**ВСЕРОССИЙСКИЙ ЗАОЧНЫЙ ФИНАНСОВО-ЭКОНОМИЧЕСКИЙ**

**ИНСТИТУТ**

КАФЕДРА АВТОМАТИЗИРОВАННОЙ ОБРАБОТКИ

ЭКОНОМИЧЕСКОЙ ИНФОРМАЦИИ

**КУРСОВАЯ работа**

по дисциплине «**Информатика**»

на тему **«Основные структуры данных»**

а

Исполнитель: Специальность:

№личного дела:

Курс

Группа:

Руководитель:

Омск – 2008

**Оглавление:**

Реферат…………………………………………………………………………3

[ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 4](#_Toc160516788)

Введение………………………………………………………………………..4

1. [Основные понятия, используемые при изучении объекта 5](#_Toc160516789)
2. [Классификация элементов объекта 6](#_Toc160516790)
3. [Подробная характеристика элементов объекта 8](#_Toc160516791)

[Заключение 18](#_Toc160516797)

[ПРАКТИЧЕСКАЯ ЧАСТЬ 19](#_Toc160516798)

[Общая характеристика задачи 19](#_Toc160516799)

[Описание алгоритма решения задачи 21](#_Toc160516800)

[Список использованной литературы 27](#_Toc160516801)

Приложение…………………………………………………………………..28

# 

**Реферат**

Тема моей курсовой - «Основные структуры данных».

Цель данной работы рассмотреть, что такое информация, данные, структура данных, тип данных, модель данных, база данных, чем они различаются; как информация переходит в структурированные данные.

Работа вышла на 29 страниц. Первая, из которых – титульный лист. Вторая – оглавление. Третья – реферат. Четвертая – введение. С пятой по семнадцатую – теоретическая часть. Восемнадцатая страница – заключение. С девятнадцатой по двадцать шестую – практическая часть. Двадцать седьмая – список используемой литературы. С двадцать восьмой по двадцать девятую – приложения. Приложение состоит из шести таблиц. На каждую таблицу в тексте присутствует ссылка.

В основной части работы приводится, основные понятия, классификация структур данных, подробная характеристика каждой классификации.

# ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

# Введение

Веками человечество накапливало знания, навыки работы, сведения об окружающем нас мире, т.е. собирало информацию. Вна­чале информация передавалась из поколения в поколение в виде преданий и устных рассказов. Возникновение и развитие книжного дела позволило передавать и хранить информацию в более надежном письменном виде. Открытия в области электричества привели к появлению телеграфа, теле­фона, радио, телевидения — средств, позволяющих оперативно передавать и накапливать информацию. Развитие прогресса обусловило резкий рост информации, в связи, с чем вопрос о её сохранении и переработке стано­вился год от года острее. С появлением вычислительной техники значи­тельно упростились способы хранения, а главное, обработки информации. Развитие вычислительной техники на базе микропроцессоров приводит к совершенствованию компьютеров и программного обеспечения. Появля­ются программы, способные обработать большие потоки информации. С помощью таких программ создаются информационные системы. Целью любой информационной системы является обработка данных об объектах и явлениях реального мира и предоставление нужной человеку информа­ции о них.

В данной работе рассматривается, что такое информация и данные, чем они различаются; как информация переходит в структурированные данные. Рассматриваются такие понятия, как «тип данных», «структура данных», «модель данных» и «база данных». В основной части работы приводится классификация структур данных, обширная информация о физическом и логическом представлении структур данных всех классов памяти ЭВМ: простых, статических, полустатических, динамических и нелинейных; а также, информация о возможных операциях над всеми перечисленными структурами.

# ****1. Основные понятия, используемые при изучении объекта****

В информатике различают два понятия «данные» и «информация». **Данные** представляют собой информацию, находящуюся в формализованном виде и предназначенную для обработки техническими системами. Под **информацией** понимается совокупность представляющих интерес фактов, событий, явлений, которые необходимо зарегистрировать и обработать. Информация в отличие от данных – это то, что нам интересно, что можно хранить, накапливать, применять и передавать. Данные только хранятся, а не используются. Но как только данные начинают использоваться, то они преобразуются в информацию. В процессе обработки информация изменяется по структуре и форме. Признаками структуры является взаимосвязь элементов информации. Структура информации классифицируется на формальную и содержательную. Формальная структура информации ориентирована на форму представления информации, а содержательная – на содержание.

Виды форм представления информации:

1. По способу отображения:

а) символьная (знаки, цифры, буквы);

б) графическая (изображения);

в) текстовая (набор букв, цифр);

г) звуковая.

1. По месту появления:

а) внутренняя (выходная);

б) внешняя (входная)

1. По стабильности:

а) постоянная;

б) переменная

1. По стадии обработки:

а) первичная;

б) вторичная.

# ****2. Классификация элементов объекта****

Структуры данных служат материалами, из которых строятся программы. Как правило, данные имеют форму чисел, букв, текстов, символов и более сложных структур типа последовательностей, списков и деревьев.

Для точного описания абстрактных структур данных и алгоритмов программ используются такие системы формальных обозначений, называемые языками программирования, в которых смысл всякого предложения определится точно и однозначно. Под ***структурой данных*** в общем случае понимают множество элементов данных и множество связей между ними. Такое определение охватывает все возможные подходы к структуризации данных, но в каждой конкретной задаче используются те или иные его аспекты. Поэтому вводится дополнительная классификация структур данных, направления которой соответствуют различным аспектам их рассмотрения.

