1 курс

ПЛАН – КОНСПЕКТ
проведения лекционного занятия по дисциплине
«Информатика»

Раздел 3. «Информационное моделирование.»

Тема № 3.2: «Списки, графы, деревья»

Подготовил: преподаватель
В.Н. Борисов

Рязань 2024

Лекционное занятие
по Теме № 3.2. «Списки, графы, деревья.»

Цель занятия: изучить со студентами основные сведения о структуре информации, данных, списках, графах, деревьях.

Вид занятия: классно-групповое, комбинированное (по проверке знаний, умений по пройденному материалу, по изучению и первичному закреплению нового материала).

Метод проведения занятия: доведение теоретических сведений.
Время проведения: 2 ч (90 мин.)

Основные вопросы:

1. Структура информации. Структура данных.
2. Списки, графы, деревья, таблицы.
3. Алгоритм построения дерева решений.

Литература:

1. 5 учебник раздела «Основной учебной литературы» рабочей программы изучения дисциплины: Босова, Л. Л. Информатика. 11 класс. Базовый уровень : учебник / Л.Л. Босова, А. Ю. Босова. — М. : БИНОМ. Лаборатория знаний, 2022. — 200 с. , ISBN 978-5-9963-3142-0, пар.10, 11 главы 3.

Основная часть (доведение теоретических сведений):

Первый вопрос: <u>Структура информации. Структура данных.</u>

Вычислительный процесс на ЭВМ реализуется, как известно, с помощью программ и данных. Сама программа тоже относится к данным. Поэтому можно сказать, что данные описывают любую информацию, с которой может работать ЭВМ. При этом под информацией понимаются любые факты и знания об объектах реального мира, процессах и отношениях и связях между ними. Все данные характеризуются рядом атрибутов (признаков, реквизитов), в том числе значением.

Кроме значения, к таким признакам относится понятие «тип данного». Тип данного определяется множеством значений данного и набором операций, которые можно выполнять над этими значениями в соответствии с их известными свойствами. Следовательно, тип данного определяет те операции, которые допустимы над соответствующим значением.

В языках программирования обычно используются такие распространенные типы данных, как целые, вещественные, символьные, битовые, указатели и пр.

Особенностью данного того или иного типа является простота организации (неструктурированность).

Структура данных – это совокупность элементов данных, между которыми существуют некоторые отношения, причем элементами данных могут быть как простые данные (скаляры), так и структуры данных.

Таким образом, структуру можно определить следующим образом: S = (D, R), где D - множество элементов данных, R – множество отношений между элементами данных.

Все связи одного элемента данных с другими образуют элемент отношений, ассоциированный с соответствующим элементом данных.

Графическое изображение структуры должно отражать ее элементы данных и связи (отношения между ними), поэтому структуру удобно изображать в виде графа. При этом вершины графа можно интерпретировать как элементы данных, а отношениям между элементами данных соответствуют ориентированные дуги или неориентированное ребра (рис. 1).

Таким образом описанную и представленную структуру данных называют абстрактной или логической, так как она рассматривается без учета ее представления в машинной памяти. Но любая структура данных должна быть представлена в машинной памяти. Такая структура данных называется физической структурой, структурой хранения, внутренней структурой или структурой памяти.

