Задача 1.

Задан набор не повторяющихся пар (A_i,A_j), A_i, A_j принадлежат множеству $A=\{A_1, A_2, ..., A_n\}$. Необходимо составить цепочку максимальной длины по правилу

$$(A_i,A_j)+(A_j,Ak)=(A_i,A_j,A_k).$$

При образовании этой цепочки любая пара может быть использована не более одного раза.

Задача 2.

Между N пунктами (N<=50) заданы дороги длиной A(i,j), где I,J-номера пунктов. Дороги проложены на разной высоте и пересекаются только в общих пунктах. В начальный момент времени из заданных пунктов начинают двигаться с постоянной скоростью M роботов (M=2 или 3), независимо меняя направление движения только в пунктах. Роботы управляются таким образом, чтобы минимизировать время до встречи всех роботов в одном месте. Скорость I-того робота может быть равна 1 или 2. Остановка роботов запрещена.

Задание:

Написать программу, которая:

1) при заданных N,M и сети дорог единичной длины (все имеющиеся A(i,j)=1) определяет минимальное время, через которое может произойти встреча всех M роботов, при этом начальное положение роботов и скорость их движения известны.

2) Выполнить те же действия, что и в п. 1, но только для различных значений A(i,j).

Примечание: В случае невозможности встречи всех M роботов в одном месте ни в какой момент времени в результате выполнения программы должно быть сформировано соответствующее сообщение.

Требование к вводу-выводу:

1) Все входные данные - целые неотрицательные числа;

2) при задании сети дорог должно быть указано количество дорог - K и пункты их начала и конца в виде пар (i,j).

Задача 3.

На плоскости расположено N точек. Имеется робот, который двигается следующим образом. Стартуя с некоторой начальной точки и имея некоторое начальное направление, робот движется до первой встреченной на его пути точки, изменяя в ней свое текущее направление на 90 градусов, т.е. поворачивая налево или направо. После этого он продолжает движение аналогично. Если робот достиг начальной точки, либо не может достичь новой точки (которую он еще не посещал), то он останавливается.

Определить, может ли робот посетить все N точек, если:

1. Определены начальные точка и направление робота.
2. Определена начальная точка, а направление робота можно выбирать.
3. Начальную точку и направление робота можно выбирать.

Координаты точек - целые числа, угол измеряется в радианах относительно оси OX.

Задача 4.

"ПУТЬ".

Найти кратчайшее расстояние между двумя вершинами в графе. Найти все возможные пути между этими двумя вершинами в графе не пересекающиеся по

а) ребрам

б) вершинам.

Задача 5.

Лабиринт задается матрицей смежности N*N, где C(i,j)=1, если узел i связан узлом j посредством дороги. Часть узлов назначается входами, часть - выходами. Входы и выходы задаются последовательностями узлов X(1),..,X(p) и Y(1),..,Y(k) соответственно.

Найти максимальное число людей, которых можно провести от

входов до выходов таким образом, чтобы:

а) их пути не пересекались по дорогам, но могут пересекаться по узлам;

б) их пути не пересекались по узлам;

Задача 6.

N шестеренок пронумерованы от 1 до N (N<=10). Заданы M (0<=M<=45) соединений пар шестеренок в виде (i,j), 1<=i<j<=N (шестерня с номером i находится в зацеплении с шестерней j). Можно ли повернуть шестерню с номером 1?

Если да, то найти количество шестерен, пришедших в движение.

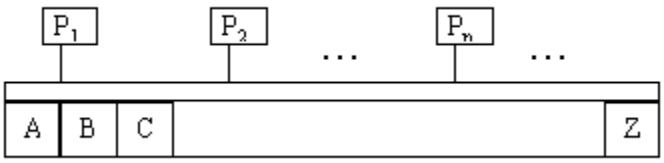
Если нет, то требуется убрать минимальное число шестерен так, чтобы в оставшейся системе при вращении шестерни 1 во вращение пришло бы максимальное число шестерен. Указать номера убранных шестерен (если такой набор не один, то любой из них) и количество шестерен, пришедших в движение.

Задача 7.

На фигуре 1.показана вычислительная система, содержащая достаточное количество процессоров, использующих общую память из 26 числовых переменных A,B,C,....,Z. Система работает шагами. На каждом шаге, каждый процессор выполняет либо оператор присваивания либо пустой оператор. Оператор присваивания - это конструкция вида

(переменная)=(арифметическое выражение)

где арифметическое выражение без скобок и содержит не более 2 переменных и арифметические операции. Процессоры вычисляют выражения и присваивают их значения переменным из левых частей операторов, а потом приступают к следующим операторам (при том одновременно). Не допускается одновременное выполнение 2 или больше операторов присваивания с одинаковой левой частью. Пустой оператор обозначаем символом &. Выполняя его, процессор простаивает 1 шаг.