***Физическая структура данных*** отражает способ физического представления данных в памяти машины и называется еще *структурой хранения*, *внутренней структурой* или *структурой памяти*.

Рассмотрение структуры данных без учета её представления в машинной памяти называется ***абстрактной*** или ***логической*** ***структурой***. Вследствие их различий существуют процедуры, осуществляющие отображение логической структуры в физическую, и наоборот.

Различаются **простые** (базовые, примитивные) типы данных и **интегрированные** (структурированные, композитные, сложные). ***Простыми*** называются такие структуры данных, которые не могут быть расчленены на составные части, большие, чем биты. ***Интегрированными*** называются такие структуры данных, составными частями которых являются другие структуры данных - простые или в свою очередь интегрированные.

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

1. Весьма важный признак структуры данных - её изменчивость - изменение числа элементов или связей между элементами структуры. По признаку изменчивости различают структуры ***статические***, ***полустатические*** и ***динамические*** (рис. 1)[[1]](#footnote-1).

Базовые структуры данных, статические, полустатические и динамические характерны для оперативной памяти и часто называются ***оперативными структурами***. Файловые структуры соответствуют структурам данных для внешней памяти.

2. Второй важный признак структуры данных - характер упорядоченности её элементов. По этому признаку структуры можно делить на ***линейные и нелинейные*** ***структуры***. В зависимости от характера взаимного расположения элементов в памяти, линейные структуры можно разделить на *структуры с последовательным распределением элементов в памяти* (векторы, строки, массивы, стеки, очереди) и *структуры с произвольным связным распределением элементов в памяти* (односвязные, двусвязные списки).

В языках программирования понятие "структуры данных" тесно связано с понятием "типы данных". Информация по каждому типу однозначно определяет:

1) структуру хранения данных указанного типа, т.е. выделение памяти и представление данных в ней, и интерпретирование двоичного представления;

2) множество допустимых значений, которые может иметь тот или иной объект описываемого типа;

3) множество допустимых операций, которые применимы к объекту описываемого типа.

# 3. Подробная характеристика элементов объекта

## Простые структуры данных

***Простые структуры данных*** называют также примитивными или базовыми структурами. Эти структуры служат основой для построения более сложных структур. В языках программирования простые структуры описываются простыми (базовыми) типами. К таким типам относятся: *числовые, битовые, логические, символьные, перечисляемые, интервальные и указатели*. В дальнейшем изложении мы будем ориентироваться на язык PASCAL. Структура простых типов PASCAL приведена на **рис. 2** (через запятую указан размер памяти в байтах).[[2]](#footnote-2) Набор простых типов может несколько отличаться от указанного.

### Числовые типы

***Целые типы.*** С помощью целых чисел может быть представлено количество объектов, являющихся дискретными по своей природе (т.е. счетное число объектов).

***Вещественные типы.*** В отличии от порядковых типов (все целые, символьный, логический), значения которых всегда сопоставляются с рядом целых чисел, значение вещественных типов определяет число лишь с некоторой конечной точностью, зависящей от внутреннего формата вещественного числа.

### Битовые типы

В ряде задач может потребоваться работа с отдельными двоичными разрядами данных. Чаще всего такие задачи возникают в системном программировании, когда, например, отдельный разряд связан с состоянием отдельного аппаратного переключателя или отдельной шины передачи данных и т.п. Данные такого типа представляются в виде набора битов, упакованных в байты или слова, и не связанных друг с другом. Операции над такими данными обеспечивают доступ к выбранному биту данного.

### Логические типы

Значениями логического типа BOOLEAN может быть одна из заранее объявленных констант false (ложь) или true (истина). Данные логического типа занимают один байт памяти. При этом значении false соответствует нулевое значение байта, а значению true соответствует любое ненулевое значение байта. Над логическими типами возможны операции нулевой алгебры - НЕ (not), ИЛИ (or), И (and), исключающее ИЛИ (xor). В этих операциях операнды логического типа рассматриваются как единое целое. Результаты логического типа получаются при сравнении данных любых типов.

### Символьный тип

Значением символьного типа char являются символы из некоторого предопределенного множества. В большинстве современных персональных ЭВМ этим множеством является ASCII (American Standard Code for Information Intechange - американский стандартный код для обмена информацией). Это множество состоит из 256 разных символов, упорядоченных определенным образом, и содержит символы заглавных и строчных букв, цифр и других символов, включая специальные управляющие символы. Допускается некоторые отклонения от стандарта ASCII, в частности, при наличии соответствующей системной поддержки это множество может содержать буквы русского алфавита. Значение символьного типа char занимает в памяти 1 байт.

### Перечислимый тип

Перечислимый тип представляет собой упорядоченный тип данных, определяемый программистом, т.е. программист перечисляет все значения, которые может принимать переменная этого типа. Значения являются неповторяющимися, количество которых не может быть больше 256.

### Интервальный тип

Один из способов образования новых типов из уже существующих - ограничение допустимого диапазона значений некоторого стандартного скалярного типа или рамок описанного перечислимого типа. Определяется заданием минимального и максимального значений диапазона. При этом изменяется диапазон допустимых значений по отношению к базовому типу, но представление в памяти полностью соответствует базовому типу.

