I.4. Структура информации

I.4.1. Зачем структурировать информацию?

Давайте сравним четыре информационных сообщения.

Первое:

«Для того, чтобы добраться до села Васино, нужно сначала долететь на самолете до Ивановска. Затем на электричке доехать до Ореховска. Там на пароме переправиться через реку Слоновую в поселок Ольховка, и оттуда ехать в Васино на попутной машине».

Второе:

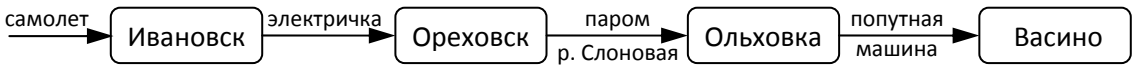
Как ехать в Васино?

1) *На самолете до Ивановска.*
2) *На электричке до Ореховска.*
3) *На пароме через р. Слоновую в пос. Ольховка.*
4) *На попутной машине до с. Васино.*

Третье:

Откуда	Куда	Транспорт
Москва	Ивановск	самолет
Ивановск	Ореховск	электричка
Ореховск	пос. Ольховка	паром (р. Слоновая)
пос. Ольховка	с. Васино	попутная машина

Четвертое:



Можно считать, что все эти (такие разные по форме!) сообщения содержат одну и ту же информации. Какие из них проще воспринимать? Очевидно, что «вытащить» полезную информацию из простого текста (первое сообщение) сложнее всего. Во втором случае мы сразу видим все этапы поездки. Третье сообщение (таблицу) и четвертое (схему) можно понять сразу, с первого взгляда. Второй, третий и четвертый варианты лучше и быстрее воспринимаются, потому что в них выделена *структура* информации, в которой самое главное – этапы поездки в Васино.

Структура (лат. *structura* — строение) – это внутреннее устройство чего-либо.

Почему книгу разбивают на главы и разделы, а не пишут сплошной текст? Зачем в тексте выделяют абзацы? Прежде всего, для того, чтобы подчеркнуть основные мысли – каждая глава, раздел, абзац содержат определенную идею. При таком выделении структуры улучшается передача информации от автора к читателю.

Кроме того, есть еще и другая причина – облегчить **поиск** нужной информации. В книгах чаще всего есть оглавление, позволяющее быстро найти нужный раздел. Слова в словарях всегда расставлены в алфавитном порядке (представьте себе, что было бы, если бы они были расположены произвольно!). В больших книгах используют *индексы* – списки основных терминов с указанием страниц, на которых они встречаются.

Оглавление:

Словарь:

автомат – *automaton*
автор – *author*
адрес – *address*
алгебра – *algebra*
алгоритм – *algorithm*
архив – *archive*
архитектура – *architecture*

Индекс:

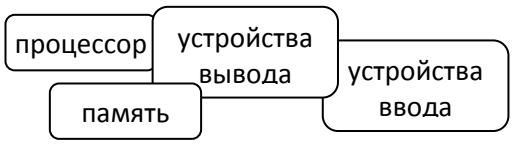
Структурирование – это выделение важных элементов в информационных сообщениях и установление связей между ними. Цели структурирования – облегчение восприятия и поиска информации, выявление закономерностей.

I.4.2. Простые структуры

Простейшая структура – это **множество**, то есть, некоторый набор элементов. Чтобы определить множество, мы должны перечислить все его элементы (например, множество, состоящее из Васи, Пети и Коли) или определить характерный признак, по которому элементы включаются в это множество (множество драконов с пятью зелеными хвостами).

Множество может состоять из конечного числа элементов (множество букв русского алфавита), бесконечного числа элементов (множество натуральных чисел) или вообще быть пустым (множество слонов, живущих на Северном полюсе). В документах конечное множество часто оформляют в виде маркированного списка, например:

- процессор;
- память;
- устройство ввода;
- устройства вывода.

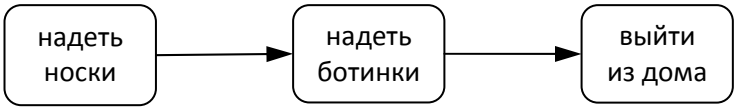


В таком списке порядок элементов не важен, от перестановки элементов множество не меняется.