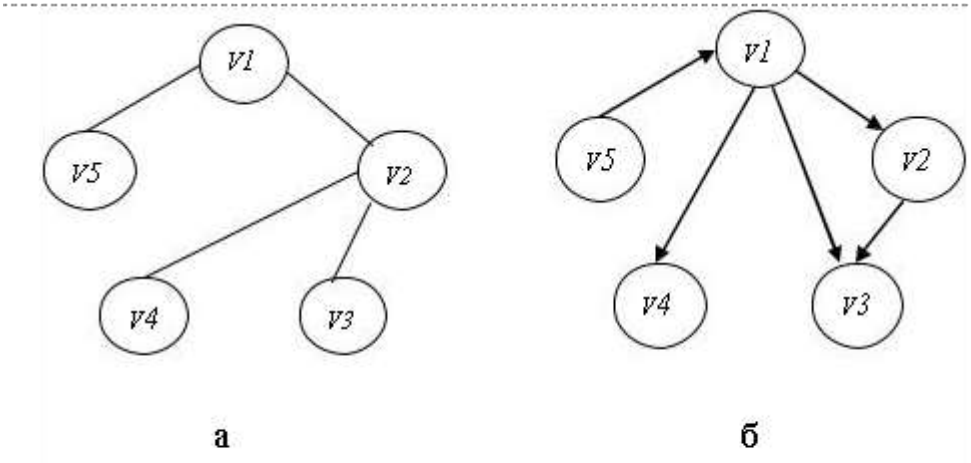

Рис 1. Неориентированный (а) и ориентированный (б) граф

Таким образом, физическая структура данных отражает способ представления данных в машинной памяти.

В общем случае между логической и соответствующей ей физической структурой существует различие, степень которого зависит от самой структуры и особенностей той физической среды, в которой она должна быть отражена.

Например, с точки зрения языков программирования двумерный массив представляет собой прямоугольную таблицу, а в памяти – это линейная последовательность ячеек, в каждой из которых хранится значение одного из элементов массива, причем элементы массива упорядочены по строкам (или столбцам).

Между логической и физической структурой должен существовать механизм, позволяющий отобразить логическую структуру в физическую.

Таким образом, каждую структуру данных можно характеризовать ее логическим (абстрактным) и физическим (конкретным) представлением, а также совокупностью операций на этих двух уровнях представления структуры (рис. 2).

Рис. 2. Отображение между логическим и физическим представлением структуры данных

Классификация структур данных.

В зависимости от отсутствия или наличия явно заданных связей между элементами данных следует различать несвязанные структуры (векторы, массивы, строки, стеки, очереди) и связные структуры (связные списки).

Важные признак структуры – ее изменчивость – изменение числа элементов и/или связей между элементами структуры. Значение элемента данных не имеется в виду, так как в этом случае это свойство было бы характерно для всех структур данных за исключением, может быть, констант и данных, хранящихся в ПЗУ. По признаку изменчивости различают статические, полустатические и динамические структуры.

Важный признак структуры данных – характер упорядоченности ее элементов. По этому признаку структуры можно делить на линейно-упорядоченные, или линейные, и нелинейные.

В зависимости от характера взаимного расположения элементов в памяти линейные структуры можно разделить на структуры с последовательным распределением их элементов в памяти (векторы, строки, массивы, стеки, очереди) и структуры с произвольным связным распределением элементов в памяти (односвязные, двусвязные, циклически связанные, ассоциативные списки). Примером нелинейных структур являются многосвязные списки, древовидные структуры и графовые структуры общего вида.

Второй вопрос: <u>Списки, графы, деревья, таблицы.</u>

Между данными, используемыми в той или иной информационной модели, всегда существуют некоторые связи, определяющие ту или иную структуру данных.

Вспомните, как мы определяли структуру данных при рассмотрении алгоритмов и программ. О каких информационных моделях тогда шла речь? С какими структурами данных вы сталкивались в программировании?

Различают линейные и нелинейные структуры данных.

В курсе информатики основной школы вы познакомились с линейным односвязным списком — последовательностью линейно связанных элементов, для которых разрешены операции добавления элемента в произвольное место списка и удаление любого элемента. Связь элементов списка осуществляется за счёт того, что каждый элемент списка содержит кроме данных адрес элемента, следующего за ним в списке. В линейном списке для каждого элемента, кроме первого, есть предыдущий элемент; для каждого элемента, кроме последнего, есть следующий элемент.

Частным случаем линейного односвязного списка является стек — последовательность, в которой включение и исключение

элементов осуществляются с одной и той же стороны этой последовательности.

Ещё одним частным случаем линейного односвязного списка является **очередь** — последовательность, у которой включение элементов производится с одной стороны последовательности, а исключение — с другой. Сторона, где происходит включение элементов, называется хвостом; сторона, где происходит исключение — головой. Понятие очереди как структуры данных очень близко к бытовому понятию «очередь» (рис. 3.2).

