№3 Издание основано в 1995 г inf.1september.ru
УЧЕБНО-МЕТОДИЧЕСКИЙ ЖУРНАЛ ДЛЯ УЧИТЕЛЕЙ ИНФОРМАТИКИ

ИНФОРМАТИКА

внутри номера CD и код доступа к электронной версии

издательский дом 1september.ru
Первое сентября
март 2012
ИНФОРМАТИКА Подписка: «Роспечать» — 32291 (бумажная версия), 19179 (электронная); «Почта России» — 79066 (бумажная версия), 12684 (электронная)

НА ОБЛОЖКЕ

Пузырек — легендарная личность в мире информатики. Мало кто из коллег не шутил с детьми про алгоритм, названный именем видного ученого Пузырька. Кстати, не так мало ребят сначала не воспринимают эти слова как шутку. А что — есть сортировка фон Неймана, есть сортировка Пузырька ☺. Алгоритм пузырьковой сортировки очень узнаваем по характерному фрагменту кода, в котором заключена основная идея алгоритма, — если два соседних элемента расположены не в требуемом порядке, их нужно поменять местами. Вот, собственно, и вся идея. Зато какая изящная и понятная!

В НОМЕРЕ

НА ДИСКЕ

ЭЛЕКТРОННЫЕ МАТЕРИАЛЫ:

- Практикумы в среде КуМир и презентация к статье про исполнитель “Водолей”
- Исходные коды программ к статье о задаче ЕГЭ С2
- Презентации к остальным материалам номера, включая раздел “В мир информатики”

ИНФОРМАТИКА

ПОДПИСНЫЕ ИНДЕКСЫ: по каталогу “Роспечати”: 32291 (бумажная версия), 19179 (электронная версия); “Почта России”: 79066 (бумажная версия), 12684 (электронная версия)

http://inf.1september.ru

Учебно-методический журнал для учителей информатики
Основан в 1995 г.
Выходит один раз в месяц

РЕДАКЦИЯ:
гл. редактор С.Л. Островский
редакторы
Е.В. Андреева,
Д.М. Златопольский (редактор вкладки “В мир информатики”)
Дизайн макета И.Е. Лукьянов
верстка Н.И. Пронская
корректор Е.Л. Володина
секретарь Н.П. Медведева
Фото: фотобанк Shutterstock
Журнал распространяется по подписке
Цена свободная
Тираж 15509 экз.
Тел. редакции: (499) 249-48-96
E-mail: inf@1september.ru
http://inf.1september.ru

ИЗДАТЕЛЬСКИЙ ДОМ “ПЕРВОЕ СЕНТЯБРЯ”
Главный редактор: Артем Соловейчик (генеральный директор)
Коммерческая деятельность: Константин Шмарковский (финансовый директор)
Развитие, IT и координация проектов: Сергей Островский (исполнительный директор)
Реклама, конференции и техническое обеспечение Издательского дома: Павел Кузнецов
Производство: Станислав Савельев
Административно-хозяйственное обеспечение: Андрей Ушков
Главный художник: Иван Лукьянов
Педагогический университет: Валерия Арсланьян (ректор)

ГАЗЕТА ИЗДАТЕЛЬСКОГО ДОМА
Первое сентября – Е.Бирюкова

ЖУРНАЛЫ ИЗДАТЕЛЬСКОГО ДОМА
Английский язык – А.Громушкина
Библиотека в школе – О.Громова
Биология – Н.Иванова
География – О.Коротова
Дошкольное образование – Д.Тюттерин
Здоровье детей – Н.Сёмина
Информатика – С.Островский
Искусство – М.Сартан
История – А.Савельев
Классное руководство и воспитание школьников – О.Леонтьева
Литература – С.Волков
Математика – Л.Рослова
Начальная школа – М.Соловейчик
Немецкий язык – М.Бузоева
Русский язык – Л.Гончар
Спорт в школе – О.Леонтьева
Управление школой – Е.Рачевский
Физика – Н.Козлова
Французский язык – Г.Чесновицкая
Химия – О.Блохина
Школьный психолог – И.Вачков

УЧРЕДИТЕЛЬ: ООО “ЧИСТЫЕ ПРУДЫ”

Зарегистрировано ПИ № ФС77-44341 от 22.03.2011 в Министерстве РФ по делам печати
Подписано в печать: по графику 10.02.2012, фактически 10.02.2012
Заказ №
Отпечатано в ОАО “Чеховский полиграфический комбинат” ул. Полиграфистов, д. 1, Московская область, г. Чехов, 142300

АДРЕС ИЗДАТЕЛЯ: ул. Киевская, д. 24, Москва, 121165
Тел./факс: (499) 249-31-38
Отдел рекламы: (499) 249-98-70
http://1september.ru

ИЗДАТЕЛЬСКАЯ ПОДПИСКА:
Телефон: (499) 249-47-58
E-mail: podpiska@1september.ru

Просто графы

К.Ю. Поляков,
д. т. н., Санкт-Петербург

1. Введение

В последние годы в школьный курс информатики стремительно вошла тема “Графы”, что связано прежде всего с включением соответствующих задач в варианты ЕГЭ и ГИА. Тема эта глубокая и интересная, имеет много серьезных практических приложений (транспортные перевозки, проектирование сетей коммуникаций, маршрутизация в Интернете, разводка печатных плат).