Если множество состоит из конечного числа элементов и его элементы должны быть расположены в строго определенном порядке, мы получаем **линейный список**. Список обычно упорядочен (отсортирован) по какому-то правилу, например, по алфавиту, по важности, по последовательности действий и т.д. В тексте он оформляется как нумерованный список, например:

1) надеть носки;
2) надеть ботинки;
3) выйти из дома.

Переставить местами элементы такого списка нельзя (это будет уже другой список). Линейный список может быть представлен в виде цепочки связанных элементов:

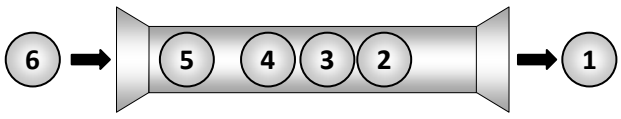


Список можно задать перечислением элементов, с первого до последнего:

(надеть носки, надеть ботинки, выйти из дома)

Теперь предположим, что нам нужно добавлять и удалять элементы в линейном списке. Знакомый вам пример – это *очередь* в кассу в магазине. Действительно, очередь – это цепочка, в которой элементы нельзя переставлять местами (если так случается, то кто-то лезет без очереди).

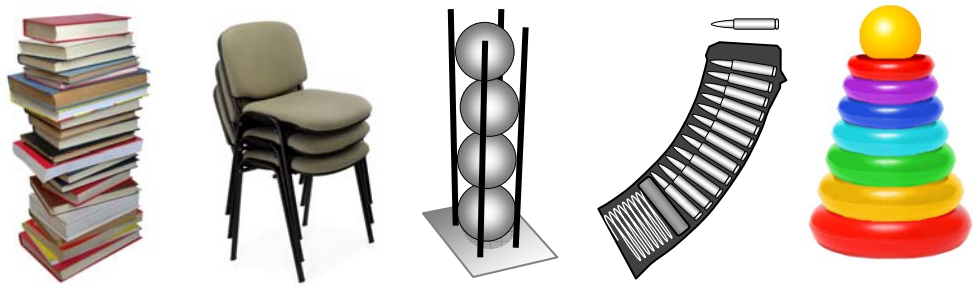
Очередь – это линейный список, в котором элементы добавляются с одного конца, а удаляются с другого («первым пришел – первым ушел»[13]).



[13] Англ. FIFO = *First In – First Out*.

Очередь можно представить в виде трубы, с одного конца которой добавляются шарики, а с другого – вынимаются. Трамваи, стоящие на перекрестке, – это тоже пример очереди, они не могут обогнать друг друга.

Возможен и другой вариант, когда элементы добавляются и удаляются с одного и того же конца списка. Это значит, что тот, кто пришел последним, уйдет первым. Такая структура называется **стек**. Например, стопка с книгами или автоматный магазин:



Во всех этих примерах для того, чтобы добраться до нужного объекта, нам нужно сначала удалить все те, что расположены выше него.

Стек – это линейный список, в котором элементы добавляются и удаляются только с одного конца («последний пришел – первым ушел»[14]).

Более общий случай – это список, в котором добавление и удаление элементов разрешается с обоих концов. Такая структура называется **дек.**

Еще одна знакомая вам структура – **таблица**. С помощью таблиц устанавливается связь между несколькими элементами. Например, в следующей таблице элементы в каждой строке связаны между собой – это свойства некоторого объекта (человека):

Фамилия	Имя	Рост, см	Вес, кг	Год рождения
Иванов	Иван	175	67	1996
Петров	Петр	164	70	1998
Сидоров	Сидор	168	63	2000

Именно так хранится информация в базах данных: строка таблицы, содержащая информацию об одном объекте, называется *записью*, а столбец (название свойства) – *полем*.

