всех полученных шеренг эта процедура повторялась до тех пор, пока в каждой из них не останется не более трех солдат. Если солдат осталось трое, то данную группу можно послать в разведку.



Требуется определить, сколько групп по три человека может быть сформировано из исходной шеренги.

Входные данные

Входной файл INPUT.TXT содержит единственное целое число N - количество солдат в шеренге (0 ≤ N ≤ 1018).

Выходные данные

В выходной файл OUTPUT.TXT выведите одно целое число – количество групп по три человека, сформированных из исходной шеренги.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 10 | 2 |
| 2 | 4 | 0 |

### **Вариант 61**

Дана строка, состоящая из N попарно различных символов. Требуется вывести все перестановки символов данной строки.

Входные данные

Входной файл INPUT.TXT содержит строку, состоящую из N символов (1 ≤ N ≤ 8), символы - буквы английского алфавита и цифры.

Выходные данные

В выходной файл OUTPUT.TXT выведите в каждой строке по одной перестановке. Перестановки можно выводить в любом порядке. Повторений и строк, не являющихся перестановками исходной, быть не должно.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | AB | AB BA |
| 2 | IOX | XOI OIX IXO XIO OXI IOX |

### **Вариант 62**

Марсиане Миша и Маша решили вместе подобрать подарок на день рождения Кати. Когда они наконец нашли то, что хотели, и упаковали предмет в красивую коробку, надо было решить, как подписать подарок. Друзья подумали, что лучшим решением будет составить общую подпись так, чтобы в ней как подстроки содержались их имена.

Учтите, что на Марсе принято подписываться полными именами, а они у марсиан могут быть достаточно длинными.

Входные данные

Входной файл INPUT.TXT содержит две строки, в которых записаны полные имена друзей. Имена, как ни странно, состоят из букв английского алфавита, из которых только первая – прописная. Длина имен от 1 до 1000 символов.

Выходные данные

В выходной файл OUTPUT.TXT выведите кратчайшую строку, в которой встречаются имена Миши и Маши одновременно. Буквы, с которых имена начинаются в этой строке нужно сделать большими. Если существует несколько решений, выведите то, которое меньше в алфавитном порядке (следует считать, что любая буква в верхнем регистре меньше, чем любая буква в нижнем регистре).

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | Misha Masha | MashaMisha |
| 2 | Julya Lyalya | JuLyalya |

### **Вариант 63**

Король Флатландии решил вырубить некоторые деревья, растущие перед его дворцом. Деревья перед дворцом короля посажены в ряд, всего там растет n деревьев, расстояния между соседними деревьями одинаковы.

После вырубки перед дворцом должно остаться m деревьев, и расстояния между соседними деревьями должны быть одинаковыми. Помогите королю выяснить, сколько существует способов вырубки деревьев.

Требуется написать программу, которая по заданным числам n и m определит, сколько существует способов вырубки некоторых из n деревьев так, чтобы после вырубки осталось m деревьев и соседние деревья находились на равном расстоянии друг от друга.

Входные данные

Входной файл INPUT.TXT содержит два целых числа n и m (0 ≤ m , n ≤ 1000).

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно целое число — искомое число способов.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 3 | 4 |

Пояснение к примеру

Если обозначить условно исходное расположение деревьев перед дворцом как «TTTTT», то возможные результаты после вырубки следующие:  
  
«TTT..», «.TTT.», «..TTT», «T.T.T».

### **Вариант 64**

Дана строка, состоящая из N символов. Требуется вывести все перестановки символов данной строки.

Входные данные

Входной файл INPUT.TXT содержит строку, состоящую из N символов (1 ≤ N ≤ 8), символы - буквы английского алфавита и цифры.

Выходные данные

В выходной файл OUTPUT.TXT выведите в каждой строке по одной перестановке. Перестановки можно выводить в любом порядке. Повторений и строк, не являющихся перестановками исходной, быть не должно.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | AB | AB BA |
| 2 | 122 | 122 212 221 |

### **Вариант 65**

Заданы три числа: a, b, c. Необходимо выяснить, можно ли так переставить цифры в числах a и b, чтобы в сумме получилось c.