С другой стороны, изучение графов с помощью классических вузовских учебников [1–3] в течение нескольких уроков — дело достаточно бессмысленное. Потому что за короткое время удается только познакомиться с довольно сложной (и не до конца устоявшейся) терминологией и очертить круг задач, которые решаются с помощью графов. Чтобы изучить методы решения этих задач, многие из которых NP-полные (которые пока решаются только перебором всех вариантов), нужно потратить не один десяток часов и написать не одну программу, которая значительно длиннее “школьных” 30–40 строчек.

Таким образом, необходимо сделать выбор — изучать графы за несколько уроков на уровне общих понятий или посвятить глубокому изучению графов достаточно большое количество времени (в объеме семестрового курса вуза).

Кроме того, имеет смысл разделять два варианта значения слова “граф”:

1) удобная форма описания структур типа дорожной сети или сети передачи данных;

2) математический объект $G := (V, E)$, где V — это непустое множество вершин, а E — множество ребер (пар вершин).

Очевидно, что в школьном курсе информатики (по крайней мере на базовом уровне) “математическое” понятие графа не требуется (так же, как, например, в базовом курсе не изучается строгое определение термина “алгоритм”, использующее понятие универсального исполнителя типа машины Тьюринга).

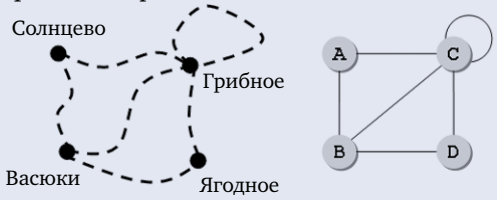
Задача настоящей статьи — представить минимальный материал, необходимый для решения задач ЕГЭ и ГИА на графы (задачи А2, В9 в демоварианте ЕГЭ-2012 [4], задачи 3 и 11 в демоварианте ГИА-2012 [5]), для освоения которого достаточно 2–3 уроков. Приводятся несколько способов решения этих задач, так что каждый может выбрать то, что ему по душе. Кроме того, рассмотрение объекта с разных сторон способствует пониманию его сущности.

Начнем с основных понятий, связанных с графами.

2. Описание графа

Итак, граф — это (непустое) множество вершин и множество соединяющих их ребер. Эта структура естественно появляется тогда, когда в задаче есть несколько однотипных объектов и каждый из них может быть связан с произвольным количеством других объектов. В качестве объектов могут быть железнодорожные станции, маршрутизаторы локальных и глобальных сетей и т.п. В первом случае ребра графа — это дороги между станциями, во втором — каналы связи (кабельные, оптоволоконные, спутниковые и т.д.).

Для человека наиболее естественно использовать изображения графов (рисунки, схемы) для объяснения связей между элементами какой-то системы. Поскольку информатика занимается автоматической обработкой данных с помощью компьютеров, сразу возникает вопрос: “Как представить информацию о графе в памяти компьютера?” Хранить ее в виде рисунка (растрового или векторного) неэффективно, потому что рисунок предназначен для восприятия человеком, а не компьютером. Компьютеру удобнее всего хранить информацию в виде таблиц (массив тоже можно считать простейшей таблицей). Для описания графа часто используют квадратную таблицу, которая описывает все возможные связи между узлами (без учета дублирования). Если, например, на пересечении строки **A** и столбца **B** записано число 1, это означает, что есть ребро, соединяющее вершины **A** и **B**; число 0 в этой ячейке означает, что такого ребра нет. Такую таблицу называют *матрицей смежности*. На рисунке показаны схема дорог, соответствующий ей граф и его матрица смежности:

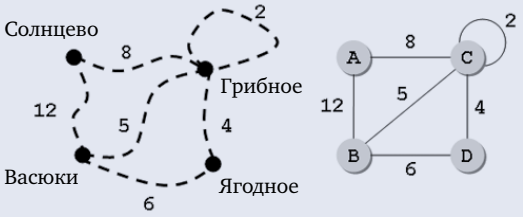


	A	B	C	D
A	0	1	1	0
B	1	0	1	1
C	1	1	1	1
D	0	1	1	0

Единица на главной диагонали (выделенной серым цветом) показывает, что в графе есть *петля* — ребро, которое начинается и заканчивается в одной и той же вершине.

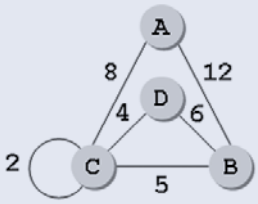
Обратите внимание, что матрица смежности симметрична относительно главной диагонали, то есть если существует ребро из вершины **A** в вершину **B**, то существует и ребро из **B** в **A**. Такой граф называют *неориентированным* — ребра не имеют направления и каждое из них учтено в матрице смежности дважды.