Возможен и другой вариант таблицы, когда роли строк и столбцов меняются. В первом столбце записываются названия свойств, а данные в каждом из следующих столбцов описывают свойства какого-то объекта. Например, вот таблица с характеристиками разных марок автомашин:

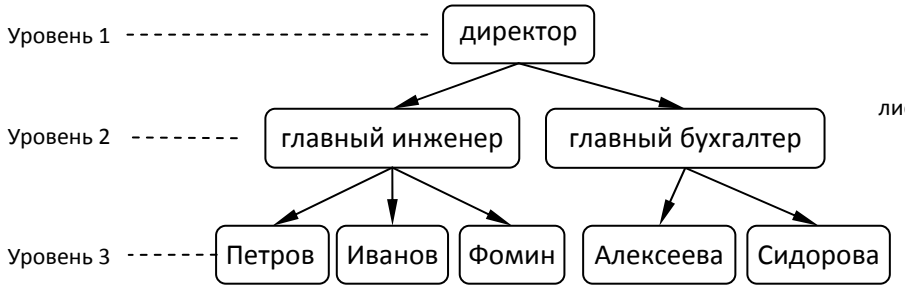
Марка	Лада Приора	Лада Калина	ВАЗ 2110	ВАЗ 21099
Мощность, л.с.	98	89	89	81
Максимальная скорость, км/ч	183	165	187	167,5
Время разгона до 100 км/ч, с	12	12,5	12	13,5

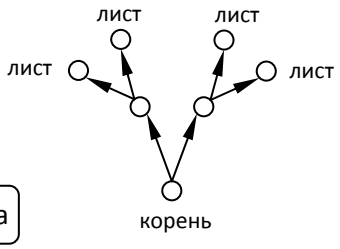
В математике и программировании таблицы обычно называют *матрицами*.

I.4.3. Иерархия (дерево)

Линейных списков и таблиц иногда недостаточно для того, чтобы представить все связи между элементами. Например, в некоторой фирме есть директор, ему подчиняются главный инженер и главный бухгалтер, у каждого из них есть свои подчиненные. Если мы захотим нарисовать схему управления этой фирмы, она получится *многоуровневой*.

[14] Англ. LIFO = *Last In – First Out*.



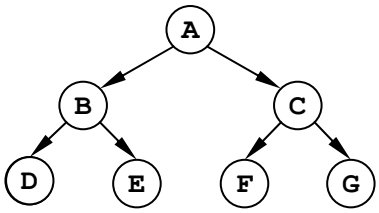


Такая структура, в которой одни элементы «подчиняются» другим, называется *иерархия* (от древнегреческого ἱεραρχία – «священное правление»). В информатике иерархию называют **деревом**. Дело в том, что если перевернуть эту схему вверх ногами, она становится похожа на дерево (точнее, на куст, см. рисунок справа). Несколько деревьев образуют **лес**.

Дерево состоит из узлов и связей между ними (они называются дугами). Самый первый узел, расположенный на верхнем уровне (в него не входит ни одна стрелка-дуга) – это *корень дерева.* Конечные узлы, из которых не выходит ни одна дуга, называются *листьями*. Все остальные узлы, кроме корня и листьев – это промежуточные узлы.

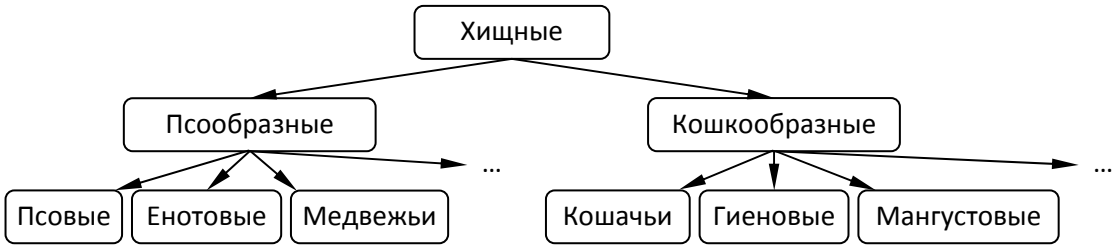
Из двух связанных узлов тот, который находится на более высоком уровне, называется «*родителем*», а другой – «*сыном*». Корень – это единственный узел, у которого нет «родителя»; у листьев нет «сыновей».