Входные данные

Входной файл INPUT.TXT содержит три целых числа: a, b, c (0 ≤ a, b, c < 109). Числа разделены пробелом.

Выходные данные

В выходной файл OUTPUT.TXT следует вывести YES, если искомая перестановка цифр возможна, в противном случае необходимо вывести NO. При положительном ответе во второй строке следует вывести число x, получаемое перестановкой цифр числа a, и число y, получаемое перестановкой цифр числа b, сумма которых равна c. Числа x и y при выводе не должны содержать ведущих нулей. Числа в строке разделены пробелом. Если решений несколько, то следует вывести ту пару, в которой число x минимально.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 12 31 25 | YES 12 13 |
| 2 | 12 31 26 | NO |
| 3 | 101 2 13 | YES 11 2 |

### **Вариант 66**

Все мы в детстве играли в кубики. Были у нас и кубики с цифрами, и кубики с буквами. Были также и разноцветные кубики. Маленький Андрюша тоже любит играть в кубики. У него есть несколько наборов кубиков, причем все кубики из одного набора раскрашены одинаково, а кубики из разных наборов - по-разному.

На столе у Андрюши лежат два кубика. Помогите ему определить, принадлежат они одному набору или нет. Стол у Андрюши стеклянный, поэтому он видит цвета всех граней кубика. Кубики принадлежат одному набору, если один из них можно комбинацией поворотов вокруг осей, проходящих через середины противоположных граней, перевести в другой.

Входные данные

Первая строка входного файла INPUT.TXT содержит описание первого кубика в формате: цвет передней грани, цвет задней грани, цвет верхней грани, цвет нижней грани, цвет левой грани, цвет правой грани. Во второй строке находится описание второго кубика в таком же формате. Цвета граней кодируются числами. Все числа во входном файле - целые, положительные и не превосходят 100.

Выходные данные

В выходной файл OUTPUT.TXT выведите «YES», если первый и второй кубики принадлежат одному набору, и «NO» - в противном случае.

Примеры

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

### **Вариант 67**

Перестановкой P[1..n] размера n называется набор чисел от 1 до n, расположенных в определенном порядке. При этом в нем должно присутствовать ровно один раз каждое из этих чисел. Примером перестановок являются 1,3,4,5,2 (для n=5) и 3,2,1 (для n=3), а, например, 1,2,3,4,5,1 перестановкой не является, так как число 1 встречается два раза.

Число i называется неподвижной точкой для перестановки P, если P[i] = i. Например, в перестановке 1,3,4,2,5 ровно две неподвижных точки: 1 и 5, а перестановка 4,3,2,1 не имеет неподвижных точек.

Даны два числа: n и k. Найдите количество перестановок размера n с ровно k неподвижными точками.

Входные данные

Входной файл INPUT.TXT содержит два целых числа n (1 ≤ n ≤ 9) и k (0 ≤ k ≤ n).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 2 | 20 |
| 2 | 9 6 | 168 |
| 3 | 2 1 | 0 |
| 4 | 9 0 | 133496 |

### **Вариант 68**

Приближается Новый Год, и в магазинах начинают появляться различные елочные украшения. На прилавках можно увидеть различные шарики, шишечки, звездочки, но все-таки самым красивым украшением является гирлянда из разноцветных лампочек. Одна из фирм, занимающихся изготовлением елочных украшений, решила в этом году изготавливать гирлянды на заказ.

Гирлянды, изготавливаемые этой фирмы, состоят из лампочек различных цветов, соединенных проводами. Всего в гирлянде n лампочек, каждая из которых покрашена в один из k цветов, и m проводов (каждый провод соединяет ровно две лампочки). Далее мы будем считать, что лампочки пронумерованы натуральными числами от 1 до n.

К сожалению, не каждый дизайн гирлянды соответствует эстетическим взглядам заказчиков. Во-первых, лампочки, соединенные одним проводом должны быть разного цвета, во-вторых, сама конфигурация гирлянды (то есть то, какие лампочки и как соединены проводами) не может быть любой.