**Указатели**

Тип указателя представляет собой адрес ячейки памяти. При программировании на низком уровне - в машинных кодах, на языке Ассемблера и на языке C, который специально ориентирован на системных программистов, работа с адресами составляет значительную часть программных кодов.

## Статические структуры данных

Статические структуры относятся к разряду непримитивных структур, которые представляют собой структурированное множество базовых, структур. Например, вектор может быть представлен упорядоченным множеством чисел. Поскольку по определению статические структуры отличаются отсутствием изменчивости, память для них выделяется автоматически в момент активизации того программного блока, в котором они описаны. Выделение памяти на этапе компиляции является столь удобным свойством статических структур, что программисты используют их для представления объектов, обладающих изменчивостью.

Статические структуры в языках программирования связаны со структурированными типами. Структурированные типы в языках программирования являются теми средствами интеграции, которые позволяют строить структуры данных сколь угодно большой сложности. К таким типам относятся: массивы, записи (в некоторых языках - структуры) и множества (этот тип реализован не во всех языках).

### Векторы

Вектор (одномерный массив) - структура данных с фиксированным числом элементов одного и того же типа. Каждый элемент вектора имеет уникальный в рамках заданного вектора номер и имя.

### Массивы

**Логическая структура.**

**Массив** - такая структура данных, которая характеризуется: фиксированным набором элементов одного и того же типа; каждый элемент имеет уникальный набор значений индексов; количество индексов определяют мерность массива (два индекса - двумерный массив); обращение к элементу массива выполняется по имени массива и значениям индексов для данного элемента.

**Физическая структура** - это размещение элементов массива в памяти ЭВМ.

**Многомерные массивы** хранятся в непрерывной области памяти. Размер слота определяется базовым типом элемента массива. Количество элементов массива и размер слота определяют размер памяти для хранения массива. Принцип распределения элементов массива в памяти определен языком программирования.

**Специальные массивы.** На практике встречаются массивы, которые в силу определенных причин могут записываться в память не полностью, а частично. Это особенно актуально для массивов настолько больших размеров, что для их хранения в полном объёме памяти может быть недостаточно. К таким массивам относятся ***симметричные*** и ***разреженные*** ***массивы***.

**Симметричные массивы.** Двумерный массив, в котором количество строк равно количеству столбцов называется квадратной матрицей. Квадратная матрица, у которой элементы, расположенные симметрично относительно главной диагонали, попарно равны друг другу, называется симметричной.

**Разреженные массивы.** Разреженный массив - массив, большинство элементов которого равны между собой.

### Множества

**Логическая структура:** Множество - такая структура, которая представляет собой набор неповторяющихся данных одного и того же типа. Множество может принимать все значения базового типа. Базовый тип не должен превышать 256 возможных значений. Поэтому базовым типом множества могут быть byte, char и производные от них типы.

**Физическая структура:** Множество в памяти хранится как массив битов, в котором каждый бит указывает является ли элемент принадлежащим объявленному множеству или нет. Т.о. максимальное число элементов множества 256, а данные типа множество могут занимать не более 32-ух байт.

**Числовые множества.** Стандартный числовой тип, который может быть базовым для формирования множества - тип byte.

**Символьные множества.** Символьные множества хранятся в памяти также как и числовые множества. Разница лишь в том, что хранятся не числа, а коды ASCII символов.

**Множество из элементов перечислимого типа.** Множество, базовым типом которого есть перечислимый тип, хранится также, как множество, базовым типом которого является тип byte.

**Множество от интервального типа.** Множество, базовым типом которого есть интервальный тип, хранится также, как множество, базовым типом которого является тип byte.

### Записи

Запись - конечное упорядоченное множество полей, характеризующихся различным типом данных. Полями записи могут быть интегрированные структуры данных - векторы, массивы, другие записи.

### Таблицы

Таблица - сложная интегрированная структура. С физической точки зрения таблица представляет собой вектор, элементами которого являются записи. Логической особенностью таблиц является то, что доступ к элементам таблицы производится не по номеру (индексу), а по ключу - по значению одного из свойств объекта, описываемого записью-элементом таблицы. ***Ключ*** - это свойство, идентифицирующее данную запись во множестве однотипных записей. Ключ может включаться в состав записи и быть одним из её полей, но может и не включаться, а вычисляться по положению записи. Таблица может иметь один или несколько ключей. Иногда различают таблицы с фиксированной и с переменной длиной записи.

## Полустатические структуры данных

Полустатические структуры данных характеризуются следующими признаками: они имеют переменную длину и простые процедуры её изменения; изменение длины структуры происходит в определенных пределах, не превышая какого-то максимального значения. Если полустатическую структуру рассматривать на логическом уровне, то можно сказать, что это последовательность данных, связанная отношениями линейного списка. Физическое представление полустатических структур данных в памяти это обычно последовательность слотов в памяти, где каждый следующий элемент расположен в памяти в следующем слоте (т.е. вектор).

### Стеки

Стек - такой последовательный список с переменной длиной, включение и исключение элементов из которого выполняются только с одной стороны списка, называемого вершиной стека. Основные операции над стеком - включение нового и исключение элемента из стека. Полезными могут быть также вспомогательные операции: определение текущего числа элементов в стеке и очистка стека.