Используются также понятия «предок» и «потомок». «Потомок» какого-то узла – это узел, в который можно перейти по стрелкам от узла-предка. Соответственно, «предок» какого-то узла – это узел, из которого можно перейти по стрелкам в данный узел.



В дереве на рисунке справа родитель узла E – это узел B, а предки узла E – это узлы A и B, для которых узел E – потомок. Потомками узла A (корня) являются все остальные узлы.

Типичный пример иерархии – различные *классификации* (животных, растений, минералов, химических соединений). Например, отряд *Хищные* делится на два подотряда: *Псообразные* и *Кошкообразные*. В каждом из них выделяют несколько семейств:



Конечно, на этой схеме показаны не все семейства, остальные обозначены многоточием.

В текстах иерархию часто оформляют в виде многоуровневого списка. Например, оглавление книги о хищниках может выглядеть так:

Глава 1. Псообразные

1.1 Псовые

1.2 Енотовые

1.3 Медвежьи

…

Глава 2. Кошкоообразные

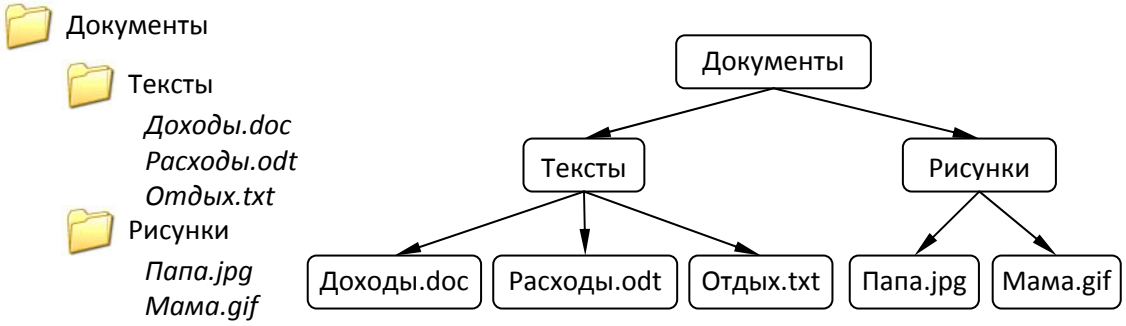
2.1 Кошачьи

2.2 Гиеновые

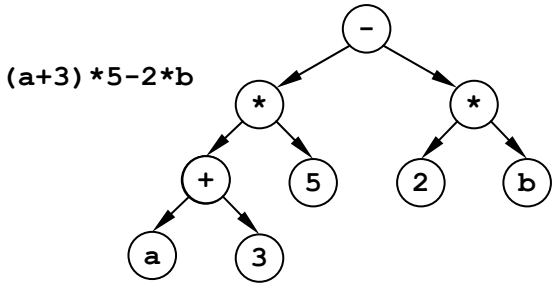
2.3 Мангустовые

…

С иерархией мы встречаемся, работая с файлами и папками: классическая файловая система имеет древовидную структуру[15]. Вход в папку – это переход на следующий (более низкий) уровень иерархии:



Алгоритм вычисления арифметического выражения тоже может быть представлен в виде дерева:



Здесь листья – это числа и переменные, тогда как корень и промежуточные узлы – знаки операций. Вычисления идут «снизу вверх», от листьев – к корню. Показанное дерево можно записать так:

```
(-(*(+(a,3),5),*(2,b)))
```

Самое интересное, что скобки здесь не обязательны; если их убрать, то выражение все равно может быть однозначно вычислено:

```
- * + a 3 5 * 2 b
```

Такая запись, которая называется *префиксной* (операция записывается *перед* данными), просматривается с конца. Как только встретится знак операции, эта операция выполняется с двумя значениями, записанными справа. В рассмотренном выражении сначала выполняется умножение:

```
- * + a 3 5 (2*b)
```

затем – сложение:

```
- * (a+3) 5 (2*b)
```

и еще одно умножение

```
- (a+3)*5 (2*b)
```

и, наконец, вычитание: `(a+3)*5-(2*b)`.