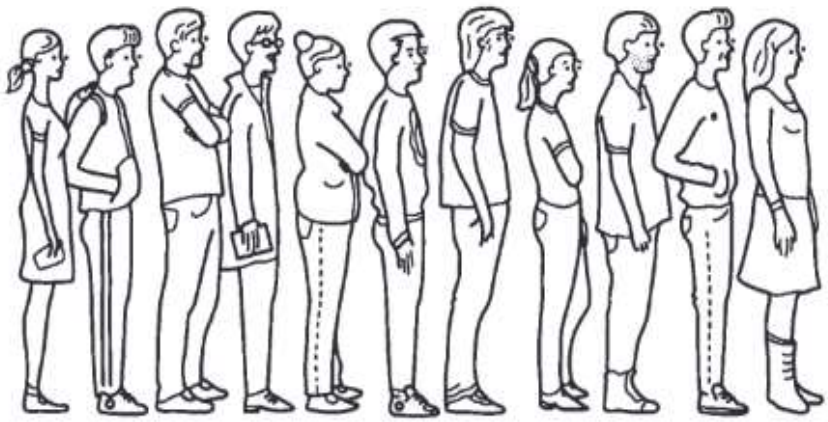

Рис. 3.2. Иллюстрация понятия «очередь»

Подумайте, какая связь между стеком и следующими объектами:

Почему стек является структурой типа LIFO (от англ. *Last In, Firts Out* — последним пришёл, первым ушёл)?

Почему очередь является структурой типа FIFO (от англ. *First In, First Out* — первым пришёл, первым ушёл)?

Примеры нелинейных структур данных вам также хорошо известны — это графы и деревья (рис. 3.3).

Граф — это множество элементов (вершин графа) вместе с набором отношений между ними.

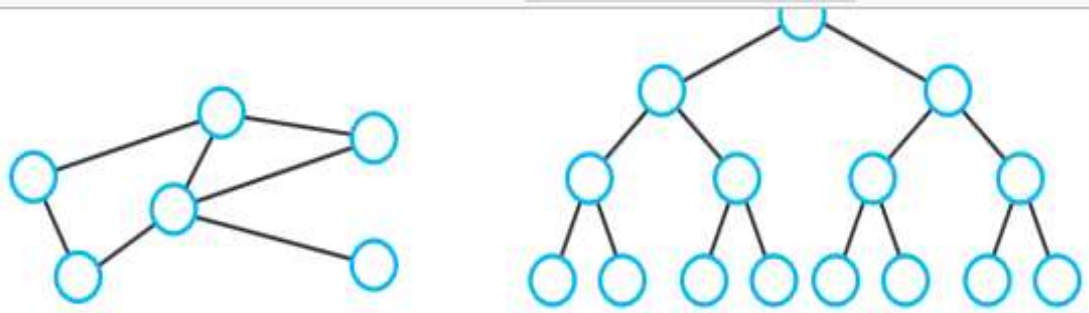

Рис. 3.3. Примеры графовых структур

Граф является многосвязной структурой, обладающей следующими свойствами:

1) на каждый элемент может быть произвольное количество ссылок;
2) каждый элемент может иметь связь с любым количеством других элементов;
3) каждая связка может иметь направление и вес.

Ненаправленная (без стрелки) линия, соединяющая вершины графа, называется **ребром**. Линия направленная (со стрелкой) называется **дугой**. При этом вершина, из которой дуга исходит, называется начальной, а вершина, куда дуга входит, — конечной. Граф называется **неориентированным**, если его вершины соединены рёбрами. Вершины **ориентированного** графа соединены дугами. Граф называется **взвешенным**, если его вершины или рёбра характеризуются некоторой дополнительной информацией — **весами** вершин или рёбер.

Графы являются основным средством для описания структур сложных объектов. С их помощью можно описать вычислительную сеть, транспортную систему, схему авиалиний и другие объекты.

Одной из разновидностей графа является дерево.

Дерево — это совокупность элементов (вершин), в которой выделен один элемент (корень), а остальные элементы разбиты на непересекающиеся множества (поддеревья). Каждое поддерево является деревом, а его корень является потомком корня дерева, т. е. все элементы связаны между собой отношением «предок — потомок». В результате образуется иерархическая структура вершин.

Частным случаем дерева является **бинарное дерево**, в котором каждая вершина может иметь не более двух потомков.

Деревья используются для представления родственных связей (генеалогическое дерево), для определения выигрышной стратегии в играх и т. д.