Один из отделов фирмы уже провел исследование и нашел наиболее «удачную» конфигурацию. Ваша же задача состоит в том, чтобы найти число способов раскрасить лампочки, чтобы получившаяся гирлянда удовлетворяла эстетическим взглядам заказчиков.

Входные данные

Первая строка входного файла INPUT.TXT содержит три целых числа: n, k, m (1 ≤ n, k ≤ 8, 0 ≤ m ≤ 10). Последующие m строк описывают провода. Описание каждого провода состоит из двух чисел u и v (1 ≤ u, v ≤ n, u ≠ v) – номеров лампочек, соединенных этим проводом.

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 2 1 1 2 | 2 |
| 2 | 4 4 0 | 256 |
| 3 | 4 4 6 1 2 1 3 1 4 2 3 2 4 3 4 | 24 |

### **Вариант 69**

Некоторые наборы из n слов длины n обладают интересным свойством - их можно расположить в клетках квадрата n×n так, что все слова набора можно прочитать как в вертикали, так и по горизонтали.

Примером такого набора слов является {"DATE", "FIND", "IDEA", "NEXT"}. Их можно расположить так:

A picture containing text, scoreboard, screen, electronics

Description automatically generated

Заметьте, что каждое слово можно прочитать как по горизонтали, так и по вертикали. Такие квадраты называются словарными квадратами, наибольший известный словарный квадрат в английском языке имеет размер 10×10.

Рассмотрим еще один пример словарного квадрата:

A screenshot of a computer

Description automatically generated with low confidence

Вам даны такие 2n слов, что из них можно построить два различных словарных квадрата размера n×n. Ваша задача состоит в том, чтобы разбить эти слова на две группы, по n слов в каждой, и построить из слов каждой группы словарный квадрат.

Гарантируется, что все данные вам слова являются английскими словами (некоторые из них могут быть достаточно редкими словами, именами, или специальными терминами).

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число n (2 ≤ n ≤ 10). Каждая из следующих 2n строк содержит слово, состоящее из заглавных букв английского алфавита. Каждое слово содержит ровно n букв.

Выходные данные

В выходной файл OUTPUT.TXT выведите два словарных квадрата, построенных из данных слов. Разделите квадраты пустой строкой.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 4 ARTS BEST CRAB DATE FIND IDEA NEXT RARE | CRAB RARE ARTS BEST  FIND IDEA NEXT DATE |

### **Вариант 70**

Студент Дима учится на втором курсе университета. На втором курсе в его университете читается курс математической логики. В этом курсе особое внимание акцентируется на автоматических доказателях – программах, позволяющих с их помощью доказывать различные сложные теоремы.

Курсовая работа по математической логике у Димы такая: необходимо доказать с помощью автоматического доказателя HOL, что шахматную доску размером 2N x 2N, из которой вырезана одна клетка, можно покрыть в один слой уголками из трех клеток.

Дима не верит в то, что это правда, и пытается составить контрпример. Ваша задача доказать Диме, что он неправ, и решить задачу для Диминых входных данных.

Входные данные

Входной файл INPUT.TXT содержит три натуральных числа N, X, Y (N ≤ 6; X,Y ≤ 2N). Этими числами задана доска 2N x 2N, из которой вырезана клетка с координатами (X, Y). X – координата по горизонтали, Y – по вертикали, (1, 1) – верхний левый угол доски.

Выходные данные

В выходной файл OUTPUT.TXT выведите 2N строк по 2N чисел – номера уголков, покрывающих соответствующие клетки. Каждый уголок характеризуется своим уникальным номером. Уголки пронумерованы начиная с единицы, без пропусков. Вырезанную клетку следует обозначить нулем.

Примеры

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

### **Вариант 71**

Несколько человек решили поехать отдохнуть на природе, подышать свежим воздухом и т.п. Как это часто бывает, некоторые из них дружат друг с другом, а некоторые - нет. Для того, чтобы не испортить никому настроение, они решили разделиться на несколько групп. При этом, в каждой группе должно быть не более 5 человек и они должны дружить друг с другом.