Для того, чтобы вычислять выражение не с конца, а с начала, префиксную форму «разворачивают» задом наперед, тогда получается *постфиксная* форма (операция *после* данных). Например, рассмотренное выше выражение может быть записано в виде

```
b 2 * 5 3 a + * -
```

[15] В современных файловых системах файл может «принадлежать» нескольким каталогам одновременно. При этом древовидная структура, строго говоря, нарушается.

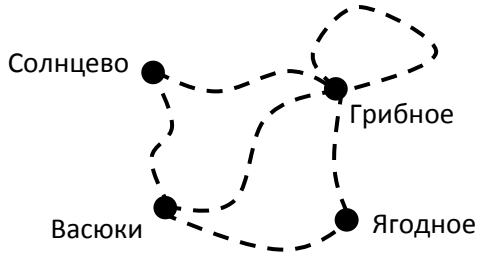
Для вычисления такого выражения скобки также не нужны, и это очень удобно для автоматических расчетов. Когда программа на языке программирования высокого уровня переводится в машинные коды, все выражения записываются в бесскобочной постфиксной форме и именно так и вычисляются.

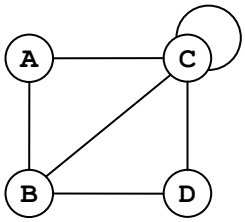
I.4.4. Графы

Подумайте, как можно структурировать такую информацию:

«От пос. Васюки три дороги идут в Солнцево, Грибное и Ягодное. Между Солнцевым и Грибным и между Грибным и Ягодным также есть дороги. Кроме того, есть дорога, которая идет из Грибного в лес и возвращается обратно в Грибное».

Можно, например, нарисовать схему дорог:





В информатике такие схемы называются *графами*.

Граф – это набор узлов (вершин) и связей между ними (рёбер).

Для хранения информации об узлах и связях показанного выше графа (см. правый рисунок) можно использовать таблицу (матрицу) такого вида,

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

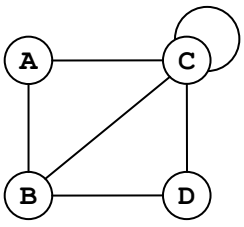
Единица на пересечении строки A и столбца B означает, что между узлами A и B есть связь. Ноль указывает на то, что связи нет. Такая таблица называется **матрицей смежности**. Она симметрична относительно главной диагонали (серые клетки в таблице).

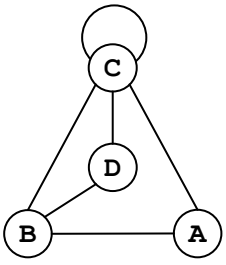
На пересечении строки C и столбца C стоит единица, которая говорит о том, что в графе есть **петля** – ребро, которое начинается и заканчивается в одной и той же вершине.

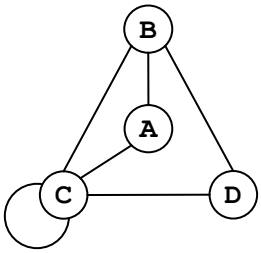
Можно поступить иначе: для каждого узла перечислить все узлы, с которыми связан данный узел. В этом случае мы получим **список смежности**. Для рассмотренного графа список смежности выглядит так:

(A (B, C), B (A, C, D), C (A, B, C, D), D (B, C))

Матрица смежности (и список смежности) не дают никакой информации о том, как именно расположены узлы друг относительно друга. Для таблицы, приведенной выше, возможны, например, такие варианты:



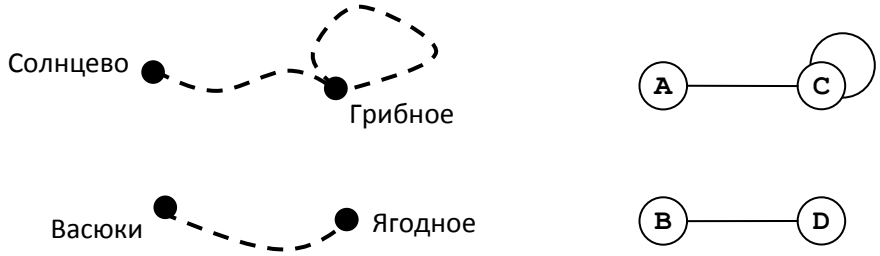




В рассмотренном примере все узлы связаны, то есть, между любой парой узлов существует **путь** – последовательность ребер, по которым можно перейти из одного узла в другой. Такой граф называется *связным*.

Связный граф – это граф, в котором все узлы связаны.

Теперь представьте себе, что дороги Васюки – Солнцево, Васюки – Грибное и Грибное – Ягодное завалило снегом (ли размыло дождем) так, что по ним не пройти и не проехать.



Эту схему тоже можно считать графом (она подходит под определение), но в таком графе есть две несвязанные части, каждая из которых – связный граф. Такие части называют **компонентами связности**.

Вспоминая материал предыдущего пункта, можно сделать вывод, что дерево – это частный случай связного графа. Но у него есть одно важное свойство – в дереве нет замкнутых путей (*циклов*). Граф в последнем примере не является деревом, потому что в нем есть циклы: ABCA, BCDB, ABCDA.

Дерево – это связный граф, в котором нет циклов.

Если в первом примере с дорогами нас интересуют еще и расстояния между поселками, каждой связи нужно сопоставить число (вес).

Такой граф называется **взвешенным**, поскольку каждое ребро имеет свой **вес**. В реальных задачах это может быть не только расстояние, но и, например, стоимость проезда или другая величина.

Как хранить информацию о таком графе? Ответ напрашивается сам собой – нужно в таблицу записывать не 1 или 0, а вес ребра. Если связи между двумя узлами нет, на бумаге можно оставить ячейку таблицы пустой, а при хранении в памяти компьютера записывать в нее условный код, например, –1. Такая таблица называется *весовой матрицей,* потому что содержит веса ребер. В данном случае она выглядит так:

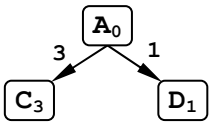
	A	B	C	D
A		12	8	
B	12		5	6
C	8	5	2	4
D		6	4	

Также как и матрица смежности, весовая матрица симметрична относительно диагонали. Две пустые ячейки говорят о том, что между узлами A и D нет связи.

Если в графе немного узлов, весовая матрица позволяет легко определить наилучший маршрут из одного узла в другой простым перебором вариантов. Рассмотрим граф, заданный весовой матрицей (числа определяют стоимость поездки между соседними пунктами):

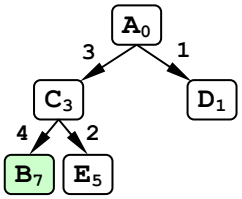
	A	B	C	D	E
A			3	1	
B			4	5	1
C	3	4			2
D	1	5			1
E		1	2	1	

Найдем наилучший путь из A в B – такой, при котором общая стоимость поездки минимальная. Сначала видим, что из пункта A напрямую в B ехать нельзя, а можно ехать только в C и D. Изобразим это на схеме:

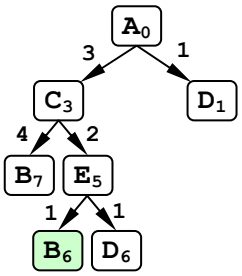


Числа около каждого ребер показывают стоимость поездки по этому участку, а индексы у названий узлов показывают общую стоимость проезда в данный узел из узла A.

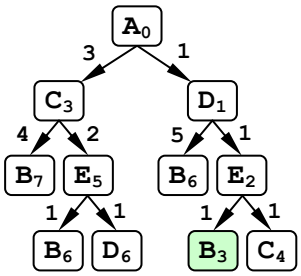
Теперь разберем варианты дальнейшего движения из узла C (узел A уже не нужно рассматривать, так как мы из него пришли).