Ещё одной знакомой вам структурой данных являются таблицы, состоящие из строк и граф (столбцов, колонок), пересечение которых образуют ячейки. Таблицы применяют для наглядности и удобства сравнения показателей.

Оформляют таблицы в соответствии с рисунком 3.4.

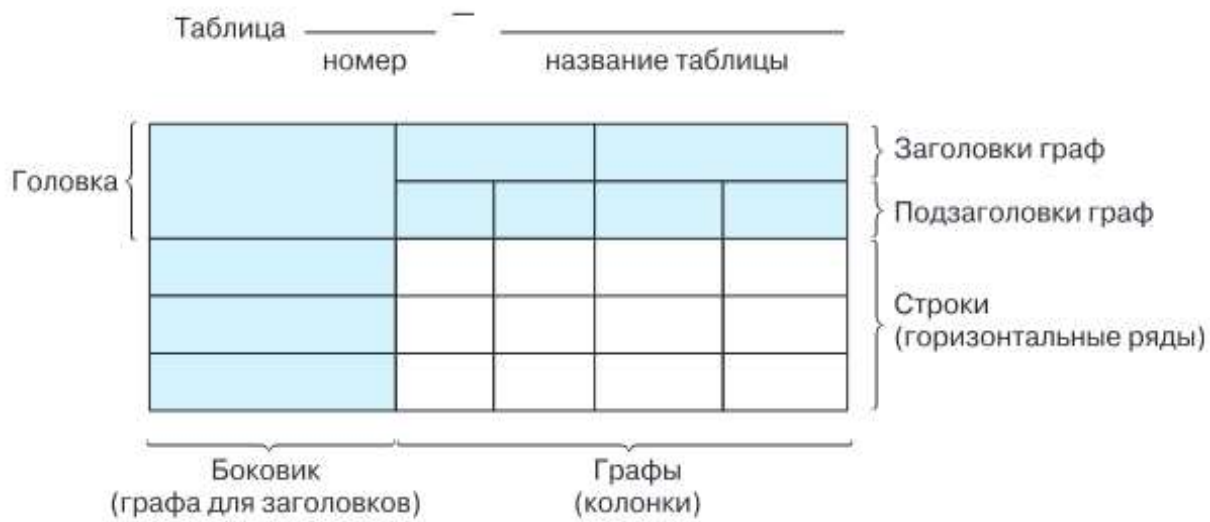

Рис. 3.4. Оформление таблицы

Название таблицы, при его наличии, должно отражать её содержание, быть точным, кратким. Название следует помещать над таблицей.

Заголовки граф и строк таблицы следует писать с прописной буквы, а подзаголовки граф — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. В конце заголовков и подзаголовков таблицы точки не ставят. Заголовки и подзаголовки граф указывают в единственном числе.

Если все показатели, приведенные в графах таблицы, выражены в одной и той же единице физической величины, то её обозначение необходимо помещать над таблицей справа. Если в графе таблицы помещены значения одной и той же физической величины, то обозначение единицы физической величины указывают в заголовке (подзаголовке) этой графы.

Эти и другие требования к оформлению таблиц содержатся в ГОСТ 2.105–95 «ЕСКД. Общие требования к оформлению текстовых документов».

В курсе информатики основной школы вы познакомились с таблицами типа:

- **«объект — свойство», содержащими информацию о свойствах отдельных объектов, принадлежащих одному классу;**

- «объект — объект», содержащими информацию о некотором одном свойстве пар объектов, принадлежащих одному или разным классам.

Таблицы, в которых отражено наличие или отсутствие связей между отдельными элементами некоторой системы, называются двоичными матрицами.

Вспомните и приведите примеры таблиц типа «объект — свойство», «объект — объект», отражающих не только количественные, но и качественные характеристики свойств (двоичные матрицы).

Табличный способ представления данных является универсальным — любую структуру данных, в том числе и представленную в форме графа, можно свести к табличной форме. Это тем более важно в связи с тем, что для компьютерной обработки табличное представление данных является предпочтительным.

Пример 1. Построим таблицу, соответствующую неориентированному графу (рис. 3.5), отражающему схему дорог между некоторыми населёнными пунктами.