Найдите такое разбиение людей на группы, в котором размер наибольшей группы был бы максимальным (среди всех разбиений).

Входные данные

Первая строка входного файла INPUT.TXT содержит целое число n (1 ≤ n ≤ 15) - количество людей. Следующие n строк содержат по n чисел. Если i-ый и j-ый люди дружат, то j-ое число i + 1-ой строки равно 1, иначе - 0.

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите число групп. Во второй строке выходного файла выведите n чисел (i-ое число - номер группы, в которой находится i-ый человек). Так как в любом случае количество групп не превзойдет n, нумеруйте группы целыми числами от 1 до n. Если решений несколько, то выведите любое.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1 1 0 1 1 0 0 0 1 | 2 1 1 2 |
| 2 | 8 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 | 3 1 1 1 1 1 2 2 3 |

### **Вариант 72**

Одной из классических NP-полных задач является так называемая «Задача о рюкзаке». Формулируется она следующим образом. Дано n предметов, каждый из которых характеризуется весом wi и полезностью pi. Необходимо выбрать некоторый набор этих предметов так, чтобы суммарный вес этого набора не превышал W, а суммарная полезность была максимальна.

Ваша задача состоит в том, чтобы написать программу, решающую задачу о рюкзаке.

Входные данные

Первая строка входного файла INPUT.TXT содержит натуральные числа n (1 ≤ n ≤ 20) и W (1 ≤ W ≤ 109). Каждая из последующих n строк содержит описание одного предмета. Каждое описание состоит из двух чисел: wi – веса предмета и pi – его полезности (1 ≤ wi, pi ≤ 109).

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите количество выбранных предметов и их суммарную полезность. Во второй строке выведите через пробел их номера в возрастающем порядке (предметы нумеруются с единицы в порядке, в котором они перечислены во входном файле).

Если искомых наборов несколько, выберите тот, в котором наименьшее число предметов. Если же после этого ответ по-прежнему неоднозначен, выберите тот набор, в котором первый предмет имеет наименьший возможный номер, из всех таких выберите тот, в котором второй предмет имеет наименьший возможный номер, и т.д.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 2 10 10 100 9 80 | 1 100 1 |
| 2 | 5 100 80 1000 50 550 50 550 50 550 50 550 | 2 1100 2 3 |
| 3 | 6 100 80 1000 50 550 50 550 50 550 50 550 100 1100 | 1 1100 6 |

### **Вариант 73**

В волшебной стране используются монетки достоинством A1, A2,..., AM. волшебный человечек пришел в магазин и обнаружил, что у него есть ровно по две монетки каждого достоинства. Ему нужно заплатить сумму N. Напишите программу, определяющую, сможет ли он расплатиться без сдачи.

Входные данные

Во входном файле INPUT.TXT записано сначала число N (1 ≤ N ≤ 109), затем - число M (1 ≤ M ≤ 15) и далее M попарно различных чисел A1, A2,..., AM (1 ≤ Ai ≤ 109).

Выходные данные

В выходной файл OUTPUT.TXT выведите количество монет, которое придется отдать волшебному человечку, если он сможет заплатить указанную сумму без сдачи. Если решений несколько, выведите вариант, в котором волшебный человек отдаст наименьшее возможное количество монет. Если без сдачи не обойтись, то выведите одно число 0. Если же у волшебного человечка не хватит денег, чтобы заплатить указанную сумму, выведите одно число -1 (минус один).

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 2 1 2 | 3 |
| 2 | 7 2 1 2 | -1 |
| 3 | 5 2 3 4 | 0 |

### **Вариант 74**

Найти количество N-значных чисел, у которых сумма цифр равна их произведению. Вывести наименьшее среди таких чисел для заданного N.

Входные данные

В единственной строке входного файла INPUT.TXT записано одно натуральное число N, которое не превышает 20.