Видим, что из C сразу можно попасть в B, стоимость проезда в этом случае равна 7. Но, возможно, это не самый лучший вариант, и нужно проверить еще путь через узел E. Действительно, оказывается, что можно сократить стоимость до 6:



Исследовать дальше маршрут, содержащий цепочку ACED, нет смысла, потому что его стоимость явно будет больше 6. Аналогично строим вторую часть схемы (вы догадались, что она представляет собой дерево!).

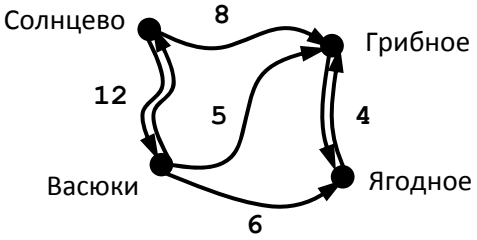


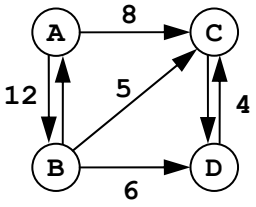
Таким образом, **оптимальный (наилучший) маршрут – ADEB**, его стоимость – 3. Маршруты ACED и ADEC, не дошедшие до узла B, далее проверять не нужно, они не улучшат результат.

Конечно, для более сложных графов метод перебора работает очень долго, поэтому используются более совершенные (но более сложные) методы, о которых мы поговорим в 11-м классе.

Наверное, вы заметили, что при изображении деревьев, которые описывают иерархию (подчинение), мы ставили стрелки от верхних уровней к нижним. Это означает, что для каждого ребра указывается направление, и двигаться можно только по стрелкам, но не наоборот. Такой граф называется *ориентированным* (или коротко *орграфом*). Он может служить, например, моделью системы дорог с односторонним движением. Матрица смежности и весовая матрица для орграфа уже не обязательно будут симметричными.

На этой схеме всего две дороги с двусторонним движением, по остальным можно ехать только в одну сторону:





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

Ребра в орграфе называют *дугами*. Дуга, в отличие от ребра, имеет начало и конец.

? *Контрольные вопросы*

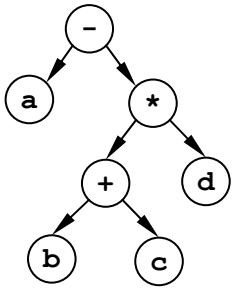
1. Что такое структура?
2. Что такое структурирование информации? Зачем оно нужно?
3. Что такое алфавитный порядок? Как поступают, если начальные символы слов совпали?
4. Расскажите, как используются оглавление, словарь и индекс для быстрого поиска нужной информации. Чем эти средства отличаются друг от друга?
5. Какими способами можно задать множество? Что такое пустое множество?
6. Приведите примеры множеств.
7. Чем отличаются множество и линейный список?
8. Что такое очередь? Приведите примеры.
9. Что такое стек? Чем он отличается от очереди? Приведите примеры.
10. Расшифруйте английские сокращения LIFO и FIFO.
11. Какая структура позволяет объединить возможности стека и очереди?
12. Что такое матрица?
13. Как можно записать табличные данные в виде списка?
14. Что такое иерархия? Приведите примеры.
15. Вспомните известные вам классификации, которые вы изучали на других предметах.
16. Как называется соответствующая структура в информатике?
17. Что такое «корень», «лист», «родитель», «сын», «предок», «потомок»?
18. В дереве 4 потомка и все они являются листьями. Нарисуйте это дерево. Сколько в нем узлов?
19. Что такое «лес»?
20. В чем разница между понятиями «ребро» и «дуга»?
21. В чем отличие понятий «дерево», «лес», «граф»?
22. Какой граф называется связным?
23. Что такое компонента связности?
24. Что такое петля? Как по матрице смежности определит, есть ли петли в графе?

25. Выберите наиболее подходящий способ структурирования информации для хранения
    а) данных по крупнейшим озерам мира;
    б) рецепта приготовления шашлыка;
    в) схемы железных дорог;
    г) схемы размещения файлов на флэш-диске.
26. Что такое орграф? Когда для представления данных используются орграфы? Приведите примеры.