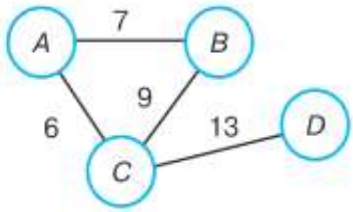

Рис. 3.5. Граф схемы дорог

Строки и столбцы таблицы будут соответствовать вершинам графа. Если две вершины являются смежными (соединены ребром), то в ячейку на пересечении соответствующих столбца и строки будем записывать вес этого ребра. В противном случае (вершины не являются смежными) в ячейку будем записывать 0. Получится таблица типа «объект — объект».

Такую таблицу называют матрицей смежности. Часто в матрицах смежности вместо нуля ставят знак минус, что обеспечивает бóльшую наглядность.

	A	*B*	*C*	*D*
A	0	7	6	0
B	7	0	9	0
C	6	9	0	13
D	0	0	13	0

	A	*B*	*C*	*D*
A	–	7	6	–
B	7	–	9	–
C	6	9	–	13
D	–	–	13	–

Матрица смежности неоринтированного графа симметрична относительно главной диагонали, идущей от левого верхнего угла кправому нижнему углу. У матрицы смежности неориентирован-ного графа такая симметрия отсутствует.

Между данными, используемыми в той или иной информационной модели, всегда существуют некоторые связи, определяющие ту или иную структуру данных. Различают линейные и нелинейные структуры данных.

Линейный односвязный список — последовательность линейно связанных элементов, для которых разрешены операции добавления элемента в произвольное место списка и удаление любого элемента. Частным случаем линейного односвязного списка является стек — последовательность, в которой включение и исключение элементов осуществляются с одной стороны этой последовательности. Ещё один частный случай линейного односвязного списка — очередь, представляющая собой последовательность, у которой включение элементов производится с одной стороны последовательности, а исключение — с другой.

Примерами нелинейных структур данных являются графы и деревья. Граф — это множество элементов (вершин графа) вместе с набором отношений между ними, называемых рёбрами (дугами) графа. Дерево — это совокупность элементов (вершин), в которой выделен один элемент (корень), а остальные элементы разбиты на непересекающиеся множества (поддеревья). Каждое поддерево является деревом, а его корень является потомком корня дерева, т. е. все элементы связаны между собой отношением «предок — потомок». Частным случаем дерева является бинарное дерево, в котором каждая вершина может иметь не более двух потомков.

Таблица — это структура данных, состоящая из строк и граф (столбцов, колонок), пересечение которых образуют ячейки. Таблицы применяют для наглядности и удобства сравнения показателей. Табличный способ представления данных является универсальным — любую структуру данных, в том числе и представленную в форме графа, можно свести к табличной форме. Это тем более важно в связи с тем, что для компьютерной обработки табличное представление данных является предпочтительным.

Пример 2. Обед в школьной столовой состоит из двух блюд и напитка. На первое можно выбрать щи или окрошку, на второе — плов или пельмени, на третье — сок или компот. Все возможные варианты представлены с помощью дерева на рисунке 3.6.

Рис. 3.6. Дерево вариантов обеда

Для того чтобы представить эту же информацию в таблице, будем двигаться по дереву от листьев к корню, описывая все возможные варианты обеда.

Для того чтобы представить эту же информацию в таблице, будем двигаться по дереву от листьев к корню, описывая все возможные варианты обеда.

Обед	Напиток	2-е блюдо	1-е блюдо
Вариант 1	Сок	Плов	Щи
Вариант 2	Компот	Плов	Щи
Вариант 3	Сок	Пельмени	Щи
Вариант 4	Компот	Пельмени	Щи
Вариант 5	Компот	Плов	Окрошка
Вариант 6	Сок	Плов	Окрошка
Вариант 7	Компот	Пельмени	Окрошка
Вариант 8	Сок	Пельмени	Окрошка

Получилась таблица типа «объект–свойства»: объектами в ней являются варианты обеда, а свойствами — составляющие его блюда. При этом число граф в полученной таблице соответствует числу уровней в дереве.