Для наглядности рассмотрим небольшой пример, демонстрирующий принцип включения элементов в стек и исключения элементов из стека. На рис. 3 (а, б ,с) изображены состояния стека:[[3]](#footnote-3)

а) пустого;

б, в, г) после последовательного включения в него элементов с именами 'A', 'B', 'C';

д, е) после последовательного удаления из стека элементов 'C' и 'B'; ж) после включения в стек элемента 'D'.

Как видно из **рис.** **3**, стек можно представить, например, в виде стопки книг (элементов), лежащей на столе. Присвоим каждой книге свое название, например A,B,C,D... Тогда в момент времени, когда на столе книг нет, про стек аналогично можно сказать, что он пуст, т.е. не содержит ни одного элемента. Если же мы начнем последовательно класть книги одну на другую, то получим стопку книг (допустим, из *n* книг), или получим стек, в котором содержится *n* элементов, причем вершиной его будет являться элемент *n+1*. Удаление элементов из стека осуществляется аналогичным образом т. е. удаляется последовательно по одному элементу, начиная с вершины, или по одной книге из стопки.

### Очереди FIFO

Очередью FIFO (First In First Out - "первым пришел - первым вышел") называется такой последовательный список с переменной длиной, в котором включение элементов выполняется только с одной стороны списка (конец), а исключение - с другой стороны (начало). Основные операции над очередью - включение, исключение, определение размера, очистка, неразрушающее чтение.

### Деки

Дек - особый вид очереди. Дек (от англ. deq - double ended queue, т.е очередь с двумя концами) - это такой последовательный список, в котором как включение, так и исключение элементов может осуществляться с любого из двух концов списка. Операции над деком: включение элемента справа, слева; исключение элемента справа, слева; определение размера; очистка.

### Строки

Строка - это линейно упорядоченная последовательность символов, принадлежащих конечному множеству символов, называемому алфавитом. Говоря о строках, обычно имеют в виду текстовые строки - строки, состоящие из символов, входящих в алфавит какого-либо выбранного языка, цифр, знаков препинания и других служебных символов. Базовыми операциями над строками являются: определение длины строки; присваивание строк; конкатенация (сцепление) строк; выделение подстроки; поиск вхождения.

## Динамические структуры данных

1. Динамические структуры по определению характеризуются отсутствием физической смежности элементов структуры в памяти непостоянством и непредсказуемостью размера (числа элементов) структуры в процессе её обработки. Для установления связи между элементами динамической структуры используются указатели, через которые устанавливаются явные связи между элементами. Такое представление данных в памяти называется ***связным***.

Достоинства связного представления данных: в возможности обеспечения значительной изменчивости структур; размер структуры ограничивается только доступным объёмом машинной памяти; при изменении логической последовательности элементов структуры требуется не перемещение данных в памяти, а только коррекция указателей. Вместе с тем связное представление не лишено и недостатков: работа с указателями требует, как правило, более высокой квалификации от программиста; на поля связок расходуется дополнительная память; доступ к элементам связной структуры может быть менее эффективным по времени.

### Связные линейные списки.

Списком называется упорядоченное множество, состоящее из переменного числа элементов, к которым применимы операции включения, исключения. Список, отражающий отношения соседства между элементами, называется ***линейным***. Если ограничения на длину списка не допускаются, то список представляется в памяти в виде связной структуры. Линейные связные списки являются простейшими динамическими структурами данных.

На **рис. 4** приведена структура односвязного списка.[[4]](#footnote-4) На нем поле INF - информационное поле, данные, NEXT - указатель на следующий элемент списка. В поле указателя последнего элемента списка находится специальный признак *nil*, свидетельствующий о конце списка.

Но, обработка односвязного списка не всегда удобна, так как отсутствует возможность продвижения в противоположную сторону. Такую возможность обеспечивает ***двухсвязный список***, каждый элемент которого содержит два указателя: на следующий и предыдущий элементы списка. Для удобства обработки списка добавляют еще один особый элемент - указатель конца списка.[[5]](#footnote-5)

Разновидностью рассмотренных видов линейных списков является кольцевой

список, который может быть организован на основе как односвязного, так и

двухсвязного списков, как показано на **рис. 6**.[[6]](#footnote-6)

Линейные списки находят широкое применение в приложениях, где непредсказуемы требования на размер памяти, необходимой для хранения данных; большое число сложных операций над данными, особенно включений и исключений.

### Нелинейные разветвленные списки

Нелинейным разветвленным списком является список, элементами которого могут быть тоже списки. Если один из указателей каждого элемента списка задает порядок обратный к порядку, устанавливаемому другим указателем, то такой двусвязный список будет линейным. Если же один из указателей задает порядок произвольного вида, не являющийся обратным по отношению к порядку, устанавливаемому другим указателем, то такой список будет нелинейным.

В обработке нелинейный список определяется как любая последовательность атомов и списков (подсписков), где в качестве атома берется любой объект, который при обработке отличается от списка тем, что он структурно неделим.

## Нелинейные структуры данных

### Графы

Граф - это сложная нелинейная многосвязная динамическая структура, отображающая свойства и связи сложного объекта. Многосвязная структура обладает следующими свойствами: на каждый элемент (узел, вершину) может быть произвольное количество ссылок; каждый элемент может иметь связь с любым количеством других элементов; каждая связка (ребро, дуга) может иметь направление и вес.