Фиг. 1

n последовательности операторов (присваивания или пустых) с одинаковой длиной L называем n-программой с длиной L (выполняется за L шагов на первых n процессоров), если на каждом шаге имеем не более 1 оператора с заданной левой частью. n-программы P и Q называем эквивалентными, если начиная с одного и того же начального состояния переменных A,B,...,Z после выполнения как P, так и Q получается одинаковый результат.

Напишите программу, которая:

Задание 1. Вводит целое k(k<25) и 1-программу, содержащую k операторов присваивания.

Задание 2. Проверяет правильность введенных операторов. Задание 3. Преобразует 1-программу в эквивалентную m-программу с минимальной длиной (добавляя если надо пустые операторы) и выводит полученный результат.

Задание 4. Не увеличивая длину построенной в Задании 3 n-программы, преобразует ее в эквивалентную m-программу, m-как можно меньше, и выводит полученный результат.

Замечание. На фигуре 2 показана 1-программа из 6 операторов и 3-программа и 2-программа - возможные решения задач 3(б) и 4(в).

а)	D=A*D	б)	A=B+C	B=C+D	D=D-E
	A=B+C		A=A-E	&	&
	A=A-E		E=A*B	D=A*D	&
	B=C+D	в)	A=B+C	B=C+D	
	D=D-E		A=A-E	D=D-E	
	E=A*B		E=A*B	D=A*D	

Фиг. 2

Задача 8.

Имеется N прямоугольных конвертов и N прямоугольных открыток различных размеров. Можно ли разложить все открытки по конвертам, чтобы в каждом конверте было по одной открытке.

Замечание. Открытки нельзя складывать, сгибать и т.п., но можно помещать в конверт под углом. Например, открытка с размерами сторон 5:1 помещается в конверты с размерами 5:1, 6:3, 4.3:4.3, но не входит в конверты с размерами 4:1, 10:0.5, 4.2:4.2.

Задача 9.

Составить программу для нахождения произвольного разбиения 20 студентов на 2 команды, численность которых отличается не более чем в 2 раза, если известно, что в любой команде должны быть студенты, обязательно знакомые друг с другом. Круг знакомств задается матрицей (20,20) с элементами

A(ij)={1,если i студент знаком с j

{0,иначе .

Задача 10.

Имеется N человек и прямоугольная таблица A[1:N,1:N];элемент A[i,j] равен 1, если человек i знаком с человеком j, A[i,j] = =A[j,i]. Можно ли разбить людей на 2 группы, чтобы в каждой группе были только незнакомые люди.

Задача 11.

На олимпиаду прибыло N человек. Некоторые из них знакомы между собой. Можно ли опосредованно перезнакомить их всех между собой? (Незнакомые люди могут познакомиться только через общего знакомого).

Задача 12.

Пусть группа состоит из N человек. В ней каждый имеет (N/2) друзей и не больше K врагов. У одного из них есть книга, которую все хотели бы прочитать и потом обсудить с некоторыми из остальных.

Написать программу, которая:

1. Находит способ передачи книги таким образом, чтобы она побывала у каждого в точности один раз, переходя только от друга к другу и наконец возвратилась к своему владельцу.

2.Разбивает людей на S групп, где будет обсуждаться книга, таким образом, чтобы вместе с каждым человеком в ту же самую группу вошло не более P его врагов.

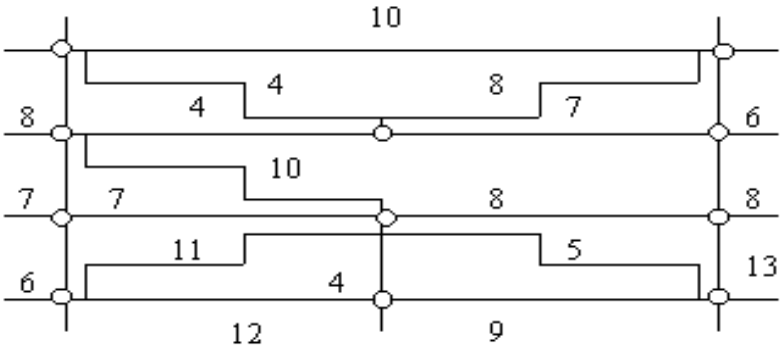
Примечание: предполагается, что S*P>=K.

Задача 13.

В заданном графе необходимо определить, существует ли цикл, проходящий по каждому ребру графа ровно один раз.

Задача 14.

