МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ
ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

М Е Т О Д И Ч Е С К И Е У К А З А Н И Я

к лабораторным работам

по программированию на языке Паскаль

ч а с т ь 4

(для студентов специальности 6.050102
“Компьютерная инженерия”
дневной формы обучения)

Рассмотрено
на заседании кафедры КИ
Протокол № 1
от 31 августа 2010 г.

Утверждено
на заседании учебно-
издательского срвета ДонНТУ
Протокол № 4
от 07 октября 2010 г.

Донецк ДонНТУ 2010

УДК 681.3(07)

Методические указания к лабораторным работам по программированию на языке Паскаль, часть 4 (для студентов специальности 6.050102 “Компьютерная инженерия” дневной формы обучения). Составители: В.И.Назаренко, К.Б.Юсупова, О.Ю.Чередникова. – Донецк: ДонНТУ, 2010. – 73 с.

Приведены методические указания к трем лабораторным работам, которые входят в цикл лабораторных работ по программированию на языке Паскаль. В лабораторной работе № 7 рассматриваются методы разработки алгоритмов и программ для задач обработки записей, лабораторная работа № 8 посвящена обработке списков, в работе № 9 изложена методика разработки многомодульных программ. Для данного комплекса лабораторных работ приведено 100 вариантов задач.

Составители: доц. Назаренко В.И.,
асс. Юсупова К.Б.,
асс. Чередникова О.Ю.

Ответственный
за выпуск проф. Святный В.А.

Рецензент доц. Федяев О.И.

Л а б о р а т о р н а я р а б о т а № 7

О Б Р А Б О Т К А З А П И С Е Й

М е т о д и ч е с к и е у к а з а н и я

В широко распространенных административно-учетных системах (АУС), которые называют также автоматизированными системами управления (АСУ), объектом машинной обработки является документ (группа документов).

АСУ с точки зрения программной реализации имеет ряд характерных признаков.

1. Использование большого количества файлов для представления входной, корректирующей, обрабатываемой и результирующей информации.

2. Превалирование операций ввода-вывода в общем времени решения задачи по сравнению с вычислительными операциями (в отличие от научно-технических задач).

3. Объединение в одном комплексе большого количества функциональных режимов работы системы (ввод заданной категории данных, проверка достоверности данных, коррекция данных в файлах, разные виды обработки данных и т.д.). При этом, как правило, разные режимы работы системы реализуются отдельными процедурами или программными модулями. Режим работы системы выбирает пользователь в процессе диалога с ЭВМ.

Каждый документ структурно разделяется на три части: "шапка", столбцы и строки. Столбцы документа, которые отображают его содержательную часть, называют также реквизитами.

В документах, обрабатываемых на ЭВМ, часто встречаются повторяемые текстовые реквизиты (например, наименование кафедры, название учебной дисциплины в АСУ вуза и т.п.). Для уменьшения затрат памяти при сохранении и обработке документов такие реквизиты заменяют кодами (шифрами), а соответствие между шифрами и текстами, которые их определяют, отображают в отдельных файлах, которые называют кодификаторами соответствующей информации. Для исходящих документов, в случае необходимости, по заданному шифру автоматически определяется из кодификатора его текстовый эквивалент, который и выдается на печать.

В программе на Паскале каждая строка документа - это отдельная запись; документ в целом (группа однородных документов) при обработке на ЭВМ - это массив записей, а при продолжительном хранении - файл, компонентами которого являются соответствующие записи.

В лабораторной работе № 7 в соответствии с условием задачи необходимо выполнить:

- ввод записей из текстового файла и формирование типизированного файла, который рассматривается в дальнейшем как архив АСУ;
- группирование архива по заданному признаку;
- вывод на экран и печать на принтере архива АСУ;
- дополнение архива новыми записями;
- удаление заданной записи из архива;
- изменение реквизитов заданной записи;
- формирование выходных документов.

Исходный текстовый файл в учебных задачах, как правило, должен содержать не менее 25 записей. Дополнение архива должно выполняться с использованием отдельного текстового файла, который содержит 3-5 записей.

Использование кодификатора при выполнении лабораторной работы № 7 необязательно.

Отчет по лабораторной работе № 7

В состав отчета по лабораторной работе №7 должны входить:

- титульный лист;
- условие задачи;
- краткое описание разработанной программы;
- текст программы;
- результаты работы программы.

Текст программы должен печататься средствами компилятора Турбо Паскаля. Если по какой-либо причине для этого используется система Word, то в этом случае должен использоваться шрифт Courier New (в этом шрифте все символы имеют одинаковую ширину, что обеспечивает печать программы в структурированном виде). Для пп. 2 и 3 рекомендуется шрифт Times New Roman.

Результаты работы программы, выводимые на принтер или в текстовый файл на диске, должны содержать следующую информацию:

- архив записей в его исходном состоянии;
- то же самое после дополнения архива, после его сортировки, удаления записи, изменения реквизитов записи;
- выходные документы, предусмотренные условием задачи.

Если в программе используется кодификатор, то его содержание также нужно вывести на принтер или на диск.

Примечание. Описание программы является обязательным. При его отсутствии отчет не принимается.

Ввод записей из текстового файла

Предположим, что личная карточка студента содержит такие реквизиты: фамилия и инициалы, дата рождения, национальность, год окончания школы, пол. Этой карточке поставим в соответствие следующее описание записи:

```
Type  Date = record
        Day : 1..31;              { число }
        Month : 1..12;            { месяц }
        Year : 1970..2000;        { год }
      end;
      StudType = record
        Fam : string[25];         { фамилия и инициалы }
        BirthDay : Date;          { дата рождения }
        Nac : string[15];         { национальность }
        SchoolYear : 1990..2010; { год окончания школы }
        Sex : char                { пол }
      end;
Var Student : StudType;
    ch : char;
    FileInput : text;
```

Примечание. Размещение описания типа `Date` отдельно, а не внутри записи `StudType`, имеет следующие преимущества:

- в программе можно объявить переменные типа `Date` (например, текущую дату);
- улучшается читабельность программы, в особенности при использовании записей с большим количеством реквизитов.

Рассмотрим два варианта ввода записи из текстового файла `FileInput`. В обоих вариантах предполагается, что реквизиты записи размещаются в одной строке текстового файла. Значения реквизитов разделяются между собою одним или несколькими пробелами.

Вариант 1.

Количество позиций, которые отводятся в строке файла каждому текстовому реквизиту, равно или больше значения, указанного в описании соответствующей строки. В данном случае

это означает, что для переменной Fam должно быть отведено не менее 25 позиций, а для переменной Nac – не менее 15 позиций.

Пример:

```
Петренко А.С.            15 5 1980 украинец      1977 м
```

Тогда программа ввода одной записи может иметь вид:

```
With Student,BirthDay do
  Begin
    Read(FileInput,Fam);
    Read(FileInput,Day,Month,Year);
    Repeat
      Read(FileInput,ch);
    Until  ch<>' ';
    Read(FileInput,Nac);
    Insert(ch,Nac,1);
    Read(FileInput,SchoolYear);
    Repeat
       Read(FileInput,ch);
    Until  ch<>' ';
    Sex:=ch;
    Readln(FileInput);
End;
```

Комментарии к программе:

1. В программе предполагается, что переменная `Student.Fam` записана с первой позиции строки текстового файла. Если это не так, то пробелы, поставленные перед реквизитом "фамилия и инициалы", будут включены в состав строки `Student.Fam`, что в дальнейшем отразится на корректности работы программы.

2. В одной строке файла одновременно размещены разнотипные элементы: числа и текст. Эти элементы отделены между собою одним или несколькими пробелами. Процедура `Read` при вводе числовой переменной автоматически выделяет ее из строки текстового файла (предполагается, что числа в текстовом файле разделены пробелами). При вводе переменной типа **`string`** пробел воспринимается как обычный символ. Поэтому в программе для индикации начала текстового реквизита с помощью цикла **`Repeat`** перебираются позиции строки файла до тех пор, пока в символьную переменную `ch` не будет прочитан символ, отличный от пробела, после чего читается строковая переменная `Nac`. Поскольку первый символ реквизита "национальность" был прочитан в переменную `ch`, то его добавление к переменной `Nac` осуществляется процедурой `Insert`.

3. Процедура `Readln(FileInput)` осуществляет переход на новую строку текстового файла после ввода обработанной в данный момент записи.

4. При отсутствии оператора **With** в программе нужно было бы использовать полные составные имена элементов записи, в частности

Read(FileInput,Student.BirthDay.Day, Student.BirthDay.Day,Student.BirthDay.Day)

и Read(FileInput,Student.Nac).

Предположим, что при формировании текстового файла не соблюдено соответствие между количеством позиций в строке файла и объявлением переменной типа **`string`**. Например, для переменной `Fam` отведено меньше 25 позиций:

```
Петренко А.С.         15 5 1980  украинец    1977  м
```

Тогда значение введенной переменной `Fam` будет иметь следующий вид:

Петренко А.С. 15 5',

при этом для переменной BirthDay.Day будет введено значение 1980 (на самом деле эта переменная имеет тип byte и получит значение 188 - последний байт машинного изображения числа 1980 в формате word (07BC)), а при попытке присвоить переменной BirthDay.Month значение "украинец" произойдет прерывание работы программы с выводом сообщения "Runtime Error 106: Invalid numeric format".

С точки зрения пользователя, вариант 1 имеет серьезный недостаток: при подготовке текстового файла необходимо постоянно следить, чтобы значеие переменной Student.Fam занимало в строке файла не менее 25 позиций, а переменной Student.Nac - не менее 15 позиций, независимо от текущей длины этих переменных. Для повышения сервиса пользователя более удобно было бы оставить лишь одно ограничение: элементы строки файла должны разделяться одним или несколькими пробелами. Пример реализации ввода записей Student из текстового файла с учетом указанного ограничения приведен ниже как вариант 2.

Вариант 2

```
Var   k,k1,k2 : byte;
      Code : integer;
      Cond : boolean;
      Sa,Sb : string;
.......................
  With Student,BirthDay do
    Begin
      Readln(FileInput,Sa);
      k:=0; k2:=0;
      Cond:=true;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1;
                  Cond:=false;
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Fam:=Sb;
                2 : Begin
                      Fam:=Fam+' '+Sb;
                      While length(Fam)<25 do
                        Fam:=Fam+' ';
                    End;
                3 : Val(Sb,Day,Code);
                4 : Val(Sb,Month,Code);
                5 : Val(Sb,Year,Code);
                6 : Begin
                      Nac:=Sb;
                      While length(Nac)<15 do
```

```
                    Nac:=Nac+' ';
                End;
              7 : Val(S2,ScoolYear,Code);
              8 : Sex:=Sb[1];
            end;
          End;
        End;
    End;
```

В варианте 2 очередная строка текстового файла вводится в строковую переменную `Sa`. После этого с помощью функций `Space` и `NotSpace`, рассмотренных в лабораторной работе № 6, выделяются отдельные слова строки `Sa`. Формирование компонентов записи `Student` происходит в операторе `Case` в соответствии с порядковым номером `k` выделенного слова. При этом учтено, что между фамилией и инициалами студента в строке входного файла есть по крайней мере один пробел.