Во многих практических задачах (например, при определении кратчайшего пути из одной вершины в другую) важную роль играет не только наличие связей между вершинами, но и “длины” этих связей, которые в теории графов называют “весами” (от слова “вес”). Весом может быть, например, длина дороги или стоимость авиаперелета. В этом случае вместо матрицы смежности используют *весовую матрицу*, в клетках которой записывают веса ребер, а если ребра нет, то клетку оставляют пустой. На рисунке показана схема, на которой указаны длины дорог, соответствующий ей граф и его весовая матрица:



	A	B	C	D
A		12	8	
B	12		5	6
C	8	5	2	4
D		6	4	

Заметим, что весовая матрица никак не определяет взаимное расположение вершин. Например, рассмотренный выше граф можно нарисовать совсем иначе, например, так:



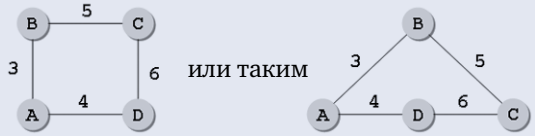
Однако с точки зрения теории графов это будет тот же самый граф, поскольку весовая матрица не изменилась. По аналогии можно вспомнить, что в математике записи 0,5, 1/2, 3/6 и 5/10 обозначают одно и то же число.

Что можно выяснить с помощью весовой матрицы? Во-первых, определить, есть ли ребро между двумя заданными вершинами, и если есть, какова его длина (вес). Для этого достаточно посмотреть в соответствующую ячейку. Например, значение, выделенное кружком на рисунке, показывает, что между вершинами **B** и **C** есть ребро с весом 5:

	A	B	C	D
A		3		4
B	3		5	
C		5		6
D	4		6	

Предполагая, что веса ребер обозначают расстояния между вершинами, можно определить длину пути, проходящего через заданные вершины, — сумму длин ребер, составляющих этот путь. В данном случае длина замкнутого пути **ABCDA** складывается из длин ребер **AB**, **BC**, **CD** и **DA**, которые определяются по таблице. Получаем: 3 + 5 + 6 + 4 = 18.

Наконец, полезно научиться рисовать граф по заданной весовой матрице. Нужно только помнить, что это можно сделать разными способами. Для последней приведенной матрицы рисунок может быть, например, таким:



или таким

После того как освоены основные понятия, можно переходить непосредственно к задачам, которые включены в ЕГЭ и ГИА.

Вопросы и задачи:

1) Как по весовой матрице графа определить количество ребер (количество петель)?

2) Как можно обозначить отсутствие связи между вершинами при хранении весовой матрицы в памяти реального компьютера (рассмотрите разные варианты)?

3) Для каждой приведенной ниже весовой матрицы:

- определите вес ребра между вершинами **B** и **D** (если оно есть);
- предполагая, что веса ребер обозначают расстояния между вершинами, определите длину пути **ABDCEA**;
- укажите, какой из трех путей — **ABDC**, **ADEC** или **AEBC** — самый короткий, а какой самый длинный:

а)

	A	B	C	D	E
A		4	3	8	7
B	4		4	2	2
C	3	4		6	3
D	2	2	6		1
E	7	8	3	1	

б)

	A	B	C	D	E
A		2	5	7	6
B	2		1	3	4
C	5	1		9	8
D	7	3	9		1
E	6	4	8	1	

в)

	A	B	C	D	E
A		1	2	3	9
B	1		5	4	1
C	2	5		5	6
D	3	4	5		7
E	9	1	6	7	

г)

	A	B	C	D	E
A		5	2	1	6
B	5		3	7	2
C	2	3		4	9
D	1	7	4		8
E	6	2	9	8	

3. Кратчайший путь

Одна из самых известных задач, связанных с графами, — определение длины кратчайшего пути из одной вершины в другую. Человек, знакомый с теорией графов, сразу скажет, что нужно использовать алгоритм Дейкстры [1–3, 6], однако на самом деле это не обязательно. Алгоритм Дейкстры действительно хорош, но он предназначен для *автоматического* решения задач этого типа. Автоматическое решение требуется, когда

- исходные данные заранее неизвестны (поступают из другого источника, например, весовая матрица передается в виде файла);
- необходимо решать множество однотипных задач;
- необходимо решать сложные задачи (для матриц с большим числом узлов и ребер).

Если же нужно решить одну простую задачу, применение универсального, но непростого алгоритма Дейкстры — это, как говорят, “из пушки по воробьям”. Кроме того, большинство школьников не помнят наизусть алгоритм Дейкстры, даже если он изучался в профильном курсе информатики. Поэтому мы будем говорить о простых методах решения, которые доступны всем, кто изучает информатику даже на базовом уровне.

Рассмотрим задачу № 3 из демонстрационного варианта ГИА [5].

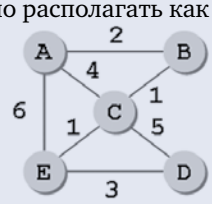
Между населенными пунктами **A**, **B**, **C**, **D**, **E** *построены дороги, протяженность которых приведена в таблице. Определите кратчайший путь между пунктами* **A** *и* **D** *(при условии, что передвигаться можно только по построенным дорогам).*

	A	B	C	D	E
A		2	4		6
B	2		1		
C	4	1		5	1
D		1	5		3
E	6		1	3	

Уточним, что на самом деле в задаче требуется определить не сам кратчайший путь (последовательность ребер), а его длину. В условии дано еще 4 варианта ответа, но в данном случае они только мешают, поэтому мы их не приводим. Хотя задача сформулирована как практическая, она сводится к поиску кратчайшего пути в графе.