При решении класса задач, связанного с нахождением кратчайшего пути в ориентированном графе, можно:

1) от исходного графа перейти к матрице смежности;
2) по матрице смежности построить дерево решений;
3) по дереву решений выбрать подходящий вариант.

Пример 3. Найдём кратчайший путь от вершины A до вершины F в графе, приведённом на рисунке 3.7.

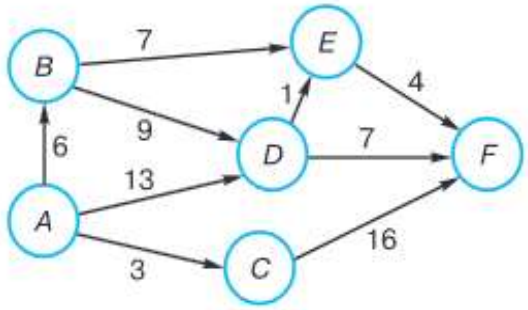

Рис. 3.7. Ориентированный граф

Составим матрицу смежности, соответствующую данному ориентированному графу:

	A	*B*	*C*	*D*	*E*	*F*
A	–	6	3	13	–	–
B	–	–	–	9	7	–
C	–	–	–	–	–	16
D	–	–	–	–	1	7
E	–	–	–	–	–	4
F	–	–	–	–	–	–

По матрице смежности построим полное дерево перебора решений — рисунок 3.8.

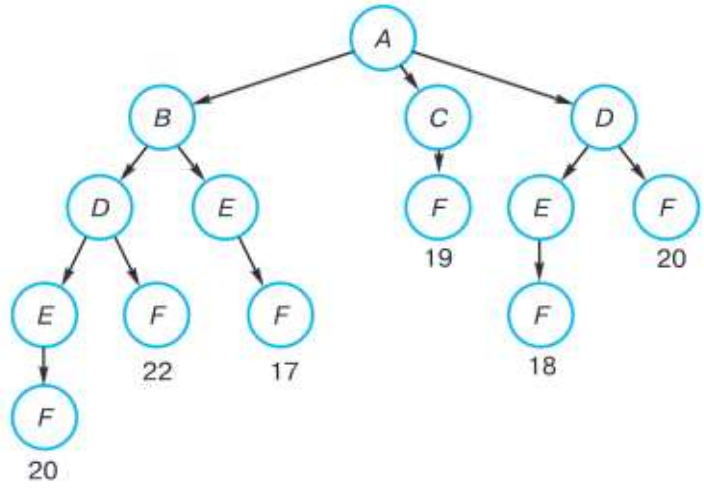

Рис. 3.8. Полное дерево перебора решений

На рисунке 3.8 видно, что кратчайший путь из вершины A в вершину F равен 17 и имеет вид $A–B–E–F$.

Третий вопрос: <u>Алгоритм построения дерева решений.</u>

Графы как информационные модели находят широкое применение во многих сферах нашей жизни. Например, с их помощью можно планировать оптимальные транспортные маршруты, кратчайшие объездные пути, расположение торговых точек и других объектов. Необходимость решения задач, связанных с поиском кратчайшего пути на графе, возникает при проектировании инженерных сетей и линий электропередач, в микроэлектронике и во многих других случаях.

Путь между вершинами A и B графа считается кратчайшим, если:

- эти вершины соединены минимальным числом ребер (в случае, если граф не является взвешенным);
- сумма весов рёбер, соединяющих эти вершины, минимальна (для взвешенного графа).

Есть множество алгоритмов определения кратчайшего пути между вершинами графа, в том числе:

1) алгоритм построения дерева решений;
2) алгоритм Дейкстры;
3) метод динамического программирования.

Алгоритм построения дерева решений, как правило, используется для нахождения кратчайшего пути в ориентированном графе. Его мы рассмотрели в предыдущем параграфе.

Заключительная часть.

1. Закончить изложение материала.
2. Ответить на возникшие вопросы.
3. Принять защиту выполненных ранее практических работ.
4. Подвести итоги занятия.
5. Выдать задание на самоподготовку (домашнее задание).

Задание на самоподготовку (домашнее задание):

1. Детально проработать, законспектировать материал занятия, размещенный в данном план-конспекте, в учебнике, указанном на с.2 текущего документа.
2. Подготовиться к опросу по пройденному материалу.