Остановимся отдельно на одной особенности приведенного выше фрагмента программы: в нем строчные переменные `Fam` и `Nac` дополняются оператором `While` до объявленной длины.

Представим, что в программе ведется поиск в архиве фамилии Fam, значение которой задано с клавиатуры. Ввод этой фамилии может быть выполнен процедурой Readln(Fam) по объявленной длине. Если за счет пробелов фамилия в архиве и фамилия, введенная из клавиатуры, имеют разную длину (при сравнении более короткая строка дополняется символами # 0, а не пробелами), то программа будет считать их неодинаковыми, вследствие чего ее работа будет некорректной. Чтобы гарантировать правильную работу операции сравнения, все строки, введенные из файла или из клавиатуры, должны быть дополнены пробелами до объявленной длины.

Имеется еще одна причина того, что введенные значения строковых переменных должны быть дополнены пробелами до их объявленной длины. Если это не сделать, то длины переменных Fam и Nac, введенные в цикле из разных строк текстового файла, могут быть различными. Тогда при печати таблицы, отображающей содержание архива записей, соответствующие графы будут иметь различную ширину, что по крайней мере неэстетично.

Следует обратить внимание на одну ошибку, которую часто допускают при формировании текстового файла.

Если после последней строки текстового файла есть одна или несколько строк, заполненных лишь пробелами, то в массив записей будет введено соответствующее количество дополнительных записей. В варианте 1 эти записи будут заполнены нулевыми значениями, в варианте 2 они будут копировать последнюю "нормальную" запись. Чтобы исключить пустые строки в конце текстового файла, рекомендуется:

1. В оболочке редактора Turbo Pascal или Norton Commander (клавиши F4 - Edit) нажать клавиши Ctrl + PgDn.

2. Если курсор станет ниже последней значащей строки текстового файла, установить его после последнего символа последней строки файла и выполнить продолжительное нажатие клавиши Del (3-5 с).

3. Проверить отсутствие пустых строк последовательным нажатием клавиши Ctrl+PgUp, а потом Ctrl+PgDn.

Примечание. Автоматическое удаление пустых строк при проверке корректности исходных текстовых файлов предусмотрена в лабораторной работе № 9.

Пример выполнения задания

Типичный образец небольшой АСУ рассматривается в приведенном ниже примере выполнения задания по лабораторной работе № 7.

Условие задачи.

Имеются следующие сведения о деятельности цехов некоторого предприятия: номер цеха (2 цифры), код изделия (6 цифр), наименование изделия, единица измерения, цена изделия, план выпуска и фактический выпуск изделия по полугодиям. Необходимо создать архив сведений о выпускаемых изделиях и сгруппировать документы архива по возрастанию кода изделия. В программе должны быть предусмотрены следующие виды коррекции архива:

- добавление документов;
- удаление документа;
- изменение в заданном документе последних четырех реквизитов (единица измерения, цена, плановый и фактический выпуски).

В обрабатывающей части программы нужно предусмотреть:

- печать архива;
- определение для заданного цеха процента выполнения плана в стоимостном выражении по полугодиям и в целом за год;
- формирование списка изделий, по которым не выполнен годовой план производства.

Решение задачи приведено в представленной ниже программе Labor7.

Строке документа, приведенного в условии задачи, в программе `Labor7` отвечает запись типа `ProductType`. Некоторых пояснений требует представление в этой записи двух реквизитов документа: номера цеха `NumberShop` и кода изделия `Kod`.

В условии указано, что номер цеха изображается двумя цифрами. С этой точки зрения данный параметр можно рассматривать как число. Однако это число имеет специфическую особенность: оно не может быть использовано для арифметических операций. Бессмысленно, например, говорить о сумме двух номеров цехов или об их произведении. Номер цеха при обработке в программе может использоваться лишь в операциях сравнения (например, при группировке записей по возрастанию номера цеха). Поскольку арифметические операции для номера цеха `NumberShop` недопустимы, переменную `NumberShop` можно рассматривать как строку типа **`string`**`[2],` элементами которой являются цифры. В программе для переменной `NumberShop` избран тип `byte` в основном с точки зрения экономии памяти (тип `byte` требует один байт памяти вместо трех байтов для типа **`string`**`[2]`).

Из аналогичных соображений для переменной `Kod` принят тип `longint` (максимальное значение переменной `Kod` по условию задачи равно 999999; максимальное значение для типа `byte` равняется 255, для типа `integer` - 32767, типа `word` - 65535, типа `longint` - 2 147 483 647). Если бы по условию задачи для кода изделия использовался литерно-цифровой шифр или цифровой код занимал более 9 цифр, то переменная `Kod` должна была бы иметь вид строки.

Соответствие между кодом изделия и его наименованием отображено в кодификаторе изделий, запись которого имеет тип `KodifType`. Наличие такого кодификатора позволяет при вводе исходных данных не указывать в документе наименования изделия, что ускоряет подготовку данных и уменьшает количество возможных ошибок. Использование кодификатора, как уже отмечалось выше, позволяет также экономить память при сохранении и обработке информации на ЭВМ.

Количество сохраненных и обрабатываемых документов в АСУ - переменная величина. В процессе эксплуатации АСУ типичными операциями являются добавление новых документов к имеющемуся архиву, удаление части документов, изменение содержания некоторых из них. Кодификаторы, в отличие от основных файлов АСУ, изменяются очень редко; в связи с этим в программе не предусмотрены операции по корректированию кодификатора. Максимальный размер кодификатора определен константой `MaxKodif`, его реальный размер - переменной `nk`. Для архивного массива ту же роль играют константа `Nmax` и переменная `np`.

В программе `Labor7` используются 5 файлов. В текстовом файле `FileInput` размещаются входные документы (сведения об изделиях), в файле `FileKodif` - кодификатор

изделий. Текстовый файл FileAdd предназначен для документов, добавляемых в архив; строки этого файла имеют такой же вид, как и строки файла FileInput. Файл FileRes используется для вывода результатов, если в качестве выводного устройства пользователь указал магнитный диск. Файл FileOut является типизированным, именно этот файл предназначен для хранения архива изделий; компоненты файла FileOut имеют тип записей ProductType.

Информация для всех текстовых файлов, кроме файла FileRes, готовится вручную с клавиатуры ПЭВМ, что заранее предполагает наличие в этой информации разного рода ошибок. Содержание текстового файла легко изменить с помощью любого текстового редактора; это создает опасность внесения во входную информацию умышленных или неумышленных ошибок. Поэтому использовать текстовые файлы как непосредственный источник обрабатываемых данных в АСУ нецелесообразно.

В реальных системах входные документы, которые размещаются в текстовых файлах, контролируются в программе с целью выявления возможных ошибок. После их исправления содержимое текстового файла выдается на печать для визуального контроля; тщательно проверенные документы программно записываются в типизированные файлы, которые представляют собою архив системы.

С точки зрения машинной обработки типизированные файлы имеют ряд преимуществ по сравнению с текстовыми файлами:

- возможность прямого доступа;
- более экономное размещение данных в памяти;
- более быстрое выполнение операций ввода-вывода.

В текстовом файле информация имеет литерное представление (каждый символ текста или цифра числа занимают отдельный байт). В типизированном файле представление информации не отличается от машинного. Поэтому при обращении к текстовому файлу затрачивается значительное машинное время на преобразование информации из внешнего представления во внутреннее (или наоборот).

Примечание. Контроль исходных данных включен в задания по лабораторной работе №9.

В программе Labor7 для хранения архива предусмотрен файл FileOut. Однако достоверность информации при переписи из файла FileInput в файл FileOut не контролируется. Контроль входной информации, как и ряд других операций, обязательных в реальных АУС, не выполняется по одной причине: при реализации этих операций потребовалось бы значительное увеличение объема программы. По этой же причине не используется типизированный файл для кодификатора изделий.

В программе Labor7 предусмотрены следующие режимы работы:

- создание архива изделий;
- сортировка компонентов архива;
- печать архива изделий;
- печать кодификатора изделий;
- добавление компонентов в архив;
- удаление компонента из архива;
- изменение компонента в архиве;
- обработка архива.

Выбор режимов работы реализован в основной части программы. Сначала на экран выдается запрос о режиме; после ввода с клавиатуры ответа (0..8) оператор **Case** выполняет обращение к процедуре, которая определяет соответствующий режим.

При вводе числового ответа с клавиатуры существует вероятность ошибочного набора (например, буква вместо цифры). При вводе ответа с помощью оператора Read это вызвало бы прерывание программы и, как следствие, необходимость ее повторного запуска. Чтобы блокировать такое прерывание, ввод переменной KeyRegime производится с помощью процедуры GetNumber, которая выполняет контроль формата вводимого числа и допустимого

диапазона его представления. При ошибке ввода программа выдает соответствующее сообщение и предлагает пользователю повторить ввод.

Запрос режима работы повторяется оператором **Repeat** до тех пор, пока не будет получен ответ KeyRegime=0. Это дает возможность пользователю любое количество раз и в любом порядке задавать необходимые режимы работы программы.

Рассмотрим теперь отдельно каждый из режимов работы.

1. ***Создание архива изделий*** (процедура CreateArchive).

В процедуре CreateArchive производится чтение строк файла FileInput, формирование записи Product (с помощью процедуры ReadProduct) и передача этой записи в типизированный файл FileOut. Здесь следует обратить внимание на то, что из текстового файла читается отдельно каждый компонент записи Product, а в типизированный файл передается целиком вся эта запись.

После создания архивного файла выполняется формирование массива кодификаторов Kodifs, для чего используется процедура MakeKodifs.В дальнейшем процедура SortKodif выполняет группировку элементов массива Kodifs по возрастанию кода Kod. Таким образом, при активизации любого режима работы программы в оперативной памяти уже будет находиться упорядоченный массив Kodifs.

Следует обратить внимание еще на одно обстоятельство. В описании кодификатора Kodif указано два компонента: Kod и Name. В то же время в процедуре ReadKodif, осуществляющей ввод кодификатора, оператор **Case** имеет три, а не две альтернативы. Этим предполагается, что название изделия Name может состоять из одного или двух слов. Тогда при k=3 к первому слову названия присоединяется второе слово, при этом они будут разделены одним пробелом.

2. ***Сортировка компонентов архива*** (процедура SortArchive);

В этом режиме сначала с помощью процедуры ReadFileOut выполняется формирование массива записей Products при чтении компонентов файла FileOut. После этого известным методом прямой выборки выполняется группирование массива Products по возрастанию элемента Kod. В процедуре SortArchive, в частности, наглядно продемонстрировано, что любой записи можно одним оператором присвоить значение другой записи такого же типа (а не отдельно по компонентам).