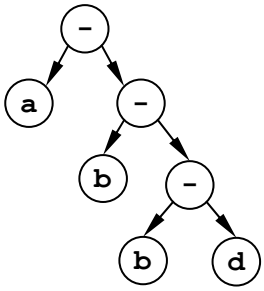
Задачи

1. Определите выражения, соответствующие каждому из деревьев, в «нормальном» виде со скобками (эту форму называют *инфиксной* – операция записывается *между* данными). Постройте для каждого из них постфиксную форму.

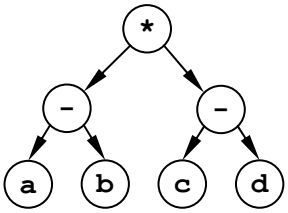
а)



б)



в)

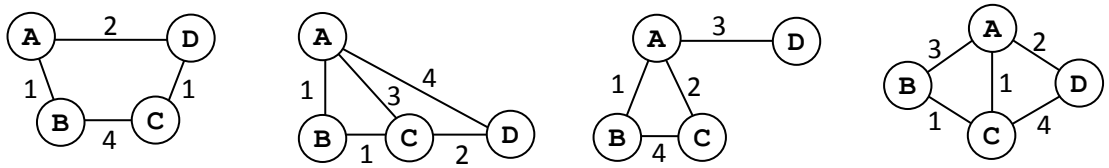


2. Постройте деревья, соответствующие следующим арифметическим выражениям. Запишите эти выражения в префиксной и постфиксной формах.
    а) **(a+b)*(c+2*d)**
    б) **(2*a-3*d)*c+2*b**
    в) **(a+b+2*c)*d**
    г) **3*a-(2*b+c)*d**
3. Вычислите выражение, записанное в постфиксной форме
    а) 12 6 + 7 3 - 1 - * 12 +
    б) 12 10 – 5 7 + * 7 – 2 *
    в) 5 6 7 8 9 + - + -
    г) 5 4 3 2 1 - - - -

    Запишите его в инфиксной и в префиксной формах. Постройте дерево, соответствующее этому выражению. Единственно ли такое дерево? В этом дереве назовите корень, листья и промежуточные узлы.
    (Ответ: 66; 34; 9; 3)
4. Нарисуйте граф, в котором 5 вершин и три компоненты связности. Постройте его матрицу смежности.
5. Структурируйте эту информацию разными способами: «Между поселками Верхние Васюки и Нижние Васюки есть проселочная дорога длиной 10 км. Село Сергеево соединяется двумя асфальтовыми шоссе с Нижними Васюками (22 км) и Верхними Васюками (16 км). В Солнечное можно доехать только из Сергеева по грунтовой дороге (5 км)». Можно ли сказать, как точно расположены эти пункты?
6. Для графа, полученного в предыдущей задаче, постройте матрицу смежности, список смежности, весовую матрицу. Является ли этот граф деревом?

7. Постройте матрицы смежности и весовые матрицы для каждого графа:

а) б) в) г)



8. Постройте графы, соответствующие каждой из матриц смежности:

а)

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

б)

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

в)

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

г)

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

9. Постройте графы, соответствующие каждой из весовых матриц:

а)

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

б)

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

в)

	A	B	C	D	E
A			2	2	6
B				2	
C	2			2	
D	2	2	2		
E	6				

г)

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

10. Стоимость перевозок между пунктами, которые для краткости обозначены буквами A, B, C, D и E, задается таблицей (весовой матрицей графа). Нужно перевезти груз из пункта A в пункт B. Для каждого из четырех вариантов определите оптимальный маршрут и полную стоимость перевозки.

а)

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

б)

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

в)

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

г)

	A	B	C	D	E
A				1	
B			4		1
C		4		4	2
D	1		4		
E		1	2		

11. Постройте орграф, соответствующий каждой из этих таблиц.

а)

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

б)

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

в)

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

г)

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