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести через пробел 2 числа: количество искомых чисел и наименьшее среди них.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 | 10 0 |
| 2 | 2 | 1 22 |

### **Вариант 75**

В каждом языке программирования, даже самом простом, есть оператор ветвления, позволяющий проверить истинность логического выражения и, в зависимости от его результата, выполнить то или иное действие. Условие оператора ветвления представляет собой логическое (булевское) выражение, результатом которого может быть либо истина (TRUE), либо ложь (FALSE). Переменные, которые могут участвовать в логическом выражении, называются булевскими (boolean). Булевские переменные могут объединяться в сложные условия при помощи логических операций (функций):

AND(x1,x2,.. xs). Операция «И», возвращает истинное значение, если все ее операнды истинны. (2 ≤ количество операндов ≤ s);

OR(x1,x2,.. xs). Операция «ИЛИ», возвращает истинное значение, если хотя бы один ее операнд истинен. (2 ≤ количество операндов ≤ s);

NOT(x1). Операция «НЕ», меняет значение операнда х1 на противоположное (операнд всегда один).

В логическом условии может использоваться несколько логических функций, вложенных друг в друга, то есть результат одной функции может использоваться другой в качестве операнда. Например, AND(A, B, OR(С,D)). Данное выражение будет истинно тогда, когда истинны А, B и (С или D).

Требуется написать программу, которая по имеющемуся логическому выражению и значению логических переменных определит результат выражения. Количество операндов у функций AND и OR всегда равно двум (s=2).

Входные данные

Первая строка входного файла INPUT.TXT содержит логическое выражение (длина не более 255 символов). Вторая строка содержит два числа, разделенных одним или несколькими пробелами: N – количество блоков (не более 10), K - количество переменных (не более 26). Далее следует N блоков, каждый имеет следующую структуру: состоит из K строк, каждая содержит выражение типа <переменная = значение>. Переменные задаются заглавными английскими буквами, значение – константами TRUE или FALSE (заглавные буквы).

Выходные данные

В выходной файл OUTPUT.TXT выведите N строк со значением результата логического выражения для переменных соответствующего блока.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | AND(A,NOT(B)) 1 2 A=FALSE B=TRUE | FALSE |
| 2 | OR(NOT(AND(A,B)),A) 3 2 A=FALSE B=TRUE A=TRUE B=TRUE A=FALSE B=FALSE | TRUE TRUE TRUE |

### **Варіант 76**

Радиолюбитель Петя решил собрать детекторный приемник. Для этого ему понадобился конденсатор емкостью C мкФ. В распоряжении Пети есть набор из N конденсаторов, емкости которых равны C1, C2, ... ,CN соответственно. Петя помнит, как вычисляется емкость параллельного соединений двух конденсаторов (Cnew = C1 + C2) и последовательного соединения двух конденсаторов (Cnew = (C1\*C2)/(C1+C2) ). Петя хочет спаять некоторую последовательно-параллельную схему из имеющегося набора конденсаторов, такую, что ее емкость ближе всего к искомой (то есть абсолютная величина разности значений минимальна). Для изготовления схемы Петя может использовать от 1 до N из имеющихся у него конденсаторов.

Напомним определение последовательно-параллельной схемы. Схема, составленная из одного конденсатора, - последовательно-параллельная схема. Любая схема, полученная последовательным соединением двух последовательно-параллельных схем, - последовательно-параллельная, а также любая схема, полученная параллельным соединением двух последовательно-параллельных схем, - последовательно-параллельная.

Входные данные

В первой строке каждого входного файла INPUT.TXT заданы числа N и C (N - целое: 1 ≤ N ≤ 6, С - вещественное с не более, чем 4 знаками после запятой: 0 < C < 1000).

Во второй строке содержится последовательность емкостей имеющихся в наличии конденсаторов C1, C2, ... ,CN . Все значения Ci - натуральные числа, не превышающие 1000.

Выходные данные

В выходной файл OUTPUT.TXT необходимо вывести YES, если Пете удастся собрать схему, емкость которой отличается не более чем на 0.01 от требуемого значения C. В противном случае следует вывести NO.

Пример

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1.66 1 2 1 | YES |

Пояснения к примеру

Последовательно соединим первый и второй конденсаторы, а затем полученную схему соединим параллельно с третьим. Полученная схема будет иметь емкость 1.(6)

### **Вариант 77**

Магараджа — это шахматная фигура, сочетающая возможности ферзя и коня. Таким образом, магараджа может ходить и бить на любое количество клеток по диагонали, горизонтали и вертикали (т.е. как ферзь), а также либо на две клетки по горизонтали и на одну по вертикали, либо на одну по горизонтали и на две по вертикали (как конь).