Процедура SortArchive содержит один формальный параметр pk. Если pk = 0, то в финальной части этой процедуры на экран выводится сообщение об окончании сортировки архива, при pk ≠0 такое сообщение не выводится. Значение pk = 0 применяется при непосредственной активизации режима сортировки из меню программы, значение pk ≠0 - при обращении к процедуре SortArchive из других режимов (например, при добавлении компонентов в архив).

3. ***Печать архива и кодификатора изделий*** (процедуры PrintArchive и PrintKodif);

В программе Labor7 может быть использовано три варианта вывода результатов на внешние устройства:

- только экран;
- экран и магнитный диск;
- экран и принтер.

Для выбора варианта вывода при старте программы производится запрос к пользователю, после чего переменной Device присваивается значение 0, 1 или 2.

При выводе на магнитный диск используется текстовый файл FileRes. Такой вариант вывода предоставляет пользователю две дополнительные возможности:

- чтение и коррекция содержимого файла результатов с помощью любого текстового редактора (например, для улучшения эстетичного вида таблиц);
- запись результатов в текстовый файл для дальнейшего их воспроизведения на принтере (особенно это полезно при отсутствии принтера на компьютере, который используется для разработки программы).

Непосредственный вывод результатов выполняет процедура `WritelnString`, входным параметром которой является строка `S`. В связи с этим перед каждым обращением к этой процедуре производится формирование выводимых данных в виде строки.

Для печати шапок выходных таблиц используются символы псевдографики (см.методические указания к лабораторной работе №2).

4. ***Добавление компонентов в архив*** (процедура `AddArchive`).

В основной части процедуры `AddArchive` сначала формируется массив `Products` (процедурой `ReadFileOut`), после чего происходит последовательное чтение строк файла `FileAdd` и включение их в состав массива `Products` таким образом, чтобы при этом не была нарушена упорядоченность массива по возрастанию кода изделия. Если же при старте режима было обнаружено, что массив `Products` не сгруппирован (`SignSort = false`), то предварительно выполняется сортировка этого массива. Расширенный массив `Products` записывается процедурой `WriteFileOut` снова в файл `FileOut`.

При добавлении каждого компонента, прочитанного из файла `FileAdd`, проверяется, нет ли такого компонента в составе массива `Products`. Если такой факт обнаружен, компонент не записывается в этот массив, а на экран выводится соответствующее сообщение.

5. ***Удаление компонента из архива*** (процедура `DeleteArchive`).

В данном режиме сначала по запросу программы читается код изделия `Kod` удаляемого компонента. Процедура `SearchProduct` определяет местоположение этого компонента в массиве `Products`. Если такой компонент обнаружен (`k>0`), то все расположенные после него компоненты массива `Products` сдвигаются на один элемент к началу массива, а размер `n` массива `Products` уменьшается на единицу. В противном случае на экран выдается сообщение о том, что компонента с кодом `Kod` в архиве нет.

Поскольку элементы массива `Products` сгруппированы по возрастанию целочисленной переменной `Kod`, то поиск компонента со значением `Kod` осуществляется в процедуре `SearchProduct` методом бинарного поиска, что значительно сокращает время работы процедуры по сравнению с методом прямого перебора.

6. ***Изменение компонента в архиве*** (процедура `ChangeArchive`).

Поиск изменяемого компонента производится аналогично предыдущему режиму, после чего с помощью процедуры `MakeComponent` читаются новые значения изменяемых реквизитов.

7. ***Обработка архива*** (процедура `WorkUpArchive`).

В этом режиме дважды просматривается массив `Products`. При первом просмотре по заданному пользователем номеру цеха `Shop` выбираются из массива `Products` соответствующие компоненты и рассчитываются суммы плановых и фактических показателей. В результате первого просмотра печатается таблица сведений о выполнении плана по цеху `Shop`. При втором просмотре в таблицу выдается список изделий, по которым не выполнен годовой план производства. Наименование изделия выбирается из кодификатора (массив `Kodifs`), для поиска компонента в массиве `Kodifs` применяется метод бинарного поиска (процедура `SearchKodif`).

8. ***Использование переменных `SignArchive` и `SignSort`.***

В программе Labor7 предусмотрено 8 режимов работы, выбор которых осуществляет переменная KeyRegime. Конечно, пользователь может активизировать их в любом порядке, тем не менее режимы 2..8 могут выполняться лишь после создания архива изделий. Это обстоятельство учитывается переменной SignArchive, которая принимает значение true лишь при создании архивного файла. Если SignArchive = false, то активизация других режимов блокируется с выдачей на экран соответствующего сообщения.

В процедурах DeleteArchive и ChangeArchive по запросу программы вводится код компоненты, которую надо удалить или изменить в ней некоторые реквизиты. Поиск заданной компоненты выполняется бинарным методом в процедуре SearchProduct. Такой метод может функционировать лишь в том случае, когда массив Products уже сгруппирован по возрастанию кода Kod. Надзор за этим обстоятельством выполняет переменная SignSort, которая принимает значение true при окончании работы процедуры SortArchive. Если при обращении к процедурам DeleteArchive или ChangeArchive будет выявлено, что SignSort = false, то без вмешательства пользователя в начале работы этих процедур выполняется группировка массива записей Products. Такая же работа, как это было отмечено выше, производится при добавлении компонентов в архив.

9. ***Ввод с клавиатуры текстовой переменной.***

Предположим, что нам нужно по запросу программы ввести с клавиатуры целочисленную переменную. Это можно сделать таким образом:

```
Var  k : integer;
.......................
Writeln('Введите  переменную  k');
Read(k);
```

При этом не имеет значения, будут ли записаны пробелы перед числом k или после этого числа. Процедура Read автоматически находит в буфере ввода местоположение первой и последней цифры числа, превращает число в машинный формат и записывает его в соответствующее поле памяти.

Иная ситуация имеет место при вводе текстовой переменной.

Пусть в программе записано следующее::

```
Var   S : string;
.......................
Writeln('Введите  строку  S');
Read(S);
```

Разумеется, перед строкой S и после нее не должно быть лишних пробелов, так как пробел – это такой же символ для строки, как и другие. Приведенный пример имеет два недостатка:

1) процедура Read не выполняет сброса буфера ввода (для файла – это переход на следующую строку);
2) перед вводом строки необходимо очистить буфер ввода, что выполняется процедурой Readln без параметров, но не процедурой Read.

Поэтому поставленную задачу надо решать следующим образом:

```
Var   S : string;
.......................
Writeln('Введите  строку  S');
Readln;
Readln(S);
```

Конкретный пример указанной методики можно увидеть в процедуре ChangeArchive (внутренняя процедура MakeComponent).

10. ***Использование процедуры FillString.***

Предположим, что в текстовом массиве есть строка

```
St[i] = 'abcdefgh';
```

По запросу программы пользователем было указано найти индекс этой строки в массиве. При этом было введено для сравнения значение

`S = 'abcdefgh  '` (3 пробела после символа “h”).

При сравнении двух строк неодинаковой длины более короткая дополняется символами #0. Так как символ пробела имеет номер #32, то программа будет считать, что строки `St[i]` и `S` неодинаковы, а это приводит к ошибкам в работе программы.

Чтобы этого не случилось, с помощью процедуры `FillString` каждая из введенных строк дополняется пробелами до объявленной длины.

11. ***Об использовании кодификаторов.***

В лабораторной работе №7 предполагается, что вся информация, которая отображается в записи, при вводе размещается в одной строке входного файла, а вся результативная информация – в одной строке таблицы на экране. При этом нужно стремиться, чтобы длина строки таблицы не превышала 66-68 позиций (иначе при печати строка таблицы может не разместиться на бумаге формата А4).

Если в задаче предполагается вывод в составе таблицы какого-нибудь названия, которое приводит к превышению рекомендованной длины строки на экране, то в этом случае целесообразно использовать кодификатор. Тогда в таблице надо отображать код этого названия, а соответствие между кодом и названием печатать отдельно.

12. ***Об использовании кода в виде строки*****.**

Как уже было показано, в состав режимов программы Labor7 при обработке архива входит удаление компогнгта архива или изменение его реквизитов. Для этого нужно указать конкретный компонент в архиве, что не всегда возможно по его наименованию (например, в списке студентов могут быть две одинаковые фамилии, в том числе и инициалы). Уникальность компонента обеспечивает его код, который в приводимых далее заданиях может называться «шифр», «инвентарный номер», «артикул» и т.п. Зачастую код задают в виде набора букв и цифр и описывают как строку-переменную. В частности, в рассмотренном примере вместо типа longint для кода был бы использован тип string[6].

Ввод и использование кода в виде строки ничем не отличается от манипуляций по отношению другим строковым переменным, кроме одного случая, а именно ввода кода с клавиатуры по запросу программы в режимах удаления и изменения компонента. Специфика такого ввода рассмотрена выше в п.9.

13. **Порядок выполнения лабораторной работы №7.**

Программа, которая реализует лабораторную работу №7, имеет довольно значительный размер (в программе `Labor7` 787 строк). Поэтому, как и для любой программы среднего или большого размера, нецелесообразно с самого начала кодировать и отлаживать ее в полном объеме.

Ниже приводятся рекомендации о порядке программной реализации лабораторной работы №7. Здесь учитывается, что в каталоге MetLab приведены полные исходные тексты всех лабораторных работ, в том числе и лабораторной работы № 7. Поэтому для экономии времени на проектирование программы для конкретного задания целесообразно в максимальной степени использовать пример, отображенный в каталоге MetLab.

1) Как правило, в заданиях по лабораторной работе № 7 кодификатор не используется. Тогда нужно выполнить следующее:

- в разделе **Const** удалить константу MaxKodif;
- в разделе **Type** удалить типы KodifType и KodifAr;
- в разделе **Var** удалить переменные nk, Kodif, Kodifs, FileKodif;
- удалить процедуры ReadKodif, MakeKodifs, SortKodif, SearchKodif, PrintKodif;
- в разделе операторов удалить обращения к процедурам Assign(FileKodif,’Kodif.txt’), PrintString(17,13,4) и 4: PrintKodif; в последних двух случаях изменить нумерацию нижележащих строк.

2) В разделах **Type** и **Var** заменить часть типов и переменных содержательными именами, которые соответствовали бы условию задачи. Если в задаче идет речь, например, об автомобилях, то было бы странно использовать имя Product (‘изделие’). Здесь вместо имен ProductType, ProductAr, Product и Products уместно записать AutoType, AutoAr, Auto, Autos.

3) Начальный этап – это создание архива. В этом случае должны работать лишь те процедуры, которые обеспечивают формирование и печать архива. Для этого

- в разделе операторов оставить лишь обращения CreateArchive и PrintArchive, остальные закомментировать (например, «2: {SortArchive} ;» и т.д.);

- оставить без изменения сервисные процедуры и функции (Space, PrintHat и др.);

- изменить необходимым образом тексты процедур ReadProduct, CreateArchive, ReadFileOut (изменив, разумеется, и имя Product на имя, например, Auto);

- в процедуре PrintArchive установить комментарий высшего уровня «(* *)» между обращениями Reset(FileOut) и Close(FileOut). На начальном этапе отладки эстетика печати не нужна, требуется лишь убедиться, что создание архива выполняется верно. Поэтому рядом с закомментированной частью процедуры PrintArchive записать:

```
i:=0;
While not eof(FileOut) do
  Begin
    Read(FileOut,Auto);
    Inc(i);
    Write(I,’  ‘);
    Write(Auto.Name, ...);
    Writeln
  End;
```

- в процедурах SearchProduct, SortArchive, AddArchive, DeleteArchive, ChangeArchive, WorkUpArchive, не изменяя их текста (заранее трудно решить, что будет изменено в этих текстах), в их разделах операторов установить комментарии высшего уровня.