В узлах графа содержится информация об элементах объекта. Связи между узлами задаются ребрами графа. Ребра графа могут иметь направленность, показываемую стрелками, тогда они называются ***ориентированными***, ребра без стрелок - ***неориентированные***. Граф, все связи которого ориентированные, называется ***ориентированным графом*** или ***орграфом***; граф со всеми неориентированными связями - ***неориентированным графом***; граф со связями обоих типов - ***смешанным графом***. Для ориентированного графа число ребер, входящих в узел, называется *полустепенью* *захода* узла, выходящих из узла - *полустепенью исхода*. Граф без ребер является ***нуль-графом***. Если ребрам графа соответствуют некоторые значения, то граф и ребра называются *взвешенными*. ***Мультиграфом*** называется граф, имеющий параллельные (соединяющие одни и те же вершины) ребра, в противном случае граф называется *простым*. Путь в графе - это последовательность узлов, связанных ребрами. Путь от узла к самому себе называется циклом, а граф, содержащий такие пути - ***циклическим***. Два узла графа смежны, если существует путь от одного из них до другого.

### Деревья

Дерево - это граф, который характеризуется следующими свойствами:

1. Существует единственный элемент (узел или вершина), на который не ссылается никакой другой элемент - и который называется ***корнем***.
2. Начиная с корня и следуя по определенной цепочке указателей, содержащихся в элементах, можно осуществить доступ к любому элементу структуры.
3. На каждый элемент, кроме корня, имеется единственная ссылка, т.е. каждый элемент адресуется единственным указателем.

Линия связи между парой узлов дерева называется обычно ***ветвью***. Те узлы, которые не ссылаются ни на какие другие узлы дерева, называются ***листьями*** или терминальными вершинами. Узел, не являющийся листом или корнем, считается промежуточным или ***узлом ветвления*** (нетерминальной или внутренней вершиной). Деревья нужны для описания любой структуры с иерархией.

***Ориентированное дерево*** - это такой ациклический орграф (ориентированный граф), у которого одна вершина, называемая корнем, имеет полустепень захода, равную 0, а остальные - полустепени захода, равные 1. Ориентированное дерево должно иметь по крайней мере одну вершину. Изолированная вершина также представляет собой ориентированное дерево. Ориентированное дерево является ациклическим графом, все пути в нем элементарны.

# Заключение

Структуры данных – это абстрактные структуры или классы, которые используются для организации данных и предоставляют различные операции над этими данными.

В данной работе были описаны структуры данных всех классов памяти ЭВМ: простых, статических, полустатических, динамических и нелинейных, а также, информация о возможных операциях над всеми перечисленными структурами. С помощью операций можно организовывать поиск, сортировку и редактиро­вание данных. Способы анализа эффективности использования структур данных важны для подсчета времени исполнения различных операций структур данных и являются полезным инструментом при выборе той или иной структуры данных для конкретной программистской задачи.

Мы рассмотрели вопрос о важности структур данных и о том, как они влияют на эффективность алгоритмов. Выбор правильного представления данных служит ключом к удачному программированию и может в большей степени сказываться на производительности программы, чем детали используемого алгоритма. Для определения того, как структуры данных влияют на производительность программ, нужно рассмотреть, как можно строго проанализировать различные операции, выполняемые структурами данных. Вряд ли когда-нибудь появится общая теория выбора структур данных.

Стоит добавить, что совокупность структур данных и операций их обработки составляет модель данных, которая является ядром любой базы данных. Модель данных представляет собой множество структур данных, ограничений целостности и операций манипулирования данными. С помощью модели данных могут быть представлены объекты предметной области и взаимосвязи между ними. База данных основывается на использовании иерархической, сетевой или реляционной модели, на комбинации этих моделей или на некотором их подмножестве.

# ПРАКТИЧЕСКАЯ ЧАСТЬ

Вариант № 5

## Общая характеристика задачи

Рассмотрим следующую задачу.

В бухгалтерии ООО «Снежок» производится расчет отчислений по каждому сотруднику предприятия:

* в федеральный бюджет;
* фонды обязательного медицинского страхования (ФФОМС – федеральный, ТФОМС – территориальный);
* фонд социального страхования (ФСС).

Процентные ставки отчислений приведены на **рис. 6.1**. Данные для расчета отчислений в фонды по каждому сотруднику приведены на **рис. 6.2**.

* 1. Построить таблицы по приведённым ниже данным.
  2. Выполнить расчёт размеров отчислений с заработной платы по каждому сотруднику предприятия, данные расчета занести в таблицу (**рис.6.2**).
  3. Организовать межтабличные связи для автоматического формирования ведомости расчета ЕСН (*единого социального налога*) по предприятию.
  4. Сформировать и заполнить ведомость расчета ЕСН (**рис 6.3**).
  5. Результаты расчета ЕСН по каждому сотруднику за текущий месяц представить в графическом виде.