Ваша задача — найти число способов расставить на доске N на N ровно K магараджей так, чтобы они не били друг друга.

Входные данные

Входной файл INPUT.TXT содержит два целых числа: N и K (1 ≤ K ≤ N ≤ 10).

Выходные данные

В выходной файл OUTPUT.TXT выведите ответ на задачу.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1 | 9 |
| 2 | 4 2 | 20 |
| 3 | 5 3 | 48 |

### **Вариант 78**

Для доступа в лаборатории НИИ Исследований Данных Строк используются ключи в виде прямоугольных карточек N×M, в которых вырезаны дырки. Эти ключи можно вставлять только одним способом (то есть ни поворачивать, ни переворачивать нельзя). При этом дырки имеют прямоугольную форму. К Васе попало два ключа от разных лабораторий. Он решил их наложить друг на друга так, чтобы получившаяся фигура имела максимальное количество дырок (просветов). При этом исходно ключи лежали в том положении, в котором их необходимо вставлять в замок, а Вася не хочет их поворачивать. Помогите Васе определить максимальное число дырок. При наложении считаются только те дырки, внутренности которых не пусты.

Входные данные

Первая строка входного файла INPUT.TXT содержит два целых числа - 1 ≤ N, M ≤ 109 - длины сторон ключа. Вторая строка содержит единственное целое число - 1 ≤ K1 ≤ 500 - число дырок в первом ключе. Далее в K1 строках написано по четыре целых числа - X1, Y1, X2, Y2 (0 ≤ X1 < X2 ≤ N, 0 ≤ Y1 < Y2 ≤ M) – координаты углов соответствующих прямоугольных дырок. Дырки в ключе не пересекаются и не касаются.

Далее следует описание второго ключа в таком же формате.

Выходные данные

В выходной файл OUTPUT.TXT выведите единственное целое число - максимальное количество дырок, которое может получить Вася.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 10 10 1 1 1 2 2 1 1 1 2 2 | 1 |
| 2 | 10 10 2 1 1 2 2 3 3 4 4 1 1 1 2 2 | 1 |
| 3 | 10 10 2 1 1 2 2 3 3 4 4 1 1 1 3 3 | 2 |

### **Вариант 79**

«Игра в 24» - это математическая игра, в которой используются специальные карточки. На каждой карточке записаны четыре числа. Задача игроков состоит в том, чтобы получить число 24, используя эти числа и арифметические операции (сложение, вычитание, умножение и деление, скобки при этом можно расставить произвольным образом). «Игра в 24» используется в некоторых школах при изучении математики в начальных классах.

В этой задаче рассматривается упрощенный вариант этой игры, в котором в число разрешенных операций входят только сложение, вычитание и умножение.

Назовем карточку для упрощенной «Игры в 24» правильной, если из указанных на ней чисел с помощью сложения, вычитания, умножения и расстановки скобок произвольным образом можно получить число 24.

Фирма American Card Manufacturer (ACM) занимается выпуском наборов карточек для этой игры. Однако, выпуск таких карточек сопряжен с некоторыми трудностями. Одна из них состоит в том, что не любой набор из четырех чисел задает «правильную» карточку.

По этой причине задача проверки «правильности» данной карточки является весьма актуальной. Ваша задача состоит в написании программы, которая будет осуществлять указанную проверку.

Входные данные

Входной файл INPUT.TXT содержит четыре натуральных числа, не превосходящих 30, которые написаны на исследуемой карточке.

Выходные данные

В выходной файл OUTPUT.TXT выведите слово YES, если карточка является правильной, и слово NO в противном случае.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 1 2 3 4 | YES |
| 2 | 1 1 1 1 | NO |

### **Вариант 80**

Для участия в соревнованиях по перетягиванию каната зарегистрировалось N человек. Некоторые из участников могут быть знакомы друг с другом. Причем, если двое из них имеют общего знакомого, то это не означает, что они обязательно знакомы друг с другом.