4) После отладки режима создания архива изменить процедуру PrintArchive для печати в виде таблицы. Здесь нужно обратить внимание на следующее.

Значения строк St[1] .. St[6] разделены на две части, чтобы печатаемая строка не занимала свыше 66 позиций. В то же время для формирования новой шапки удобно иметь эти строки неразорванными. Для этого, например, в строке St[2] нужно удалить символы «’+’» и нажать клавишу Enter.

5) Вводить в действие последовательно режимы сортировки, добавления, удаления, изменения и, наконец, обработки архива. При этом в операторе Case и в соответствующих процедурах удалить установленные ранее комментарии и изменить нужным образом тексты этих процедур.

14. ***О печати результатов***.

Как было ранее указано, в программе Labor7 предусмотрено три варианта вывода на внешнее устройство:

- только экран;

- экран и магнитный диск;

- экран и принтер.

Второй вариант целесообразно использовать для окончательной печати результатов работы программы. Он позволяет:

- сохранить для отчета результаты, если нет принтера или принтер неисправен;

- вставить дополнительные фразы в текст результатов, чтобы было видно, что вначале отпечатан исходный архив, затем архив с дополнением, с удалением и т.д.

Текст программы Labor7:

```
Program Labor7;
{ Создание, печать, коррекция и обработка архива сведений }
{ о продукции, которая выпускается цехами предприятия }
Uses Crt,Printer;
Const
    Nmax = 400;      { макс. количество элементов в архиве }
    MaxKodif = 50;   { макс. количество компонентов кодификатора }
    Enter = 13;      { код клавиши Enter }
    PressKey = 'Нажмите клавишу ENTER';
Type
    ProductType = record  { тип компонента архива изделий }
      NumberShop : byte;  { номер цеха }
      Kod : longint;      { код изделия }
      Dimens : string[5]; { единица измерения }
      Price  : real;      { цена изделия }
      Plan,               { план выпуска по полугодиям }
      Fact                { факт.выпуск по полугодиям }
             : array[1..2] of real;
    end;
    ProductAr = array[1..Nmax] of ProductType;
    KodifType = record    { тип компонента кодификатора }
      Kod : longint;      { код изделия }
      Name : string[35];  { наименование изделия }
    end;
    KodifAr = array[1..MaxKodif] of KodifType;
    string80 = string[80];
    StringAr = array[1..10] of string80;
Var
    np : word;          { количество компонентов архива }
    nk,                 { количество компонентов кодификатора }
    KeyRegime,          { ключ выбора режима работы }
    Device : byte;      { устройство вывода результатов: }
                        { 0 - экран; 1 - экран и магн.диск; }
                        { 2 - экран и принтер }
    SignArchive,        { признак создания архива }
    SignSort : boolean; { признак сортировки архива }
    Reply : char;       { символ нажатой клавиши }
    Product : ProductType;{ компонент архива }
    Products : ProductAr; { массив компонентов архива }
    Kodif : KodifType;    { компонент кодификатора }
    Kodifs : KodifAr;     { массив компонентов кодификатора }
    St : StringAr;        { строки для печати таблиц }
    FileInput,          { файл входных документов }
    FileAdd,            { файл добавляемых документов }
    FileKodif,          { файл кодификатора изделий }
    FileRes : text;     { файл результатов }
    FileOut             { архивный файл изделий }
        : file of ProductType;
{ ------------------------------------------------------ }
Procedure WaitEnter;
{ Задержка выполнения программы, пока не будет нажата }
{ клавиша Enter }
Var ch : char;
Begin
  Repeat
```

```
    ch:=ReadKey;
  Until ord(ch) = Enter;
End { WaitEnter };
{ ------------------------------------------------------ }
Procedure PrintString(X,Y:byte; S:string);
{ Печать строки S с позиции X строки экрана с номером Y }
Begin
  GotoXY(X,Y); Write(S);
End { PrintString };
{ ------------------------------------------------------ }
Procedure WritelnString(S:string);
{ Вывод строки на экран, магнитный диск и принтер }
Begin
  Writeln(S);
  Case Device of
    1 : Writeln(FileRes,S)
    2 : Writeln(Lst,S);
  end;
End { WritelnString };
{ ------------------------------------------------------ }
Procedure PrintKeyAndWaitEnter;
{ Печать строки-константы PressKey с позиции 1 строки экрана 25 }
{ и задержка выполнения программы до нажатия клавиши Enter }
Begin
  PrintString(1,25,PressKey);
  WaitEnter;
  ClrScr;
End { PrintKeyAndWaitEnter };
{ ------------------------------------------------------ }
Procedure CheckPageScreen(Var j:byte);
{ Контроль размера страницы на экране }
Const LengthPage = 23; { количество строк на одной странице }
      Sr = 'Следующая страница - Enter ';
Begin
  Inc(j);
  If j=LengthPage then
    Begin
      j:=0;
      PrintString(1,25,Sr);
      WaitEnter;
      ClrScr;
    End;
End { CheckPageScreen };
{ ------------------------------------------------------ }
Function Space(S:string; k:byte):byte;
{ Поиск ближайшего пробела }
Var  i : byte;
Begin
  Space:=0;
  For i:=k to length(S) do
    If S[i]=' ' then
      Begin
        Space:=i; Exit
      End;
End { Space };
```

```
{ ------------------------------------------------------- }
Function NotSpace(S:string; k:byte):byte;
{ Поиск ближайшего непробела }
Var  i : byte;
Begin
  NotSpace:=0;
  For i:=k to length(S) do
    If S[i]<>' ' then
      Begin
        NotSpace:=i; Exit
      End;
End { NotSpace };
{ ------------------------------------------------------- }
Function FillString(S:string; len,pk:byte):string;
{ Дополнение строки S до длины len пробелами: }
{ слева (pk=0) или справа (pk=1)               }
Var  i : byte;
Begin
  If length(S)<len then
    For i:=1 to len-length(S) do
      If pk=0 then
        S:=' '+S
      Else
        S:=S+' ';
  FillString:=S;
End { FillString };
{ ------------------------------------------------------- }
Function GetNumber(MinNumber,MaxNumber:real;
                   m1,n1,m2,n2:byte):real;
{ Ввод числа с клавиатуры;  MinNumber,MaxNumber - допусти- }
{ мый диапазон;  m1,n1,m2,n2 - формат сообщения о допусти- }
{ мом диапазоне }
Var  k : integer;
     Number : real;
     Cond : boolean;
Begin
  Repeat
    Cond:=true;
    {$I-} Read(Number); {$I+}
    k:=IOResult;
    If k<>0 then
      Begin
        Writeln(#7'Неправильный формат числа');
        Writeln('Повторите ввод');
        Cond:=false
      End
    Else
      If (Number<MinNumber) or (Number>MaxNumber) then
        Begin
          Write(#7'Число должно быть в диапазоне от ,');
          Writeln(MinNumber:m1:n1,' до ',MaxNumber:m2:n2,'.');
          Writeln('Повторите ввод');
          Cond:=false
        End;
  Until Cond;
```

```
  GetNumber:=Number;
End { GetNumber };
{ ------------------------------------------------------- }
Procedure UsesDevice;
{ Запрос об устройстве для вывода результатов }
Begin
  Writeln('Укажите устройство для вывода результатов: ');
  Writeln('   0 - только экран');
  Writeln('   1 - экран и магнитный диск');
  Writeln('   2 - экран и принтер');
  Device:=Round(GetNumber(0,2,1,0,1,0));
End { UsesDevice };
{ ------------------------------------------------------- }
Procedure PrintHat(n:byte);
{ Вывод шапки таблицы из n строк }
Var  i : byte;
Begin
  For i:=1 to n do
    WritelnString(St[i]);
End { PrintHat };
{ ------------------------------------------------------- }
Procedure ReadProduct(Var F:text);
{ Ввод из текст.файла FileInput или FileAdd компонента архива }
Var k,k1,k2 : byte;
     Code : integer;
     Cond : boolean;
     Sa,Sb : string80;
Begin
  With Product do
    Begin
      Readln(F,Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1; Cond:=false
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Val(Sb,NumberShop,Code);
                2 : Val(Sb,Kod,Code);
                3 : Dimens:=FillString(Sb,5,1);
                4 : Val(Sb,Price,Code);
                5 : Val(Sb,Plan[1],Code);
                6 : Val(Sb,Plan[2],Code);
                7 : Val(Sb,Fact[1],Code);
                8 : Val(Sb,Fact[2],Code);
```

```
              end;
            End;
        End;
    End;
End { ReadProduct };
{ ------------------------------------------------------ }
Procedure ReadKodif;
{ Ввод из текстового файла FileKodif компонента кодификатора }
Var  k,k1,k2 : byte;
      Code : integer;
      Cond : boolean;
     Sa,Sb : string80;
Begin
  With Kodif do
    Begin
      Readln(FileKodif,Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                  k2:=length(Sa)+1; Cond:=false
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Val(Sb,Kod,Code);
                2 : Name:=FillString(Sb,35,1);
                3 : Begin
                      While Name[length(Name)]=' ' do
                        Delete(Name,length(Name),1);
                      Name:=Name+' '+Sb;
                      Name:=FillString(Name,35,1);
                    End;
              end;
            End;
        End;
    End;
End { ReadKodif };
{ ------------------------------------------------------ }
Procedure MakeKodifs;
{ Чтение файла FileKodif и формирование массива Kodifs }
Begin
  Reset(FileKodif);
  nk:=0;
  While not eof(FileKodif) do
    Begin
      ReadKodif;
      Inc(nk); Kodifs[nk]:=Kodif;
```

```
    End;
  Close(FileKodif);
End { MakeKodifs };
{ ------------------------------------------------------ }
Procedure SortKodif;
{ Группирование массива Kodifs по возрастанию компонента Kod }
Var  i,m : byte;
     Kod : longint;
    Cond : boolean;
Begin
  Cond:=true; m:=nk-1;
  While Cond do
    Begin
      Cond:=false;
      For i:=1 to m do
        Begin
          Kod:=Kodifs[i].Kod;
          If Kod>Kodifs[i+1].Kod then
            Begin
              Kodif:=Kodifs[i]; Kodifs[i]:=Kodifs[i+1];
              Kodifs[i+1]:=Kodif; Cond:=true;
            End;
        End;
      Dec(m);
    End;
End { SortKodif };
{ ------------------------------------------------------ }
Procedure CreateArchive;
{ Создание архивного файла изделий и массива Kodif }
Begin
  Reset(FileInput); Rewrite(FileOut);
  While not eof(FileInput) do
    Begin
      ReadProduct(FileInput);
      Write(FileOut,Product);
    End;
  Close(FileInput); Close(FileOut);
  SignArchive:=true;
  SignSort:=false;
  MakeKodifs; SortKodif;
  Writeln(' Создание архива закончено');
  PrintKeyAndWaitEnter;
End { CreateArchive };
{ ------------------------------------------------------ }
Procedure ReadFileOut(Var n:word);
{ Чтение из архива и формирование массива записей Products }
Begin
  Reset(FileOut);
  n:=0;
  While not eof(FileOut) do
    Begin
       Inc(n);
       Read(FileOut,Products[n]);
    End;
  Close(FileOut);
```

```
End { ReadFileOut } ;
{ ------------------------------------------------------ }
Procedure WriteFileOut(n:word);
{ Пересылка массива записей Products в архив }
Var i : word;
Begin
  Rewrite(FileOut);
  For i:=1 to n do
    Write(FileOut,Products[i]);
  Close(FileOut);
End { WriteFileOut };
{ ------------------------------------------------------ }
Function SearchProduct(Kod:longint):word;
{ Бинарный поиск в массиве Products компонента с кодом Kod }
Var  i1,i2,m : word;
Begin
  SearchProduct:=0;
  i1:=1; i2:=np;
  While i1<=i2 do
    Begin
      m:=(i1+i2) div 2;
      If Kod=Products[m].Kod then
        Begin
          SearchProduct:=m; Exit
        End
      Else
        If Kod>Products[m].Kod then
          i1:=m+1
        Else
          i2:=m-1;
    End;
End { SearchProduct };
{ ------------------------------------------------------ }
Function SearchKodif(Kod:longint):byte;
{ Бинарный поиск в массиве Kodifs элемента с кодом Kod }
Var  i1,i2,m : byte;
Begin
  SearchKodif:=0;
  i1:=1; i2:=nk;
  While i1<=i2 do
    Begin
      m:=(i1+i2) div 2;
      If Kod=Kodifs[m].Kod then
        Begin
          SearchKodif:=m; Exit
        End
      Else
        If Kod>Kodifs[m].Kod then
          i1:=m+1
        Else
          i2:=m-1;
    End;
End  { SearchKodif };
{ ------------------------------------------------------ }
Procedure SortArchive(pk:byte);
```

```
{ Сортировка архива по возрастанию кода изделия }
Var   i,imax,m : word;
     KodMax : longint;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut(np);
  m:=np;
  While m>1 do
    Begin