СТАВКИ ЕСН

|  |  |
| --- | --- |
| **Фонд,**  **В который производится**  **отчисление** | **Ставка, %** |
| ТФОМС | 2,00 |
| Федеральный бюджет | 20,00 |
| ФСС | 3,20 |
| ФФОМС | 0,80 |
| ИТОГО | 26,00 |

**Рис. 6.1.** Процентные ставки отчислений

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Табельный номер** | **ФИО**  **сотрудника** | **Начислено за**  **Месяц, руб.** | **Федеральный**  **Бюджет, руб.** | **ФСС, руб.** | **ФФОМС, руб.** | **ТФОМС, руб.** | **Итого, руб.** |
| 001 | Иванов И.И. | 15 600,00 |  |  |  |  |  |
| 002 | Сидоров А.А | 12 300,00 |  |  |  |  |  |
| 003 | Матвеев К.К. | 9 560,00 |  |  |  |  |  |
| 004 | Сорокин М.М. | 4 620,00 |  |  |  |  |  |
| 005 | Петров С.С. | 7 280,00 |  |  |  |  |  |

**Рис. 6.2.** Данные для расчета ЕСН за текущий месяц по каждому сотруднику

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ООО «Снежок»   |  |  | | --- | --- | | Расчетный период | | | с | по | | \_\_.\_\_.200\_ | \_\_.\_\_.200\_ |   ВЕДОМОСТЬ РАСЧЕТА ЕСН   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | **Табельный номер** | **ФИО**  **сотрудника** | **Федеральный**  **Бюджет, руб.** | **ФСС, руб.** | **ФФОМС, руб.** | **ТФОМС, руб.** | **Итого, руб.** | | 001 | Иванов И.И. |  |  |  |  |  | | 002 | Сидоров А.А |  |  |  |  |  | | 003 | Матвеев К.К. |  |  |  |  |  | | 004 | Сорокин М.М. |  |  |  |  |  | | 005 | Петров С.С. |  |  |  |  |  | | ВСЕГО ПО ВЕДОМОСТИ | | | | | | | |

**Рис. 6.3.** Ведомость расчета ЕСН

## Описание алгоритма решения задачи

1. Запустить табличный процессор MS Excel.
2. Создать книгу с именем «ООО Снежок»
3. Лист 1 переименовать в лист с названием **Ставки ЕСН.**
4. На рабочем листе **Ставки ЕСН** MS Excelсоздать таблицу процентных ставок отчислений.
5. Заполнить таблицу процентных ставок отчислений исходными данными (рис. 7.1).



**Рис. 7.1.** Расположение таблицы «**Процентные ставки отчислений**»

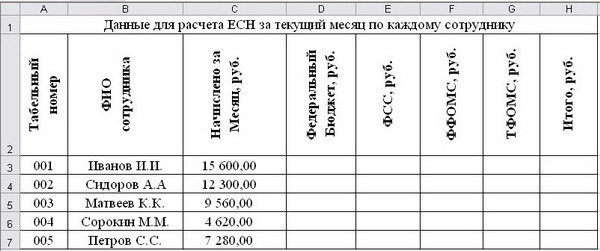
на рабочем столе **Ставки ЕСН** MS Excel

1. Разработать структуру шаблона таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**» (рис. 7.2).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Колонка**  **Электронной**  **таблицы** | **Наименование**  **(реквизит)** | **Тип данных** | **Формат данных** | |
| **длина** | **точность** |
| A | Табельный номер | 000 | 3 |  |
| B | ФИО сотрудника | текстовой | 50 |  |
| C | Начислено за месяц, руб. | денежный | 20 | 2 |
| D | Федеральный  Бюджет, руб. | денежный | 20 | 2 |
| E | ФСС, руб. | денежный | 20 | 2 |
| F | ФФОМС, руб. | денежный | 20 | 2 |
| G | ТФОМС, руб. | денежный | 20 | 2 |
| H | Итого, руб. | денежный | 20 | 2 |

**рис. 7.2.** Структура шаблона таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**»

1. Лист 2 переименовать в лист с названием **Сотрудники.**
2. На рабочем листе **Сотрудники** MS Excel создать таблицу, в которой будут содержаться данные для расчета ЕСН за текущий месяц по каждому сотруднику.
3. Заполнить таблицу «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**» исходными данными (рис. 7.3).



**Рис. 7.3.** Расположение таблицы

«**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**»

на рабочем столе **Сотрудники** MS Excel

1. Заполнить графу **Федеральный бюджет, руб.** таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**», находящейся на листе **Сотрудники** следующим образом:

Занести в ячейку D3 формулу: =ПРПЛТ(C3;1;1;-'Ставки ЕСН'!B4)

Занести в ячейку D4 формулу: =ПРПЛТ(C4;1;1;-'Ставки ЕСН'!B4)

Занести в ячейку D5 формулу: =ПРПЛТ(C5;1;1;-'Ставки ЕСН'!B4)

Занести в ячейку D6 формулу: =ПРПЛТ(C6;1;1;-'Ставки ЕСН'!B4)

Занести в ячейку D7 формулу: =ПРПЛТ(C7;1;1;-'Ставки ЕСН'!B4)

1. Заполнить графу **ФСС, руб.** таблицы «**Данные для расчета ЕСН за**

**текущий месяц по каждому сотруднику**», находящейся на листе **Сотрудники** следующим образом:

Занести в ячейку E3 формулу: =ПРПЛТ(C3;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку E4 формулу: =ПРПЛТ(C4;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку E5 формулу: =ПРПЛТ(C5;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку E6 формулу: =ПРПЛТ(C6;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку E7 формулу: =ПРПЛТ(C7;1;1;-'Ставки ЕСН'!B5)