Следующая фигура показывает запутанную сеть дорог района города. Представьте, что мусорная машина должна пройти по всем дорогам хотя бы один раз, чтобы собрать мусор. Число на каждой стороне показывает время, которое должна потратить мусорная машина, чтобы проехать этот интервал. На перекрестках машина должна ждать время, равное числу пересекающихся дорог.



Составьте программу, показывающую как выбрать необходимый путь для сбора мусора в кратчайшее время.

Есть 11 остановок.

от 1 до 2 путь 10 мин.

от 1 до 3 4

от 1 до 4 8

от 2 до 3 8

от 2 до 5 6

от 3 до 4 4

от 3 до 5 7

от 4 до 7 7

от 4 до 6 10

от 4 до 8 7

от 8 до 6 7

от 8 до 10 6

от 10 до 6 11

от 6 до 9 4

от 10 до 9 12

от 6 до 11 5

от 6 до 4 8

от 5 до 4 8

от 4 до 11 13

от 9 до 11 5

Задача 15.

N различных станков один за другим объединены в конвейер. Имеется N рабочих. Задана матрица C[N , N], где C[i,j] производительность i-ого рабочего на j-ом станке. Определить

а) на каком станке должен работать каждый из рабочих, чтобы производительность была максимальной.

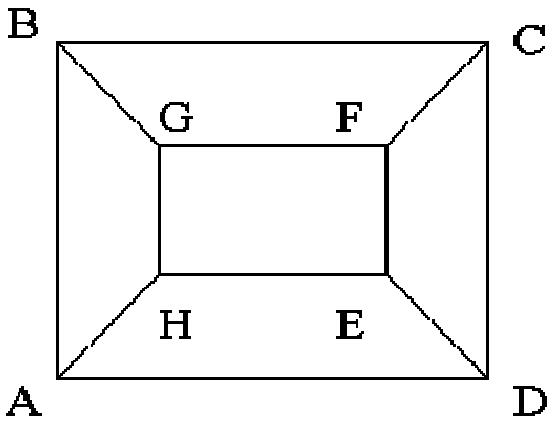
б) то же, но станки расположены параллельно и выполняют однородные операции.

Задача 16.

На плоскости задан граф с N вершинами. Количество ребер, соединенных с каждой

вершиной, равно 3.

Пример:



Пусть вершины X,Y и Z являются соседями вершины T. Будем считать, что Y левый, а Z - правый сосед вершины T относительно вершины X, если ориентированный угол XTZ меньше ориентированного угла XTY (положительным будем считать направление против часовой стрелки). Например вершина E является правым соседом вершины H относительно A, а G - левым, поскольку ориентированный угол AHE меньше ориентированного угла AHG. (Ребра считаются отрезками).

Составьте программу, которая:

/(ПЕРВЫЙ ПУНКТ ВЫПОЛНЯТЬ НЕ НУЖНО)*1. Вводит координаты вершин графа и его ребра и рисует граф на экране компьютера, производя при этом подходящее масштабирование(Ребра выводятся как отрезки).()*/

2. Пусть заданы две начальные соседние вершины X_0 и X_1 и последовательность вида LLRRL... Тогда программа находит путь на графе X_0 X_1 X_2...X_n для вершин которого выполнено:

-первые два являются заданными X_0 и X_1

-X_{i+1} является левым или правым соседом X_i относительно X_{i-1} в зависимости от заданной последовательности, при том L означает левый, а R -правый.

Пример:В заданном графе пусть даны начальные вершины A и H и последовательность LRRLLR. Тогда программа должна найти путь AHGFEDCB.

3. Рисует на экране путь, найденный в п.2.

4. Пусть даны начальная и конечная вершина. Программа должна найти путь, проходящий через минимальное число вершин, вывести его на экран и найти 2 первые вершины и

управляющую последовательность для этого пути, как определено в п.2.

Задача 17.

Имеется N городов. Для каждой пары городов (I,J) можно построить дорогу, соединяющую эти два города и не заходящие в другие города. Стоимость такой дороги A(I,J). Вне городов дороги не пересекаются.

Написать алгоритм для нахождения самой дешевой системы дорог, позволяющей попасть из любого города в любой другой. Результаты задавать таблицей B[1:N,1:N], где B[I,J]=1 тогда и только тогда, когда дорогу, соединяющую города I и J, следует строить.

Задача 18.

В картинной галерее каждый сторож работает в течение некоторого непрерывного отрезка времени. Расписанием стражи называется множество пар $[T_1(i), T_2(i)]$ - моментов начала и конца дежурства i-го сторожа из интервала [0,EndTime].

Для заданного расписания стражи требуется:

(а) проверить, в любой ли момент в галерее находится не менее двух сторожей.

Если условие (а) не выполнено, то:

(б) перечислить все интервалы времени с недостаточной охраной (менее 2 сторожей).