      KodMax:=Products[1].Kod; imax:=1;
      For i:=2 to m do
        If Products[i].Kod>KodMax then
          Begin
            KodMax:=Products[i].Kod; imax:=i
          End;
      If imax<m then
        Begin
          Product:=Products[m];
          Products[m]:=Products[imax];
          Products[imax]:=Product;
        End;
      Dec(m);
    End;
  WriteFileOut(np);
  SignSort:=true;
  If pk=0 then           { вывод сообщения при pk = 0 }
    Begin
      Writeln('Сортировка архива закончена');
      PrintKeyAndWaitEnter;
    End;
End { SortArchive };
{ ------------------------------------------------------- }
Procedure PrintArchive;
{ Вывод архива изделий на экран, принтер и магнитный диск }
Var i : word;
    j : byte;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  Reset(FileOut);
  St[1]:='             АРХИВ ВЕДОМОСТЕЙ О ВЫПУСКАЕМОЙ ПРОДУКЦИИ';
  St[2]:='-----------------------------------------------'+
         '--------------';
  St[3]:='| N  | N  |  Код  |Един.|        | План выпуска |'+
         ' Факт.выпуск |';
  St[4]:='|п/п |цеха|изделия|изм. | Цена |--------------|'+
```

```
        '-------------|';
  St[5]:='|    |       |       |       |       |   1   |   2   '+
        '   1  |   2  |';
  St[6]:='-----------------------------------------------'+
        '-------------|';
  PrintHat(6);
  j:=6; i:=0;
  While not eof(FileOut) do
    Begin
      Read(FileOut,Product);
      Inc(i); Str(i:3,St[1]);
      With Product do
        Begin
          Str(NumberShop:3,St[2]); Str(Kod:6,St[3]);
          St[4]:=Dimens;  Str(Price:6:2,St[5]);
          Str(Plan[1]:6:0,St[7]); Str(Plan[2]:6:0,St[8]);
          Str(Fact[1]:6:0,St[9]); Str(Fact[2]:6:0,St[10]);
        End;
      St[1]:='|'+St[1]+'. '+St[2]+'   '+St[3]+'   '+St[4]+
             St[5]+'  '+St[7]+'  '+St[8]+'  '+St[9]+'  '+
             St[10]+' |';
      WritelnString(St[1]);
      CheckPageScreen(j);
    End;
  St[6]:='-----------------------------------------------'+
        '--------------';
  WritelnString(St[6]);
  Close(FileOut);
  PrintKeyAndWaitEnter;
End { PrintArchive };
{ ------------------------------------------------------ }
Procedure PrintKodif;
{ Вывод кодификатора изделий на экран, принтер и магнитный диск }
Var i,j : byte;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  St[1]:='                КОДИФИКАТОР ВЫПУСКАЕМОЙ ПРОДУКЦИИ';
  St[2]:='-----------------------------------------------'+
        '---------------';
  St[3]:='| N п/п |   Код изделия  |   Наименование изделия'+
        '              |';
  St[4]:='|-----------------------------------------------'+
        '--------------|';
  PrintHat(4);
  j:=4;
  For i:=1 to nk do
    Begin
      Kodif:=Kodifs[i];
      Str(i:4,St[1]); Str(Kodif.Kod:6,St[2]);
      St[1]:='|'+St[1]+'.       '+St[2]+'         '+
```

```
              Kodif.Name+'  |';
      WritelnString(St[1]);
      CheckPageScreen(j);
     End;
  St[4]:='-----------------------------------------------'+
         '---------------';
  WritelnString(St[4]);
  PrintKeyAndWaitEnter;
End { PrintKodIf };
{ ------------------------------------------------------- }
Procedure AddArchive;
{ Добавление компонентов в архив изделий }
Var  k : word;
    Sr : string;
{ ------------------------------- }
Procedure Addition;
{ Добавление одного компонента в архив }
Label  10;
Var  i,k : word;
Begin
  k:=0;
  For i:=1 to np do
    If Product.Kod<Products[i].Kod then
      Begin
        k:=i; Goto 10
      End;
  10:
  If k=0 then
    k:=np+1
  Else
    For i:=np+1 downto k+1 do
      Products[i]:=Products[i-1];
  Products[k]:=Product
End { Addition };
{ ------------------------------- }
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut(np);
  If not SignSort then
    Begin
      SortArchive(1); SignSort:=true;
    End;
  Reset(FileAdd);
  While not eof(FileAdd) do
    Begin
      ReadProduct(FileAdd);
      k:=SearchProduct(Product.Kod);
      If k>0 then
        Begin
          Str(Product.Kod,Sr);
```

```
          WritelnString('В архиве уже есть изделие с кодом '+Sr);
        End
      Else
        Addition;
    End;
  WriteFileOut(np);
  Close(FileAdd);
  Writeln('Дополнение архива закончено');
  PrintKeyAndWaitEnter;
End { AddArchive };
{ ------------------------------------------------------ }
Procedure DeleteArchive;
{ Удаление компонента из архива изделий }
Var  i,k : word;
     Kod : longint;
     Sr : string;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut(np);
  If not SignSort then
    Begin
      SortArchive(1); SignSort:=true;
    End;
  ClrScr;
  WritelnString('Укажите код удаляемого компонента');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);,
  k:=SearchProduct(Kod);
  If k>0 then
    Begin
      For i:=k to np-1 do
        Products[i]:=Products[i+1];
      Dec(np);
      WriteFileOut(np);
      Writeln('Удаление компонента из архива закончено');
    End
  Else
    WritelnString('В архиве нет компонента с кодом  '+Sr);
  PrintKeyAndWaitEnter;
End { DeleteArchive };
{ ------------------------------------------------------ }
Procedure ChangeArchive;
{ Изменение компонента в архиве изделий }
Var   k : word;
     Kod : longint;
    Sr : string;
{ ------------------------------------------------------ }
Procedure MakeComponent;
{ Формирование изменяемого компонента }
Var  k,k1,k2 : byte;
```

```
        Code : integer;
        Cond : boolean;
       Sa,Sb : string80;
Begin
  With Product do
    Begin
      Readln; Readln(Sa);
      WritelnString(Sa);
      Cond:=true; k2:=0; k:=0;
      While Cond do
        Begin
          k1:=NotSpace(Sa,k2+1);
          If k1=0 then
            Cond:=false
          Else
            Begin
              k2:=Space(Sa,k1+1);
              If k2=0 then
                Begin
                   k2:=length(Sa)+1; Cond:=false
                End;
              Inc(k);
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Dimens:=FillString(Sb,5,1);
                2 : Val(Sb,Price,Code);
                3 : Val(Sb,Plan[1],Code);
                4 : Val(Sb,Plan[2],Code);
                5 : Val(Sb,Fact[1],Code);
                6 : Val(Sb,Fact[2],Code);
              end;
            End;
        End;
    End;
End { MakeComponent };
{ ------------------------------------------------------- }
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut(np);
  If not SignSort then
    Begin
      SortArchive(1); SignSort:=true;
    End;
  ClrScr;
  WritelnString('Укажите код изменяемого компонента');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);
  k:=SearchProduct(Kod);
  If k>0 then
    Begin
```

```
      WritelnString('     Укажите такие реквизиты :');
      WritelnString('ед.измерения цена план-1 план-2 факт-1 факт-2');
      MakeComponent;
      Products[k]:=Product;
      WriteFileOut(np);
      Writeln('Изменение компонента в архиве закончено');
    End
  Else
    WritelnString('В архиве нет компонента с кодом '+Sr);
  PrintKeyAndWaitEnter;
End { ChangeArchive };
{ ------------------------------------------------------- }
Procedure WorkUpArchive;
{ Обработка архива изделий }
Var   i : word;             { параметр цикла }
  Shop,                     { номер цеха }
  j,k,m : byte;
    Kod : longint;          { код изделия }
  PlanSt1,PlanSt2,PlanGod, { суммарные плановые показатели }
  FactSt1,FactSt2,FactGod, { суммарные фактические показатели }
  ProcSt1,ProcSt2,ProcGod, { процент выполнения плана }
  BufSt : real;             { буферная переменная }
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut(np);
  Writeln('Укажите номер цеха');
  Shop:=Round(GetNumber(1,99,1,0,2,0));
  PlanSt1:=0; PlanSt2:=0;
  FactSt1:=0; FactSt2:=0; k:=0;
  For i:=1 to np do
    If Shop=Products[i].NumberShop then
      Begin
        Inc(k);
        BufSt:=Products[i].Plan[1]*Products[i].Price;
        PlanSt1:=PlanSt1+BufSt;
        BufSt:=Products[i].Plan[2]*Products[i].Price;
        PlanSt2:=PlanSt2+BufSt;
        BufSt:=Products[i].Fact[1]*Products[i].Price;
        FactSt1:=FactSt1+BufSt;
        BufSt:=Products[i].Fact[2]*Products[i].Price;
        FactSt2:=FactSt2+BufSt;
      End;
  If k=0 then
    Writeln('В архиве нет сведений о цехе ',Shop)
  Else
    Begin
      PlanGod:=PlanSt1+PlanSt2; FactGod:=FactSt1+FactSt2;
      ProcSt1:=100*FactSt1/PlanSt1;
      ProcSt2:=100*FactSt2/PlanSt2;
      ProcGod:=100*FactGod/PlanGod;
```

```
    ClrScr; Str(Shop:2,St[1]);
    St[1]:='        СВЕДЕНИЯ О ВЫПОЛНЕНИИ ПЛАНА ПО ЦЕХУ '
           +St[1];
    St[2]:='-----------------------------------------'+
          '-----------------';
    St[3]:='|          |    Полугодие                |'+
          '   За    |';
    St[4]:='|               -------------------------------'+
          ' год   |';
    St[5]:='|               |      1     |     2      |'+
          '              |';
    St[6]:='-----------------------------------------'+
          '--------------';
    PrintHat(6);
    Str(PlanSt1:10:2,St[1]); Str(PlanSt2:10:2,St[2]);
    Str(PlanGod:10:2,St[3]);
    St[1]:='| План выпуска     |  '+St[1]+'  |  '+St[2]+
          '|  '+St[3]+' |';
    WritelnString(St[1]);
    Str(FactSt1:10:2,St[1]); Str(FactSt2:10:2,St[2]);
    Str(FactGod:10:2,St[3]);
    St[1]:='| Фактический выпуск  | '+St[1]+' |  '+St[2]+
          '|  '+St[3]+' |';
    WritelnString(St[1]);
    Str(ProcSt1:8:2,St[1]); Str(ProcSt2:8:2,St[2]);
    Str(ProcGod:8:2,St[3]);
    St[1]:='| Процент выполнения | '+St[1]+' | '+
          St[2]+'  |  '+St[3]+' |';
    WritelnString(St[1]);
    St[6]:='-----------------------------------------'+
          '--------------';
    WritelnString(St[6]);
  End;
PrintKeyAndWaitEnter;
St[1]:=' ';
St[2]:=' СПИСОК ИЗДЕЛИЙ, ПО КОТОРЫМ НЕ ВЫПОЛНЕН ГОДОВОЙ '+
      'ПЛАН ПРОИЗВОДСТВА';
St[3]:='-----------------------------------------'+
      '------------';
St[4]:='|N п/п|   Код      |     Наименование изделия    '+
      '|выполнение|';
St[5]:='|       | изделия |                              '+
      '| плана, % |';
St[6]:='-----------------------------------------'+
      '------------';
PrintHat(6);
j:=6; m:=0;
For i:=1 to np do
 Begin
   ProcGod:=100*(Products[i].Fact[1]+Products[i].Fact[2])/
            (Products[i].Plan[1]+Products[i].Plan[2]);
   If ProcGod<100 then
     Begin
       St[1]:=''; Inc(m);
       Kod:=Products[i].Kod;
```

```
          k:=SearchKodif(Kod);
          If k>0 then
            St[1]:=Kodifs[k].Name
          Else
            St[1]:='                                    ';
          Str(m:2,St[2]); Str(Kod:6,St[3]);
          Str(ProcGod:8:2,St[4]); St[2]:='|  '+St[2]+' | '+
              St[3]+'  | '+St[1]+'|'+ St[4]+' |';
          WritelnString(St[2]);
          CheckPageScreen(j);
        End;
    End;
  St[6]:='-----------------------------------------------'+
         '-------------';
  WritelnString(St[6]);
  Writeln('Обработку архива закончено');
  PrintKeyAndWaitEnter;
End { WorkUpArchive };
{ ------------------------------------------------------ }

Begin
  Assign(FileInput,'Input.txt');
  Assign(FileAdd,'Add.txt');
  Assign(FileKodif,'Kodif.txt');
  Assign(FileRes,'Res.txt');
  Assign(FileOut,'Out.dat');

  ClrScr;
  UsesDevice;
  If Device=1 then
    Rewrite(FileRes);
  SignArchive:=false;
  Repeat
    ClrScr;
    PrintString(15, 8,'Укажите режим работы программы:');
    PrintString(17, 9,'0 - конец работы;');
    PrintString(17,10,'1 - создание архива изделий;');
    PrintString(17,11,'2 - сортировка компонентов архива;');
    PrintString(17,12,'3 - печать архива изделий;');
    PrintString(17,13,'4 - печать кодификатора изделий;');
    PrintString(17,14,'5 - добавление компонентов в архив;');
    PrintString(17,15,'6 - удаление компонента из архива;');
    PrintString(17,16,'7 - изменение компонента в архиве;');
    PrintString(17,17,'8 - обработка архива');
    Writeln;
    KeyRegime:=Round(GetNumber(0,8,1,0,1,0));
    Case KeyRegime of
        0 :              ;
        1 : CreateArchive;
        2 : SortArchive  ;
        3 : PrintArchive ;
        4 : PrintKodif   ;
        5 : AddArchive   ;
        6 : DeleteArchive;
        7 : ChangeArchive;
```

```
        8 : WorkUpArchive
      Else KeyRegime:=0;
    end;
  Until KeyRegime=0;
  If Device=1 then
    Close(FileRes);
End.
```