Способ 1. *Построение графа.*

По заданной таблице (весовой матрице графа) определяем, что граф содержит 7 ребер: **AB** (длиной 2), **AC** (4), **AE** (6), **BC** (1), **CD** (5), **CE** (1) и **DE** (3). Нарисуем граф, соответствующий этим данным (вершины можно располагать как угодно):

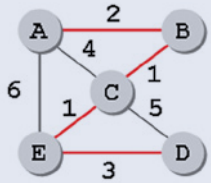


Теперь перебираем все возможные пути из вершины **A** в вершину **D**, не проходящие дважды через одну и ту же вершину, и считаем их длины:

ABCD: 2 + 1 + 5 = 8
ABCED: 2 + 1 + 1 + 3 = 7
ACD: 4 + 5 = 9
ACED: 4 + 1 + 3 = 8
AECD: 6 + 1 + 5 = 12
AED: 6 + 3 = 9

Чтобы не запутаться и не потерять какой-то путь, мы сначала рассмотрели все пути, начинающиеся с ребра **AB**, затем — все пути, начинающиеся с ребра **AC**, и т.д. Более того, пути перебираются в так называемом *лексикографическом порядке*, то есть в таком порядке, в каком они были бы расположены в словаре (в данном случае — в английском). Такой порядок делает перебор систематическим и поэтому уменьшает вероятность пропуска какого-то пути.

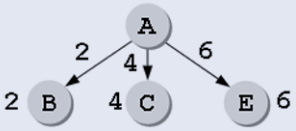
Сравнивая полученные длины, находим, что кратчайший путь **ABCED** (он выделен красным цветом) имеет длину 7:



Прелесть этого способа состоит в том, что в простых задачах можно сразу увидеть ответ, перебрав все варианты в уме и таким образом сэкономив время. Однако при этом есть риск пропустить какой-то вариант (на это и рассчитаны отвлекающие варианты ответов — *дистракторы*).

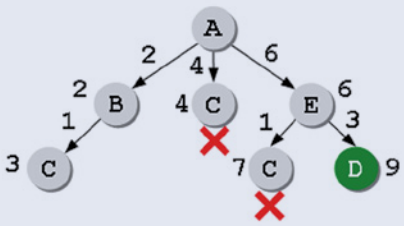
Способ 2. *Построение дерева возможных путей.*

Можно использовать другой способ, при котором явно строится схема возможных путей в графе (структура типа "*дерево*"). Сначала по таблице определяем, что из исходной вершины **A** можно ехать только в **B**, **C** и **E**:



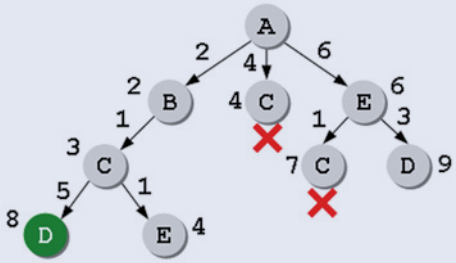
Числа около стрелок обозначают длины дорог (веса соответствующих ребер), а числа около вершин — расстояния от начальной вершины **A** по этому пути.

Далее рассматриваем все пути, состоящие из двух ребер: из **B** можно ехать в **C** (или вернуться в **A**, но такой вариант нас не интересует), из **C** — в **B**, **D** и **E**, а из **E** — в **C** и **D**. Однако первый же из рассмотренных путей, **ABC**, имеет длину 3, что меньше, чем длина ребра **AC** (4). Поэтому все пути, начинающиеся с ребра **AC**, можно далее вообще не рассматривать — в любом случае переезд из **A** в **C** через **B** будет на 1 короче, чем напрямую. По этой же причине не нужно рассматривать пути, начинающиеся с **AEC** (длина этого пути равна 7, что больше уже известного пути в **C** длиной 3):

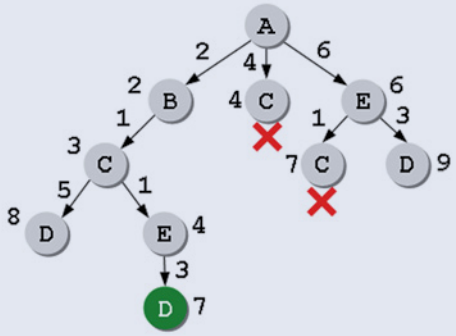


Итак, мы нашли один путь (**AED**) из **A** в **D**, который имеет длину 9. Пока запоминаем его, однако нужно рассмотреть еще ветку **ABC**, которая может дать более короткий путь.