Организаторы соревнований заинтересованы в их качественном проведении. Они хотят разделить всех участников на две команды так, чтобы в первой команде было K человек, а во второй – N-K человек. Из всех возможных вариантов формирования команд, организаторы хотят выбрать такой вариант, при котором сумма сплоченностей обеих команд максимальна. Сплоченностью команды называется количество пар участников этой команды, знакомых друг с другом. Ваша задача – помочь организаторам найти требуемое разделение участников на две команды.

Входные данные

В первой строке входного файла INPUT.TXT задаются три числа N, K, M, разделенные одиночными пробелами, где N – общее число зарегистрированных участников, K – требуемое количество человек в первой команде, M – количество пар участников, знакомых друг с другом.

Каждая из следующих M строк содержит два различных числа, разделенные пробелом – номера двух участников, знакомых друг с другом. Все участники нумеруются от 1 до N.

Ограничения: все числа целые, 0 < K < N < 25, 0 ≤ M ≤ N(N-1)/2

Выходные данные

Выходной файл OUTPUT.TXT должен содержать одну строку, состоящую из K чисел, каждое из которых задает номер участника, попавшего в первую команду. Числа должны быть разделены пробелами. Если существует несколько решений данной задачи, то выведите любое из них.

Пример

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

### **Вариант 81**

Рассмотрим слова, состоящие из первых n букв английского алфавита. Морфизм – это функция f, которая по букве возвращает слово. Рассмотрим пример морфизма: f(A) = ABC, f(B) = A, f(C) = BC.

Если мы рассмотрим слово w = c1c2...cl и применим к нему морфизм f, мы получим слово f(w) = f(c1)f(c2)...f(cl). Например, для морфизма из предыдущего параграфа f(ABC) = ABCABC.

Мы можем применять морфизм к слову несколько раз. Положим f0(w) = w, и для k > 0 положим fk(w) = f(fk−1(w)).

По заданному морфизму f, слову w, числу k и числу p, найдите p-й символ слова fk(w).

Входные данные

Первая строка входного файла INPUT.TXT содержит числа n, k и p (1 ≤ n ≤ 10, 0 ≤ k ≤ 109,1 ≤ p ≤ 20). Вторая строка входного файла содержит слово w. Его длина не превышает 50. Следующие n строк содержат f(A), f(B), и т.д. Каждое значение – это строка, содержащая от 1 до 50 символов.

Выходные данные

В выходной файл OUTPUT.TXT выведите p-й символ fk(w), или «-» (минус) - если такой символ отсутствует.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 3 1 5 ABC ABC A BC | B |
| 2 | 3 1 7 ABC ABC A BC | - |

### **Вариант 82**

В последнее время Петя очень заинтересовался одной игрой. Вкратце правила таковы: Имеется море размером (2•n+1) x (2•n+1) клеток. Изначально в некоторых клетках находятся острова, в некоторых – вражеские корабли, в центре стоит корабль игрока. За один ход игрок может либо переместиться, либо выстрелить; пропустить ход нельзя.

Перемещение производится на одну из восьми соседних клеток, при условии, что она существует и свободна. При этом считается, что корабль сначала поворачивается на месте носом к выбранной клетке, а затем двигается в нее.

При выстреле корабль производит залп обоими бортами. Ядра при этом летят перпендикулярно текущему курсу корабля. Ядра каждого борта поражают первую встретившуюся в данном направлении цель (корабль или остров), но не далее трех клеток от корабля игрока (не считая клетку, на которой находится сам корабль игрока). Например, если корабль приплыл из клетки (2, 1) в клетку (1, 2), а затем выстрелил, то он уничтожит вражеский корабль на клетке (4, 5), но не поразит корабль на (5, 6). Если к тому же на клетке (2, 3) будет стоять корабль или остров, то корабль на (4, 5) останется цел. Изначально корабль игрока стоит на клетке (n, n) с таким направлением, будто он приплыл из клетки (n−1, n).