Результаты работы программы Labor7

(таблицы приводятся в частичном виде)

1. Печать архива после отработки режима 1 (архив не сгруппирован).
(см. на следующей странице)

АРХИВ ВЕДОМОСТЕЙ О ВЫПУСКАЕМОЙ ПРОДУКЦИИ

N п/п	N цеха	Код изделия	Един. изм.	Цена	План выпуска		Факт.выпуск	
					1	2	1	2
1.	1	320011	шт.	221.50	200	180	180	260
2.	1	324531	шт.	35.14	500	600	510	610
3.	5	621412	шт.	53.14	440	460	442	434
4.	5	623462	шт.	35.30	3200	2900	3060	3070
5.	14	386410	шт.	15.60	400	600	400	410
6.	3	410410	шт.	8.20	2500	2600	2100	2200
7.	3	410411	шт.	9.60	1100	1200	1170	1190
8.	12	529270	шт.	445.20	500	700	510	710
9.	12	521570	шт.	5.80	5000	8000	6600	7500
10.	12	530510	шт.	19.80	8200	8200	8050	8130
.....								
25.	5	620405	шт.	890.00	160	170	162	155
26.	5	611411	шт.	876.90	110	120	123	125
27.	5	601515	шт.	41.40	560	600	578	588
28.	5	622277	шт.	28.60	1800	2000	1810	1856

3. Печать кодификатора изделий (режим 4).

КОДИФИКАТОР ВЫПУСКАЕМОЙ ПРОДУКЦИИ

N п/п	Код изделия	Наименование изделия
1.	212233	Седловина
2.	320011	Кронштейн
3.	324531	Поддон цилиндра
4.	337288	Ресивер
5.	338666	Кольцо обжимное
6.	385005	Корпус двигателя
7.	385571	Станина
8.	385629	Корпус насоса
9.	386388	Маховик
10.	386410	Заготовка вала
.....		

```
:  36.       623462         Стакан конусный                          :
:  37.       623466         Стакан цилиндрический                    :
:  38.       759003         Прокладка                                :
:  39.       769344         Пластина Ф1305                           :
:  40.       780088         Заполнитель С154а                        :
:  41.       799444         Ферропласт                               :
:  42.       899444         Втулка коническая                        :
-------------------------------------------------------------------
```

4. Печать архива после отработки режимов 5 и 2 (архив сгруппирован).

```
Содержимое файла FileAdd:
05  212233  шт.    38.45    2800   2500   2810   2356
14  899444  кг     55.15     650    700    660    720
03  456643  шт.   435.00     180    180    182    166
             АРХИВ ВЕДОМОСТЕЙ О ВЫПУСКАЕМОЙ ПРОДУКЦИИ
-------------------------------------------------------------------
: N  : N  :  Код  :Един.:       :  План выпуска :  Факт.выпуск  :
:п/п :цеха:изделия:изм. : Цена  ---------------------------------
|    |    |       |     |       |   1   |   2   |   1   |   2   |
-------------------------------------------------------------------
|  1.   5   212233   шт.   38.45   2800   2500    2810    2356  :
:  2.   1   320011   шт.  221.50    200    180     180     260  :
:  3.   1   324531   шт.   35.14    500    600     510     610  :
:  4.   1   337288   шт.  665.70    200    250     220     222  :
:  5.   1   338666   шт.   12.10   1900   2500    1990    2110  :
:  6.  14   385005   шт.  280.75    110    120      95      98  :
:  7.  14   385571   шт.  444.40    150    160     155     159  :
:  8.  14   385629   шт.  188.25    250    250     260     252  :
:  9.  14   386388   шт.   82.14    560    700     577     655  :
:  .....................................................  :
: 25.   5   621412   шт.   53.14    440    460     442      434 :
: 26.   5   622277   шт.   28.60   1800   2000    1810    1856 :
: 27.   5   623462   шт.   35.30   3200   2900    3060    3070 :
: 28.  14   759003   кг    54.70    660    600     630     650  :
: 29.  14   769344   кг    88.80    440    450     460     470  :
: 30.  14   780088   кг    95.50    740    750     730     760  :
: 31.  14   899444   кг    55.15    650    700     660     720  :
-------------------------------------------------------------------
```

5. Таблицы обработки архива (режим 8).

```
                СВЕДЕНИЯ О ВЫПОЛНЕНИИ ПЛАНА ПО ЦЕХУ 14
-------------------------------------------------------------------
:                    :         Полугодие           :     За      :
:                    ------------------------------     год     :
:                    :       1:            2       :             :
-------------------------------------------------------------------
: План выпуска       :   342687.40  :  363119.50  :   705806.90 :
: Фактический выпуск :   343157.03  :  355680.80  :   698837.83 :
: Процент выполнения :       100.14 :      97.95  :      99.01  :
-------------------------------------------------------------------

 СПИСОК ИЗДЕЛИЙ, ПО КОТОРЫМ НЕ ВЫПОЛНЕН ГОДОВОЙ ПЛАН ПРОИЗВОДСТВА
-------------------------------------------------------------------
```

:N п/п:	Код	Наименование изделия	:выполнение:
:	изделия :		: плана, % :
: 1 :	337288	: Ресивер	: 98.22 :
: 2 :	338666	: Кольцо обжимное	: 93.18 :
: 3 :	385005	: Корпус двигателя	: 83.91 :
: 4 :	386388	: Маховик	: 97.78 :
: 5 :	386410	: Заготовка вала	: 81.00 :
: 6 :	410410	: Резец проходной	: 84.31 :
: 7 :	521682	: Пускатель магнитный	: 88.18 :
: 8 :	521695	: Реле тепловое	: 96.22 :
: 9 :	530510	: Трансформатор тока	: 98.66 :
: 10 :	620405	: Штамп пресс-формы	: 96.06 :
: 11 :	621412	: Панель монтажная	: 97.33 :

Варианты заданий

В каждой программе, которая реализует соответствующий вариант задачи по лабораторной работе №7, обязательными режимами работы являются:

- создание типизированного файла, содержащего обрабатываемые записи архива;
- группировка архива по какому-либо признаку;
- вывод архива на экран и, по указанию пользователя, на принтер или магнитный диск.