1. Заполнить графу **ФФОМС, руб.** таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**», находящейся на листе **Сотрудники** следующим образом:

Занести в ячейку F3 формулу: =ПРПЛТ(C3;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку F4 формулу: =ПРПЛТ(C4;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку F5 формулу: =ПРПЛТ(C5;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку F6 формулу: =ПРПЛТ(C6;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку F7 формулу: =ПРПЛТ(C7;1;1;-'Ставки ЕСН'!B5)

1. Заполнить графу **ТФОМС, руб.** таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**», находящейся на листе **Сотрудники** следующим образом:

Занести в ячейку G3 формулу: =ПРПЛТ(C3;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку G4 формулу: =ПРПЛТ(C4;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку G5 формулу: =ПРПЛТ(C5;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку G6 формулу: =ПРПЛТ(C6;1;1;-'Ставки ЕСН'!B5)

Занести в ячейку G7 формулу: =ПРПЛТ(C7;1;1;-'Ставки ЕСН'!B5)

1. Заполнить графу **Итого, руб.** таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**», находящейся на листе **Сотрудники** следующим образом:

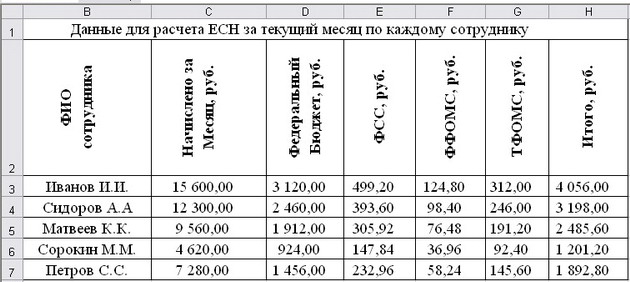
Занести в ячейку H3 формулу: =СУММ(D3:G3)

Размножить введённую в ячейку H3 формулу для остальных ячеек (с H4 по H7) данной графы.

Таким образом, будет выполнен *цикл*, управляющим параметром которого является номер строки.

1. Таблица «**Данные для расчета ЕСН за текущий месяц по каждому**

**сотруднику**» на рабочем столе **Сотрудники** MS Excel заполнена (рис. 7.4).



**Рис. 7.4.** Расположение таблицы

«**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**»

на рабочем столе **Сотрудники** MS Excel

1. Лист 3 переименовать в лист с названием «**Ведомость**».
2. На рабочем листе **Ведомость** MS Excel создать таблицу «**Ведомость расчёта ЕСН**».

Путём создания межтабличных связей заполнить созданную ведомость расчёта ЕСН полученными данными из таблицы «**Данные для расчета ЕСН за текущий месяц по каждому сотруднику**» (рис. 7.5).

Заполнить графу **Федеральный бюджет, руб.** таблицы «**Ведомость расчёта ЕСН**», находящейся на листе **Ведомость** следующим образом:

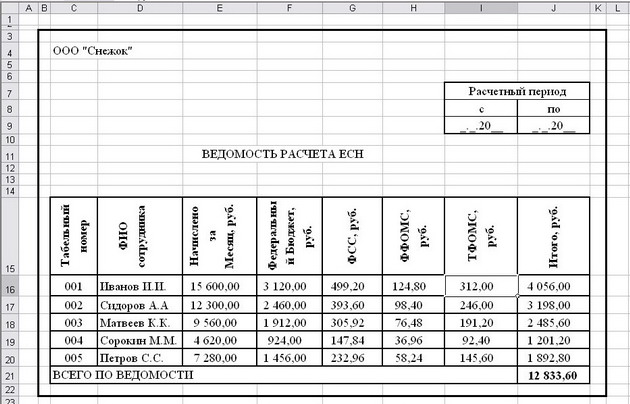
Занести в ячейку F16 формулу: =Сотрудники!D3

Размножить введённую в ячейку F16 формулу для остальных ячеек (с F17 по F20) данной графы.

1. Заполнить графу **ФСС, руб.** таблицы «**Ведомость расчёта ЕСН**», находящейся на листе **Ведомость** следующим образом:

Занести в ячейку G16 формулу: =Сотрудники!E3

Размножить введённую в ячейку G16 формулу для остальных ячеек (с G17 по G20) данной графы.



**Рис. 7.5.** Расположение таблицы **«Ведомость расчёта ЕСН»**

на рабочем столе **Ведомость** MS Excel

1. Заполнить графу **ФФОМС, руб.** таблицы «**Ведомость расчёта ЕСН**», находящейся на листе **Ведомость** следующим образом:

Занести в ячейку H16 формулу: =Сотрудники!F3

Размножить введённую в ячейку H16 формулу для остальных ячеек (с H17 по H20) данной графы.

1. Заполнить графу **ТФОМС, руб.** таблицы «**Ведомость расчёта ЕСН**», находящейся на листе **Ведомость** следующим образом:

Занести в ячейку I16 формулу: =Сотрудники!G3

Размножить введённую в ячейку I16 формулу для остальных ячеек (с G17 по G20) данной графы.