После каждого хода игрока вражеские корабли одновременно делают ход. Каждый из кораблей ходит на ту из восьми соседних клеток, сумма модулей разностей координат которой и координат корабля игрока минимальна. Формально, если корабль игрока находится на клетке (xs, ys), то выбирается такая клетка (x, y), для которой |xs−x|+|ys−y| минимально. Если при этом оказывается, что в данной клетке находится остров, то вражеский корабль погибает. Если хотя бы один вражеский корабль попадает на клетку с кораблем игрока, то игрок проигрывает. Если хотя бы два вражеских корабля оказываются на одной клетке, то они погибают и на этой клетке образуются обломки, которые далее действуют так же, как остров за тем исключением, что снаряды перелетают через обломки, а не поражают их. При уничтожении корабля ядром тоже образуются обломки. На месте острова обломки не образуются. Игрок побеждает, если на поле нет ни одного живого вражеского корабля.

Пете стало очень интересно, можно ли в каждой конкретной ситуации выиграть или нет. Для этого он решил написать программу. Нет, Вам не надо ему помогать. Он с этой задачей уже справился, а вот справитесь ли вы?

Входные данные

В первой строке входного файла INPUT.TXT задано число n (1 ≤ n ≤ 6), определяющее размер поля. Далее следуют 2•n+1 строк по 2•n+1 символов в каждой. При этом i-й символ (j+1)-й строки описывает клетку поля с координатами (i−1, j−1) (клетки нумеруются с 0). Значение символов следующее:

«.» (точка) – пустая клетка;

«t» (t маленькое английское) – вражеский корабль;

«O» (o большое английское) – остров;

«+» (плюс) – корабль игрока.

Символ «+» всегда присутствует и располагается только в клетке с координатами (n×n). Количество остальных символов может быть любым, и ограничено только размерами доски.

Выходные данные

В первой строке выходного файла OUTPUT.TXT выведите количество ходов, необходимых для выигрыша. В последующих строках выведите координаты корабля после каждого из ходов, по одной паре в строке. Минимизировать число ходов не обязательно; тем не менее, если вражеских кораблей уже не осталось, дальнейшие ходы делать не следует. Если решения не существует, выведите IMPOSSIBLE в единственной строке файла. Если существует несколько решений, выведите любое.

Примеры

|  |  |  |
| --- | --- | --- |
| № | INPUT.TXT | OUTPUT.TXT |
| 1 | 5 ..........t .......t... O......t... t.......... .........tO ..t..+..... ........... ...O.....Ot ...O....... ....t...... ..t..O..... | IMPOSSIBLE |
| 2 | 5 ....t..O.O. ....t...... ........... ........... ........... ..O..+..... ........... ........... ........... .t......... .....O..... | 7 5 5 5 5 5 5 5 4 5 3 6 2 5 1 |

### **Вариант 83**

В рамках подготовки к чемпионату мира Кирилл придумал Ане задачу. Он написал N знаковых 32-битных чисел и попросил вычислить значение некоторого выражения S. Пусть a1, …, aN - все эти числа. Тогда выражение это

S = (a1 xor a2 xor … xor an) xor (b1 xor b2 xor … xor bn-1),

где

bi = F(ai, ai+1) xor F(ai, ai+2) xor … xor F(ai, an).

В этой формуле под знаком xor понимается побитовое «исключающее или», а F(a, b) = x - 1, где x - максимальная степень двойки, на которую делится нацело a-b, если a ≠ b, и F(a, b) = -1, если a = b. Все операции xor выполняются слева направо, если скобки не указывают иной порядок.

Аня, как большая специалистка в области циклов, быстро написала требуемую программу, однако программа работала слишком долго. Чтобы лучше разобраться в этом вопросе, она попросила вас написать программу, которая бы укладывалась в отведенное время. Помогите ей это сделать.

Входные данные

В первой строке входного файла INPUT.TXT содержится число N (1 ≤ N ≤ 105). В следующих N строках содержится N 32-битных знаковых целых чисел ai по одному на строке.

Выходные данные

В выходной файл OUTPUT.TXT выведите значение выражения.

Примеры

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