Списки, которые формируются, также должны быть сгруппированы по определенному признаку (по возрастанию шифра или кода, по алфавиту фамилий и т.п.).

Обязательными операциями при коррекции архива являются дополнение архива новыми записями и удаление из архива существующих записей. Реквизиты, которые изменяются, указываются в условии задачи.

Количество обрабатываемых записей, как правило, должно быть не менее 20.

Имена переменных в программе должны быть содержательными.

Проценты и средние значения каких-либо параметров обязательно печатать с одной или двумя цифрами в дробной части числа (в зависимости от условия задачи).

---

1. В расписании движения самолетов из аэропорта г.Донецка указаны следующие сведения: номер рейса (4 цифры); аэропорт назначения; расстояние в км; стоимость билета (взрослый и детский); время в часах и минутах (отправление и прибытие). Сформировать таблицу рейсов для заданного аэропорта назначения, а также таблицу сведений о трех рейсах с максимальной продолжительностью полета. Корректируемые реквизиты: стоимость билета, время отправления и прибытия.

---

2. В каталоге студий звукозаписи имеются следующие данные: название группы, код (две буквы и три цифры), название альбома, год выпуска альбома, название студии. Необходимо сформировать таблицу групп, которые выпустили альбом в заданном году. Определить две группы, которые выпустили наибольшее количество альбомов. При удалении альбома из архива или при изменении его реквизитов для идентификации альбома использовать его код. Корректируемые реквизиты: название и год выпуска альбома.

---

3. В журнале успеваемости академической группы по программированию имеются следующие данные: фамилия и инициалы студента, номер зачетной книжки в виде 09/27710, где 09 - год поступления в вуз, оценки по пяти лабораторным работам, количество пропусков занятий. Сформировать список студентов, которые не сдали какую-нибудь работу (номер ее задается). Определить трех студентов, которые имеют наибольшее количество пропусков

(студентов, которые сдали пять лабораторных работ, не учитывать). Корректируемые реквизиты: оценки по лабораторным работам, количество пропусков занятий.

---

4. В магазине имеются следующие записи о товарах: наименование, единица измерения, артикул (10 цифр), цена (грн и коп), количество прибывшего, проданного и товара, который остался за текущие сутки. Составить таблицу проданного товара, сгруппировав ее по уменьшению общей стоимости продажи, и таблицу пяти товаров, наибольших по количеству остатка. Корректируемые реквизиты: цена и количества.

---

5. На заводе радиоэлектроники выпускают звуковоспроизводящую технику и имеются следующие данные: название прибора (например, Рекорд-601м), назначение (магнитофон, проигрыватель и т.п.), код (2 буквы и 3 цифры), год создания, стоимость, гарантийный срок эксплуатации. Необходимо составить список магнитофонов, разработанных в заданном году, а также список приборов, гарантийный срок которых больше трех лет. Корректируемые реквизиты: стоимость, срок гарантии.

---

6. В каталоге программного обеспечения имеются следующие данные: имя и расширение имени файла, тип файла (текстовый, типизированный, нетипизированный), атрибут файла (каталог, обычный, скрытый, системный), емкость файла, дата и время создания. Составить список текстовых файлов, а также таблицу файлов с емкостью свыше 64 Кбайт, сгруппировав ее по возрастанию даты и времени создания файла. Корректируемые реквизиты: емкость, дата и время создания файла.

---

7. При проведении итогов референдума в г.Донецке от разных районов была получена следующая информация: название района, код района (буква и две цифры), общее число жителей, которые имеют право голоса, число жителей, которые принимали участие в референдуме, число жителей, голосовавших "Да" и голосовавших "Нет". Необходимо составить список районов, большинством голосов проголосовавших "Да", и список районов, в которых голосовало менее 50% жителей, имеющих право голоса. Корректируемые реквизиты: данные о числе жителей (4 графы).

---

8. В областной больнице ведется учет больных по следующим данным: номер медицинской карты (две буквы и три цифры), фамилия и инициалы больного, местожительство (город или район области), номер палаты, дата прибытия (год, месяц, день), диагноз. Необходимо вывести список больных, находящихся в заданной палате, а также список больных, находящихся в больнице свыше 14 дней. Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд - день. Корректируемые реквизиты: номер палаты, диагноз.

---

9. На станции технического обслуживания автомобилей ведется учет автомобилей, которые прошли капитальный ремонт, по таким данным: марка машины, регистрационный номер (две буквы и пять цифр), пробег (в км) после предшествующего ремонта, год выпуска машины. Необходимо составить список машин, которые имеют пробег более 100 000 км, а также список пяти самых новых автомобилей, которые прошли ремонт. Корректируемые реквизиты: пробег, год выпуска.

---

10. В библиотеке имеются следующие данные об вузовских учебниках: код (три цифры, точка и две цифры в скобках (например, 681.(07)), название, фамилия и инициалы автора, год издания, цена, нормативное количество экземпляров книги, фактическое количество. Необходимо вывести список книг, изданных в заданном году, и список книг, которые имеются в библиотеке в одном экземпляре. Корректируемые реквизиты: год издания, количество экземпляров. При формировании исходных данных считать, что для каждой книги задан лишь один автор. Название книги можно указывать в сокращенном виде.

---

11. В районном военкомате ведется учет юношей допризывного и призывного возраста. Имеются следующие данные: фамилия и инициалы, дата рождения, номер личного дела (буква и четыре цифры), адрес (улица, дом, квартира), пригодность к службе ("Да" или "Нет"). Необходимо вывести список юношей, которые призываются на службу в августе текущего года (по достижению 18 лет). Корректируемые реквизиты: номер личного дела, пригодность к службе.

---

12. В аптеке ведется учет лечебных средств. Имеются следующие данные: название лекарства, серия (7 цифр), цена упаковки, количество упаковок в аптеке, год и месяц выпуска, срок хранения (в годах и месяцах). Необходимо вывести список лекарств, не пригодных к употреблению на заданный год и месяц, и пять самых дорогих лекарств. Корректируемые реквизиты: цена, количество, срок хранения.

---

13. В заводском цехе ведется журнал расхода материалов по таким данным: наименование материала, шифр (6 цифр), единица измерения (шт., кг и т.п.), расход в сутки, количество, имеющееся в цехе. Необходимо вывести список материалов, которые закончатся через заданное количество дней, а также список пяти наименее расходуемых материалов. Корректируемые реквизиты: расход, количество.

---

14. За материально ответственным лицом кафедры числятся материальные ценности, записанные в журнале: наименование предмета, инвентарный номер (4 цифры), номер лаборатории, год и месяц приобретения, стоимость (грн.), срок службы (годы). Необходимо вывести список предметов, которые подлежат списанию в заданный год, а также список пяти самых долгосрочных предметов. Если наименование предмета состоит из двух слов, то объединять эти слова с помощью знака подчеркивания. Корректируемые реквизиты: количество, стоимость, срок службы.

---

15. В заводском цехе ведется учет электроэнергии, которая расходуется машинами и приборами. Имеются следующие данные: название машины или прибора, инвентарный номер (буква и четыре цифры), потребляемая мощность, год выпуска, стоимость (грн.). Вывести список десяти наиболее энергоемких приборов, а также суммарную мощность, потребляемую цехом при всех включенных приборах. Принять во внимание, что название машины (прибора) может состоять из одного или двух слов. В последнем случае эти слова объединяются с помощью знака подчеркивания. Корректируемые реквизиты: мощность, стоимость.

---

16. На АТС ведется учет междугородных разговоров абонентов по таким данным: регистрационный код (10 цифр), фамилия и инициалы абонента, домашний адрес (улица, дом, квартира), номер телефона, количество и сумма междугородных телефонных разговоров за месяц (в грн и коп), отметка об оплате услуг АТС («Да», «Нет»). Необходимо вывести список 10-ти абонентов, которые наиболее широко используют междугородную сеть (по суммарной стоимости разговоров), а также в алфавитном порядке список абонентов, не оплативших услуги АТС. Корректируемые реквизиты: количество, сумма.

---

17. В заводском цехе ведется учет рабочего времени. В журнал заносятся такие данные: фамилия и инициалы рабочего, должность, табельный номер (5 цифр), общее количество рабочих часов в неделю, количество отработанных часов, количество часов, пропущенных по болезни, количество часов прогулов. Необходимо вывести список 10-ти рабочих с наибольшим количеством прогулов, а также посчитать процент использования рабочего времени в среднем по цеху. Корректируемые реквизиты: количество отработанных и пропущенных часов.

---

18. Диспетчер автовокзала отмечает автобусы, прибывшие на вокзал и ушедшие с вокзала. В журнал заносятся следующие данные: номер рейса, номер автобуса (две буквы и пять цифр), фамилия и инициалы водителя, время отправления (часы, минуты), время прибытия (часы, минуты). Составить список автобусов, которые находятся в дороге в заданное

время. Это время ввести с клавиатуры как 4-значное число hhmm, где hh – часы, mm – минуты. Определить также три автобуса с наибольшими промежутками времени между отправлением и прибытием (часы и минуты). Корректируемые реквизиты: время отправления и прибытия.

---

19. На кафедре для преподавателей имеется расписание занятий на каждый день недели. Оно составлено по таким данным: фамилия и инициалы преподавателя, код преподавателя в журнале учета (3 цифры), день недели, номера аудиторий на 1-й, 2-й, 3-й, 4-й и 5-ой парах соответственно. Необходимо вывести список преподавателей, работающих в указанный день в заданной аудитории, а также общее число часов, которые отрабатываются преподавателями кафедры в этот день. Корректируемые реквизиты: номера аудиторий по дням недели.

---

20. На метеостанции ведется учет погодных условий в течение месяца. В журнал ежедневно заносятся следующие данные: число, температура воздуха днем и ночью, давление воздуха, процент содержания кислорода, радиоактивный фон, продолжительность светового дня. Необходимо вывести список погодных условий тех дней, в которые радиоактивный фон превышает заданную величину, а также определить среднюю дневную и ночную температуры за месяц. Корректируемые реквизиты: все, кроме числа месяца.