По таблице находим, что из **C** можно ехать в **B**, **D** и **E**. Возвращаться в **B** нет смысла, поэтому рассматриваем последние два варианта:



Получаем еще один (лучший на данный момент!) путь **ABCD** из **A** в **D** длиной 8. С другой стороны, длина пути **ABCE** меньше, чем длина предыдущего известного пути из **A** в **E** (путь **AE**, длина 6), поэтому на этой ветке, возможно, удастся еще улучшить результат. И действительно, из **E** имеет смысл ехать только в **D** (возвращаться в **A** или в **C** не стоит), и этот путь имеет длину 7:



Таким образом, мы построили *дерево*, которое учитывает все возможные пути. Некоторые ветки дерева (**AC** и **AEC**) отсечены, потому что эти пути заведомо не улучшают результат. Длина кратчайшего пути **ABCED** равна 7.

Способ 3. *Перебор возможных путей без построения дерева.*

Сначала выпишем все ребра, соединяющие начальную вершину **A** с другими вершинами, и их длины:

AB: 2
AC: 4
AE: 6

Теперь рассмотрим все пути, состоящие из двух ребер. Получив путь **ABC** длиной 3, сразу отбрасываем все пути, проходящие через ребро **AC** (так же, как в способе 2):

ABC: 2 + 1 = 3
AEC: 6 + 1 = 7
AED: 6 + 3 = 9 (цель достигнута)

Видим, что все пути, проходящие через **AEC**, тоже можно отбросить. Остается проверить ветку **ABC**:

ABCD: 3 + 5 = 8 (цель достигнута)
ABCE: 3 + 1 = 4

Продолжая последнюю оставшуюся ветку **ABCE**, находим:

ABCED: 4 + 3 = 7 (цель достигнута)

Нерассмотренных путей, которые могли бы улучшить решение, больше не осталось, поэтому выбираем лучший путь: **ABCED** длиной 7.

Способ 4. *Использование алгоритма Дейкстры* [1–3, 6].

Здесь мы не будем приводить детальное описание алгоритма Дейкстры, которое можно найти в указанной литературе, а просто покажем, как с его помощью можно решить предложенную задачу.

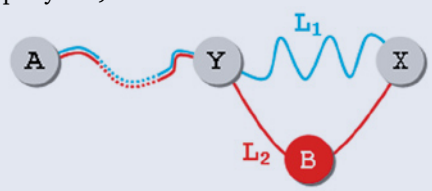
Составляем таблицу, в первой строке выписываем все вершины, кроме начальной вершины **A**; во вторую строку заносим длины ребер, соединяющих начальную вершину с данной (если такое ребро есть), а в третью записываем имя начальной вершины:

B	C	D	E
2	4		6
A	A	A	A

Первая строка меняться не будет, поэтому она выделена цветом. Во второй строке будет храниться длина кратчайшего (на данный момент) пути из **A** в соответствующую вершину. Нижняя строка хранит предпоследнюю вершину на этом пути. Пока мы учли только пути, состоящие из одного ребра, поэтому в нижней строке везде указана вершина **A**. Постепенно мы будем расширять множество “учтенных” путей и в конце концов найдем длину действительно кратчайшего пути в каждую вершину.

На каждом этапе работы алгоритма Дейкстры окончательно определяется длина кратчайшего пути до одной из вершин. Сначала ищем минимальное значение во всей второй строке таблицы — это число 2 для вершины **B**. Можно доказать, что это и есть истинная длина кратчайшего пути из **A** в **B**.

Проверяем, не удастся ли сократить путь, если ехать в остальные вершины через вершину **B**. Здесь используется очень простая идея: дорога из некоторой вершины **Y** в другую вершину **X** через **B** может оказаться короче, чем напрямую. Например, предположим, что среди всех рассмотренных ранее путей из **A** в **X** лучшим является путь, включающий ребро **YX**, и его длина равна L_1 (синяя линия на рисунке).



Теперь “подключаем” вершину **B**, и может оказаться, что сумма длин дорог **YB** и **BX** будет меньше, чем длина дороги **YX**, поэтому длина L_2 пути через **B** (красная линия) будет меньше, чем L_1.

В нашем случае, “подключая” вершину **B**, мы рассматриваем пути, в которых последние звенья — это **BC**, **BD** и **BE**. В данном графе ребер **BD** и **BE** нет, поэтому нужно лишь проверить, не будет ли путь **ABC** короче, чем **AC**. Оказывается, это действительно так: вместо того чтобы ехать по прямой дороге из **A** в **C** (ее длина 4), лучше ехать из **A** в **B**, а затем из **B** в **C** (общая длина пути 3). Поэтому длина кратчайшего пути во второй строчке меняется на 3, а предпоследняя вершина в третьей строке — на **B**:

B	C	D	E
2	3		6
A	B	A	A

На следующем этапе вершину **B** уже не рассматриваем (серый фон означает, что этот столбец далее не будет изменяться):

B	C	D	E
2	3		6
A	B	A	A

Среди незакрашенных клеток второй строки ищем минимальное значение — это 3 для вершины **C**. Проверяем, не позволяет ли объезд через **C** сократить путь. Во-первых, появляется путь из **ACD**, его длина равна 4 + 5 = 9:

B	C	D	E
2	3	9	6
A	B	C	A

Кроме того, вместо пути **AE** (длиной 6) можно ехать сначала в **C** (длина 3), а затем — в **E** (длина 1), то есть фактически использовать путь **ABCE** длиной 3 + 1 = 4:

B	C	D	E
2	3	9	4
A	B	C	C

Теперь вершина **C** тоже исключается из рассмотрения, из оставшихся вершин значение во второй строке минимально для вершины **E**:

B	C	D	E
2	3	9	4
A	B	C	C

Если ехать из **A** сначала в **E** (длина 4), а затем по дороге **ED** (длина 3), то общая длина пути оказывается равна 7, и это позволяет улучшить результат: кратчайший путь из **A** в **D** имеет длину 7:

B	C	D	E
2	3	7	4
A	B	E	C

В этой задаче не требуется определять сам путь (важна только его длина), но в третьей строке таблицы есть все данные для этого. Путь “раскручивается” с конечной вершины **D**: в третьей строке видим, что в нее мы попадаем из **E**. Смотрим далее: в **E** приезжаем из **C**, в **C** — из **B**, а в **B** — из **A**. Поэтому кратчайший путь — **ABCED** длиной 7.

Задачи для тренировки

Между населенными пунктами **A**, **B**, **C**, **D**, **E**, **F** построены дороги, протяженность которых приведена в таблице. Отсутствие числа в ячейке таблицы означает, что прямой дороги между пунктами нет. Определите длину кратчайшего пути между пунк-

тами **A** и **F** (при условии, что передвигаться можно только по построенным дорогам).

а) (демонстрационный вариант ЕГЭ, ответ: 9)

	A	B	C	D	E	F
A		2	4			
B	2		1		7	
C	4	1		3	4	
D			3		3	
E		7	4	3		2
F					2	

б) (ответ: 17)

	A	B	C	D	E	F
A		5				
B	5		9	3	8	
C		9			4	
D		3			2	
E		8	4	2		7
F					7	

в) (ответ: 14)

	A	B	C	D	E	F
A		4				
B	4		6	3	6	
C		6			4	
D		3			2	
E		6	4	2		5
F					5	

г) (источник — *http://ege.yandex.ru*, ответ: 15)

	A	B	C	D	E	F
A			3			
B			9		4	
C	3	9		3	8	
D			3		2	
E		4	8	2		7
F					7	

д) (ответ: 18)

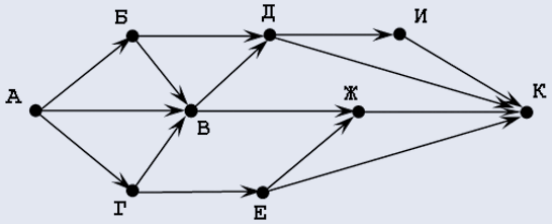
	A	B	C	D	E	F
A			2			
B			6		3	
C	2	6		5	7	
D			5		4	
E		3	7	4		9
F					9	

3. Количество путей

В этом году в ЕГЭ и ГИА введена совсем новая задача, в которой требуется определить количество различных путей из одной вершины в другую. Как показали первые обсуждения, она представляет сложность даже для многих учителей информатики, не говоря уже про учеников.

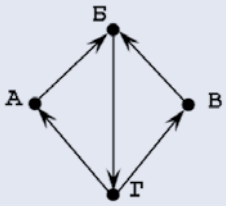
Рассмотрим задачу B9 из демонстрационного варианта ЕГЭ [4]:

На рисунке — схема дорог, связывающих города **А**, **Б**, **В**, **Г**, **Д**, **Е**, **Ж**, **И**, **К**. *По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города* **А** *в город* **К**?



Если говорить на языке теории графов, в этой задаче используется *ориентированный* граф (или кратко *орграф*) — по каждому ребру можно двигаться только в одном направлении, указанном стрелкой (аналогия из жизни — дорога с односторонним движением). Отметим, что если предположить, что движение возможно в обоих направлениях, количество путей будет бесконечно (можно сколько угодно ездить туда и обратно между соседними городами).

Тогда возникает вопрос — а для любого ли ориентированного графа количество путей из одной вершины в другую конечно? Простой пример показывает, что это не так:



Действительно, здесь есть циклы **АБГ** и **БГВ**, в которых можно "крутиться" бесконечно долго. Таким образом, для разрешимости задачи необходимо, чтобы в графе не было циклов.

Теперь займемся решением. Число вершин в таких задачах достаточно велико (здесь — 9), поэтому матрица смежности размером 9×9 при ручном решении нам вряд ли поможет.

Способ 1. *Перебор всех возможных путей "методом наблюдения".*

Попробуем найти все возможные пути, глядя на схему. Сначала рассмотрим пути, начинающиеся с отрезка **АБ**. Чтобы не запутаться, будем начинать с верхней части графа, постепенно "спускаясь" ниже и ниже:

АБДИК, АБДК, АБВДИК, АБВДК, АБВЖК

Теперь выпишем пути, которые начинаются с ребра **АВ** (также сверху вниз):

АВДИК, АВДК, АВЖК

и, наконец, пути, начинающиеся с участка **АГ**:

АГВДИК, АГВДК, АГВЖК, АГЕЖК, АГЕК.

Всего получилось 13 путей. Это наиболее простой метод, но он нередко приводит к неправильному результату, потому что легко пропустить какой-то путь.