(в) добавить наименьшее число сторожей с заданной, одинаковой для всех длительностью дежурства, чтобы получить правильное расписание (т.е. удовлетворяющее условию (а)).

(г) проверить, можно ли обойтись без добавления новых сторожей, если разрешается сдвигать времена дежурства каждого из сторожей с сохранением длительности дежурства.

(д) Если это возможно, то составить расписание с наименьшим числом сдвигов.

ВХОДНЫЕ ДАННЫЕ:

(Все моменты времени задаются в целых минутах.)

EndTime - момент окончания стражи, т.е. охраняется отрезок времени [0, EndTime].

N -число сторожей.

$T_1[i]$, $T_2[i]$, i=1,..N - моменты начала и окончания дежурства i-го сторожа.

Length - длительность дежурства каждого дополнительного сторожа.

ВЫХОДНЫЕ ДАННЫЕ:

(1) Ответ на пункт (а) в форме да/нет.

(2) При ответе "нет" на п. (а) - список пар (k,l) - начал и концов всех малоохраняемых интервалов с указанием числа сторожей в каждом (0 или 1).

(3) Число дополнительных сторожей и моменты начала и окончания дежурства каждого дополнительного сторожа.

(4) Ответ на пункт (г) в форме да/нет. Если "да", то номера сторожей, смена которых сдвигается, и значения сдвигов.

(5) В ответ на пункт (д): наименьшее число сторожей, смена которых сдвигается, их номера и значения сдвигов.

ПРИМЕЧАНИЕ

Программа должна допускать независимое тестирование пунктов (в), (г), (д).

Задача 19.

Вводится N - количество домов и К - количество дорог. Дома пронумерованы от 1 до N. Каждая дорога определяется тройкой чисел - двумя номерами домов - концов дороги и длиной дороги. В каждом доме

живет по одному человеку. Найти точку - место встречи всех людей, от которой суммарное расстояние до всех домов будет минимальным.

Если точка лежит на дороге, то указать номера домов - концов этой дороги и расстояние от первого из этих домов. Если точка совпадает с домом, то указать номер этого дома.

Примечание: длины дорог - положительные целые числа.

Задача 20.

N колец сцеплены между собой (задана матрица A(n*n), A(i,j)=1 в случае, если кольца i и j сцеплены друг с другом и A(i,j)=0 иначе). Удалить минимальное количество колец так,

чтобы получилась цепочка.

Задача 21.

Янка положил на стол N выпуклых K-гранников и N различных типов наклеек по K штук каждая. Ночью кто-то наклеил наклейки на грани, по одной на грань.

Помогите Янке расставить многогранники так, чтобы наклейка каждого типа была видна ровно K-1 раз.

Задача 22.

Имеется N точек и M проводков. Проводком можно соединить некоторую пару различных точек, причем пара может быть соединена не более чем одним проводком. Все проводки должны быть использованы.

Пусть D_i - количество проводков, которые будут соединены с точкой с номером i, i=1, ..., N.

Необходимо соединить N точек с помощью M проводков таким образом, чтобы сумма $S=D_1*D_1 + D_2*D_2 + ... + D_n*D_n$ была максимальной.

Вывести величины D_i в неубывающем порядке и. по требованию

(priznak=1), список соединений.

ВВОД:
<Введите N:> N (N<=100)
<Введите M:> M (M<=1000)
<PRIZNAK=> PRIZNAK
ВЫВОД:
<Результирующая конфигурация:> D_i в неубывающем порядке.
<Сумма S> S
<Список соединений>
<Точку 1 соединить с> список точек
.....
<Точку N соединить с> список точек

Задача 23.

Задано N городов с номерами от 1 до N и сеть из M дорог с односторонним движением между ними. Каждая дорога задается тройкой (i, j, k), где i - номер города, в котором дорога начинается, j -

номер города, в котором дорога заканчивается, а k - ее длина (число k - натуральное). Дороги друг с другом могут пересекаться только в концевых городах.

Все пути между двумя указанными городами A и B можно упорядочить в список по неубыванию их длин (если есть несколько путей одинаковой длины, то выбираем один из них). Необходимо найти один из путей, который может быть вторым в списке.

Вывести его длину L и города, через которые он проходит.
ВВОД:
<Количество городов N:> N <Количество дорог M:> M
<Начало, конец и длина дороги 1:> i_1, j_1, k_1
.....
<Начало, конец и длина дороги M:> i_M, j_M, k_M
<Города A и B, между которыми надо найти путь:> A, B
ВЫВОД:
<Пути нет>
или
<Путь проходит по городам> A, i_1, i_2, ..., B
<Длина пути> L