1. Заполнить графу **Итого, руб.** таблицы «**Ведомость расчёта ЕСН**», находящейся на листе **Ведомость** следующим образом:

Занести в ячейку J16 формулу: =Сотрудники!H3

Размножить введённую в ячейку J16 формулу для остальных ячеек

(с J17 по J20) данной графы.

Занести в ячейку J21 формулу: =СУММ(J16:J20)

1. В документ **ООО Снежок** MS Excel добавить новый лист.
2. Полученный новый лист переименовать в лист с названием **График**.
3. Результаты расчета ЕСН по каждому сотруднику представить в графическом виде на рабочем листе **График** MS Excel (рис. 7.6).



**Рис. 7.6.** Расположение графика результата расчета ЕСН по каждому сотруднику

на рабочем столе **График** MS Excel

# Список использованной литературы

1. Башлы П.Н. Информатика /П.Н. Башлы. – Ростов н/Д: Феникс, 2006.- 249с.
2. Вирт Н. Алгоритмы и структуры данных./Пер. с англ. - М.: Мир, 1999.–360с.
3. Евсюков В.В. Экономическая информатика: Учеб. пособие – Тула: Издательство «Гриф и К», 2003. – 371с.: ил. базы данных.
4. Информатика: Практикум по технологии работы на комп. / Под ред. Н.В. Макаровой. - М.: Финансы и статистика, 2000. - 384с.
5. Модели и структуры данных В.Д. Далека, А.С. Деревянко, О.Г. Кравец, Л.Е. Тимановская. Учебное пособие. Харьков: ХГПУ, 2000. - 241с.
6. Меняев М.Ф. Информационные технологии управления: Учеб. пособие. В 3 кн.: Книга 1: Информатика. – М.: Омега – Л, 2003. - 464с.
7. Меняев М.Ф. Информационные технологии управления: Учеб. пособие. В 3 кн.: Книга 2: Информ. Ресурсы. – М.: Омега – Л, 2003. - 432с.
8. Практикум по эконом. информатике: Учеб. пособие: В 3-х ч. – Ч.I / Под ред. Е.Л. Шуремова, Н.А. Тимаковой, Е.А. Мамонтовой – М.: Финансы и статистика; Перспектива, 2002. – 300с.
9. Советов Б. Я.. Информационные технологии. Учебник для студентов вузов. 2006. - 263 с.
10. Степанов А.Н. Информатика: Учебник для вузов. 5-е изд. – Спб.: Питер, 2007. – 765с.: ил. база данных.
11. Уоллес Вонг. Основы программирования для «чайников». ДИАЛЕКТИКА. Москва – Санкт-Петербург – Киев. 2001. – 335 с.
12. Интернет.

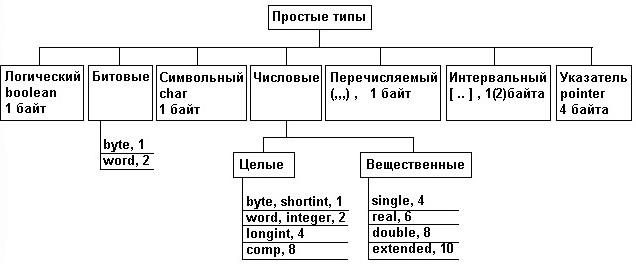
**Приложение №1**

**Рис. 1.** Классификация структур данных



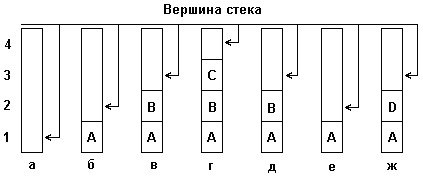
**Приложение №2**

**Рис. 2.** Структура простых типов PASCAL



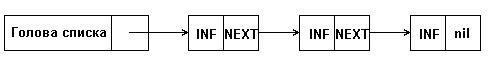
**Приложение №3**

**Рис. 3.** Включение и исключение элементов из стека



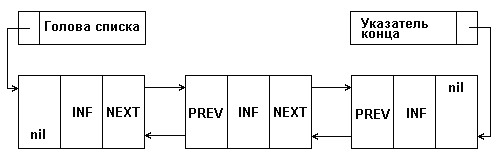
**Приложение №4**

**Рис. 4** Структура односвязного списка



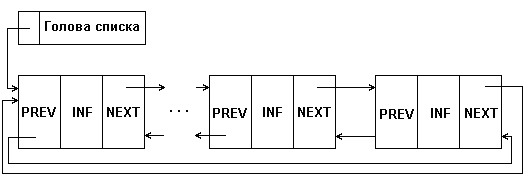
**Приложение №5**

**Рис. 5** Структура двухсвязного списка



**Приложение №6**

**Рис. 6** Структура кольцевого двухсвязного списка



1. См. Приложение 1 (стр. 28) [↑](#footnote-ref-1)
2. См. Приложение 2 (стр. 28) [↑](#footnote-ref-2)
3. См. Приложение 3 (стр. 29) [↑](#footnote-ref-3)
4. См. Приложение 4 (стр. 29) [↑](#footnote-ref-4)
5. См. Приложение 5 (стр. 29) [↑](#footnote-ref-5)
6. См. Приложение 6 (стр. 29) [↑](#footnote-ref-6)