Способ 2. *Пошаговое построение всех возможных путей.*

Сначала рассмотрим все вершины, куда можно попасть из начальной вершины **А** за один шаг:

АБ
АВ
АГ

Теперь выписываем все пути из вершины **А**, содержащие два ребра (продолжения уже построенных путей на одно ребро):

АБД АБВ
АВД АВЖ
АГВ АГЕ

Следующий шаг (в рамку обведены пути, дошедшие до конечного пункта **К**):

АБДИ **АБДК** **АБВД АБВЖ**
АВДИ **АВДК** **АВЖК**
АГВД АГВЖ АГЕЖ **АГЕК**

Итак, четыре пути из **А** в **К** мы уже нашли. Продолжая движение по остальным маршрутам, находим семь путей, состоящих из четырех ребер:

АБДИК **АБВДИ** **АБВДК** **АБВЖК**
АВДИК
АГВДИ **АВГДК** **АГВЖК** **АГЕЖК**

и далее два пути из пяти ребер:

АБВДИК
АГВДИК

Таким образом, всего найдено 4 + 7 + 2 = 13 путей, причем по построению других путей нет (в списке не осталось путей, не доведенных до конечной точки).

На взгляд автора, этот способ более надежен, чем предыдущий, поскольку на каждом шаге нас интересуют только ребра, исходящие из одной вершины. Это значит, что нужно анализировать не весь граф, а только его небольшую часть, поэтому труднее ошибиться.

К сожалению, оба уже рассмотренных способа плохо работают, если граф сложный и количество путей велико (несколько десятков). В этом случае приходится применять "тяжелую артиллерию" — методы, которые не требуют перечисления всех возможных путей, но позволяют найти их количество "малой кровью". В их основе лежит следующая довольно простая и очевидная идея: если в некоторую вершину **X** можно попасть только из вершин Y_1, Y_2, …, Y_M, то количество путей из начальной точки **А** в **X** вычисляется как

$$N_X = N_{Y_1} + N_{Y_2} + \ldots N_M, \qquad (*)$$

где N_Z обозначает количество путей из вершины **А** в некоторую вершину **Z**.

Способ 3. *Подстановка.*

Для начала заметим, что для вершин, в которые можно приехать только из **А**:

$$N_Б = N_Г = 1.$$

Применим формулу (*) для конечной вершины **К**, учитывая, что в **К** можно попасть только из **И**, **Д**, **Ж** и **Е**:

$$N_К = N_И + N_Д + N_Ж + N_Е.$$

Теперь нужно записать аналогичные выражения для слагаемых в правой части и подставить их в предыдущую формулу. Повторяя эту процедуру несколько раз, мы сведем правую часть к арифметическому выражению, включающему только известные величины $N_Б$ и $N_Г$.

Применяем формулу (*) для вершин **И**, **Д**, **Ж** и **Е**:
$N_И = N_Д$, $N_Д = N_Б + N_В$, $N_Ж = N_В + N_Е$, $N_Е = N_Г$.
Подставляем эти выражения в формулу для $N_К$:
$N_К = N_Д + N_Д + N_Ж + N_Е =$
$= 2(N_Б + N_В) + N_В + 2N_Е = 2N_Б + 3N_В + 2N_Г.$
Поскольку $N_Б = N_Г = 1$, сразу получаем
$N_К = 4 + 3N_В$.
Вычисляем $N_В$, снова применяя формулу (*) и учитывая прямой путь из **А**:

$$N_В = 1 + N_Б + N_В = 3,$$

поэтому $N_К = 4 + 3 \cdot 3 = 13$.

Обратите внимание, что использованный здесь подход (подстановка) фактически сводит решение исходной задачи (вычисление $N_К$) к решению нескольких более простых задач того же типа (вычислению $N_И$, $N_Д$, $N_Ж$ и $N_Е$) до тех пор, пока мы не придем к простейшим задачам с известными решениями ($N_Б = N_Г = 1$). Такой прием, называемый *динамическим программированием* [6], применяется также при решении задачи С3 из демонстрационного варианта ЕГЭ-2012 [4].

Способ 4. *Подстановка-2.*

Попробуем систематизировать применение подстановки и тем самым уменьшить вероятность случайной ошибки. Для этого выпишем для каждой вершины список всех вершин, из которых можно в нее попасть:

Б ← А **В ← АБГ** **Г ← А**
Д ← БВ **Е ← Г** **Ж ← ВЕ**
И ← Д **К ← ИДЖЕ**

Теперь составляем таблицу: в первом столбце будем записывать имя вершины, во втором — из каких вершин можно в нее попасть, и в третьем — количество путей из начальной вершины в данную. Очевидно, что в третьем столбце можно сразу поставить единицы для тех вершин, в которые можно попасть только из **А** (это **Б** и **Г**):

Вершина	Предыдущие	*N*
Б	А	1
В	АБГ	
Г	А	1
Д	БВ	
Е	Г	
Ж	ВЕ	
И	Д	
К	ИДЖЕ	

Далее ищем все вершины, куда можно доехать через вершины с уже известным количеством путей (**А**, **Б** и **Г**), — это **В** и **Е**:

Вершина	Предыдущие	*N*
Б	А	1
В	АБГ	3
Г	А	1
Д	БВ	
Е	Г	1
Ж	ВЕ	
И	Д	
К	ИДЖЕ	

Теперь можно определить количество путей для вершин **Д** и **Ж**

Вершина	Предыдущие	*N*
Б	А	1
В	АБГ	3
Г	А	1
Д	БВ	4
Е	Г	1
Ж	ВЕ	4
И	Д	
К	ИДЖЕ	

затем — для вершины **И** и, наконец, для конечной вершины **К**:

Вершина	Предыдущие	*N*
Б	А	1
В	АБГ	3
Г	А	1
Д	БВ	4
Е	Г	1
Ж	ВЕ	4
И	Д	4
К	ИДЖЕ	13

Таким образом, правильный ответ — 13 путей.

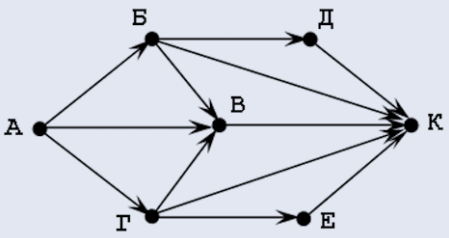
Мы заполняли таблицу в определенном порядке, каждый раз “выискивая” вершины, для которых мы уже можем подсчитать количество путей (то есть для всех предыдущих вершин количество путей уже известно). Фактически при этом выполняется *топологическая сортировка* вершин графа [7], но изучать ее на школьном уровне совершенно необязательно.

Отметим еще раз, что нам не потребовалось выписывать сами пути, поэтому этот метод хорошо работает и для сложных графов с большим количеством путей.

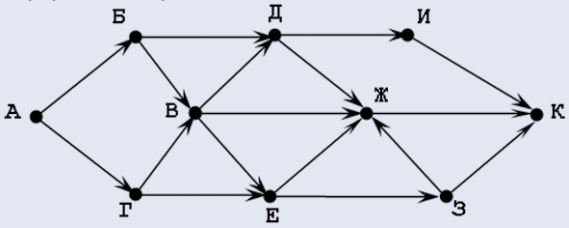
Задачи для тренировки

На рисунке показана схема дорог. По каждой дороге можно двигаться только в одном направлении, указанном стрелкой. Сколько существует различных путей из города **А** в город **К**?

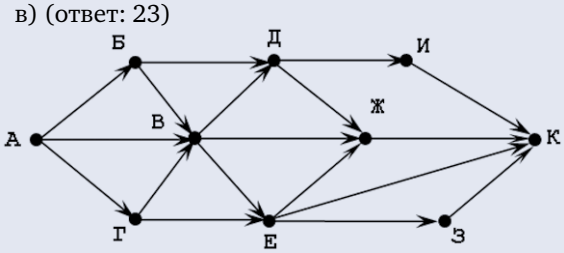
а) (демонстрационный вариант ГИА, ответ: 7)



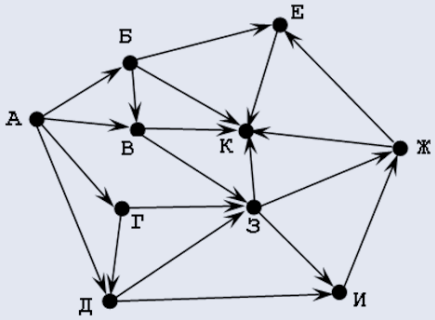
б) (ответ: 17)



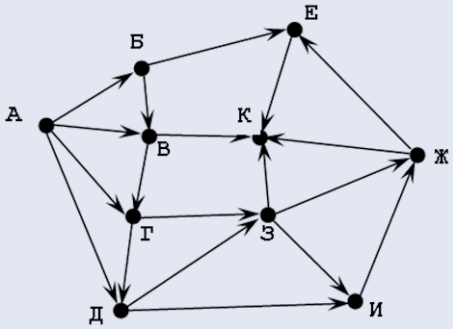
в) (ответ: 23)



г) (ответ: 33)



д) (ответ: 46)



Автор благодарит д. ф.-м. н. М.А. Ройтберга за полезное обсуждение данного материала и ценные замечания.

Литература

1. *Кристофидес Н.* Теория графов. Алгоритмический подход. М.: Мир, 1978.

2. *Оре О.* Теория графов. М.: Наука, 1968.

3. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы: построение и анализ. Часть VI. Алгоритмы для работы с графами. М.: Вильямс, 2006.

4. Демонстрационный вариант КИМ для проведения ЕГЭ 2012 года по информатике (*http://www.fipi.ru/view/sections/222/docs/578.html*).

5. Демонстрационный вариант КИМ для проведения ГИА 2012 года по информатике (*http://www.fipi.ru/view/sections/223/docs/579.html*).

6. *Поляков К.Ю., Шестаков А.П., Еремин Е.А.* Алгоритмизация и программирование // Информатика, № 17, 2011, с. 4–33.

7. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К.* Алгоритмы: построение и анализ. Глава 22.4. Топологическая сортировка. М.: Вильямс, 2006.