---

21. В книге заявок жилищного ремонтного управления содержатся сведения о заявках: код регистрации заявки (буква и три цифры), фамилия и инициалы заявителя, его адрес (улица, дом, квартира), тип ремонта (малый, средний, большой), даты заявки и планового срока выполнения ремонта, фактический срок выполнения ремонта (начало и конец). В плановых и фактических сроках указывать год, месяц, день. Напечатать сведения о заявителях, у которых ремонт выполнен с опозданием в три и больше месяцев, а также сведения о трех первоочередниках на малый ремонт. Корректируемые реквизиты: даты.

---

22. В часовой мастерской имеются сведения о проведении ремонтов в текущем году: код регистрации заявки (буква и три цифры), фамилия и инициалы заказчика, его адрес (улица, дом, квартира), марка часов, стоимость ремонта, дата поступления в ремонт и плановый срок выполнения (количество дней, от 1 до 40), дата окончания ремонта. В датах отмечать год, месяц, день. Выполнить анализ и отпечатать сведения о задержках выполнения ремонта в заданном месяце. Определить максимальную и среднюю задержки. Корректируемые реквизиты: даты.

---

23. В справочном бюро содержатся следующие сведения: шифр записи (2 буквы и 3 цифры), фамилия и инициалы, дата рождения, адрес (район, улица, дом, квартира), домашний телефон (при его отсутствии записывать символ “тире”). Напечатать сведения о гражданах с заданной фамилией и гражданах, старших заданного количества лет. Определить трех самых старых жителей города. Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд - день. Корректируемые реквизиты: адрес и телефон.

---

24. В жилищном управлении имеются следующие сведения о жителях: шифр записи (2 буквы и 3 цифры), адрес (улица, дом, квартира), фамилия и инициалы квартиросъемщика, количество проживающих, жилая площадь квартиры, дата переезда по данному адресу. Напечатать сведения о квартиросъемщиках, у которых имеется лишняя площадь (норма на человека 13 кв.м). Определить трех квартиросъемщиков, которые занимают минимальную площадь из расчета на одного человека. Корректируемые реквизиты: количество проживающих и дата переезда.

---

25. В личных карточках рабочих предприятия отображены такие сведения: табельный номер (5 цифр), номер цеха, фамилия и инициалы, дата рождения, профессия, год приема на работу, разряд. Вычислить для заданного цеха общее количество рабочих и количество рабочих, которые имеют соответственно разряды 3, 4, 5, 6 (абсолютные и относительные значения в % к общему количеству рабочих). Определить номер цеха, который имеет в среднем

самый молодой состав рабочих. Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд - день. Корректируемые реквизиты: номер цеха, табельный номер и разряд.

---

26. Данные о студентах вуза содержат следующие сведения: фамилия и инициалы, дата рождения, номер студбилета в виде 09/35411, где 09 - это год поступления в вуз, сокращенное название факультета, группа, шифр (3 цифры) и название дисциплины, оценки экзаменационной сессии (4 предмета). Для заданной дисциплины определить количество оценок 5, 4, 3 и 2 (абсолютные значения и %). Напечатать список отличников. Для задания дисциплины использовать кодификатор. Корректируемые реквизиты: оценки.

---

27. В инвентарной книге библиотеки для каждой книги указаны следующие данные: инвентарный номер (7 цифр), фамилия и инициалы автора, название, год издания, место издания (город), издательство, цена (грн, коп.). Вычислить, сколько и на какую сумму книг каждого издательства имеется в библиотеке. Напечатать список книг издательства "Мир". Корректируемые реквизиты: инвентарный номер, цена. При формировании входных данных считать, что для каждой книги задан только один автор. Название книги можно указывать в сокращенном виде.

---

28. В данных табельного учета рабочих завода содержатся следующие сведения: фамилия и инициалы, табельный номер (6 цифр), год рождения, пол, стаж работы, разряд, номер цеха, заработная плата за каждый из предыдущих трех месяцев. Для каждого цеха вычислить: количество рабочих и общую среднюю зарплату для 4, 5 и 6-го разрядов. Напечатать список рабочих, которые имеют среднюю зарплату свыше 1500 грн. Корректируемые реквизиты: стаж, разряд, зарплата за последний месяц.

---

29. В месячной ведомости о рабочих с почасовой оплатой труда указываются: номер цеха, табельный номер (5 цифр), фамилия и инициалы, разряд, количество рабочих дней в месяце и общее количество отработанных часов, стоимость одного часа (грн, коп.). Вычислить для каждого цеха: количество рабочих, сумму зарплаты, среднее количество отработанных часов на один день работы. Напечатать также общие по заводу сведения о десяти рабочих с максимальной зарплатой. Корректируемые реквизиты: разряд, количество рабочих дней и отработанных часов.

---

30. По каждому из изделий, которые выпускаются заводом, имеются следующие сведения: номер цеха, код изделия (7 цифр), наименование изделия, единица измерения (шт., кг и т.п.), количество изделий, себестоимость, оптовая цена. Требуется для каждого цеха вычислить суммарные затраты Z, суммарную прибыль P и коэффициент эффективности K. Определить в целом по заводу три самых массовых изделия. Корректируемые реквизиты: количество, себестоимость, цена.

Примечание. Коэффициент эффективности вычисляется по формуле $K = (Z+P)/Z$.

---

31. В документации приемной комиссии для каждого абитуриента указаны следующие сведения: номер абитуриента (4 цифры), фамилия и инициалы, шифр специальности (4 цифры), название специальности, оценки на вступительных экзаменах (3 экзамена). Если по одному из экзаменов абитуриент получил "2", то для следующих указывается оценка "0". Для каждой специальности определить общее количество абитуриентов, не сдавших вступительные экзамены, в том числе количество абитуриентов, не сдавших соответственно первый, второй или третий экзамен. Определить специальность, на которую был максимальный конкурс. Корректируемые реквизиты: номер абитуриента и оценки.

---

32. Имеются следующие сведения о наборах данных (НД), расположенных в библиотеке на магнитном диске: шифр НД (две буквы и три цифры), фамилия и инициалы пользователя, время создания S (год, месяц, число, час, минуты), время последнего использования P (год,

месяц, число, час, минуты), общее количество *k* обращений к НД. Определить средний период обращений к НД: *F = (P-S)/k* (часы, минуты). Напечатать сведения о наборе данных с минимальным средним периодом обращений. Корректируемые реквизиты: *P* и *k*.

---

33. По результатам защиты дипломных проектов имеются следующие сведения: сокращенное название факультета, фамилия и инициалы студента, номер зачетной книжки в виде 05/25618, где 05 - это год поступления в вуз, оценка по защите, использование ЭВМ ("Да" или "Нет"). По каждому факультету напечатать сведения об использовании ЭВМ в дипломном проектировании: название факультета, количество и процент. Напечатать также список студентов, которые получили на защите оценку "5" и использовали при дипломировании ЭВМ. Корректируемые реквизиты: оценка и информация про ЭВМ.

---

34. Имеются сведения о наличии ПЭВМ на кафедрах вуза: сокращенное название кафедры, тип ПЭВМ, инвентарный номер (6 цифр), параметры (тактовая частота, емкость оперативной памяти, емкость винчестера), стоимость, использование (учебная работа, НИР), год выпуска. Напечатать список учебных ПЭВМ с тактовой частотой не менее 500 Мгц и "возрастом" до 5 лет, а также определить их общую стоимость. Указать список трех кафедр, на которых установлено наибольшее количество ПЭВМ для НИР. Корректируемые реквизиты: стоимость, количество, использование.

---

35. На почтамте имеются следующие данные о подписных периодических изданиях (газеты и журналы): номер квитанции (6 цифр), фамилия и инициалы подписчика, домашний адрес (почтовое отделение, улица, дом, квартира), индекс издания (5 цифр), название издания, номер первого месяца подписки, количество месяцев подписки, стоимость подписки. При отказе от подписки "Количество месяцев = 0". Напечатать две группы сведений:

а) по изданиям - индекс, название, количество подписчиков, сумма подписки, среднее количество месяцев подписки;

б) по почтовым отделениям - количество изданий, количество подписок, сумма подписок.

Для названия издания использовать кодификатор (индекс, название).

Корректируемые реквизиты: номер и количество месяцев подписки, стоимость подписки.

---

36. Для текущего семестра в учебной части вуза имеются сведения об изучаемых дисциплинах: шифр дисциплины (4 цифры), название дисциплины (одно или два слова), сокращенное название кафедры (не более чем 4 символа), сокращенное название специальности (не более чем 3 символа), количество часов (лекций, лабораторных занятий, практических занятий), тип отчетности (экзамен, зачет). Для каждой кафедры напечатать: название кафедры, количество специальностей, количество дисциплин, общее количество часов аудиторной нагрузки (в том числе отдельно лекций, лабораторных и практических занятий), общее количество экзаменов и зачетов. Определить две специальности с максимальным количеством дисциплин. Для задания дисциплины использовать кодификатор. Корректируемые реквизиты: количество часов и тип отчетности.

---

37. По каждой из лабораторий вуза имеются такие данные: номер лаборатории (3 цифры), название лаборатории, сокращенное название кафедры (не более чем 4 символа), тип лаборатории (учебная или научная), ее площадь, количество рабочих мест. Для заданной кафедры сформировать таблицу: название кафедры, количество лабораторий (общее, учебных, научных), их площадь, количество рабочих мест. Дать сведения о трех кафедрах, которые имеют максимальную площадь лабораторий. Корректируемые реквизиты: площадь и количество рабочих мест.

---

38. Турнирная таблица чемпионата страны по футболу содержит следующие сведения: номер команды (2 цифры), название команды, город, фамилия и инициалы капитана,

количество проведенных игр, результаты игр (количество побед, поражений, ничьих), количество забитых и пропущенных мячей. Нужно подсчитать количество очков для каждой команды и напечатать данные о командах в порядке их расположения в турнирной таблице. Определить две команды с лучшей разностью забитых и пропущенных мячей. Корректируемые реквизиты: все числа.

---

39. Даны записи о расходовании электроэнергии на заводах области. Структура записи: номер завода (4 цифры), сокращенное название завода, фамилия и инициалы директора и главного энергетика, расход электроэнергии в тыс. кВт-час (плановый и фактический). Для каждого завода подсчитать размер отклонения фактического расхода от планового (абсолютное значение и относительное в % с учетом знака отклонения). Напечатать сведения о двух заводах с максимальным относительным значением экономии электроэнергии. Корректируемые реквизиты: плановый и фактический расход электроэнергии.

---

40. Даны сведения о времени выполнения задачи на ЭВМ, работающей в мультизадачном режиме (время измеряется в минутах, секундах и миллисекундах, например 1821580, то есть 18' 21,580"). Структура записи: шифр задачи (буква и 5 цифр), код отдела (3 цифры), фамилия и инициалы программиста, общее время выполнения задачи, время центрального процессора. Сформировать таблицу: код отдела, количество задач, суммарное время выполнения задач, суммарное процессорное время, процент процессорного времени. Напечатать сведения о трех программистах, у которых суммарное время выполнения всех задач минимальное. Корректируемые реквизиты: время задачи и центрального процессора.

---

41. Даны сведения за месяц о пропусках занятий студентами групп. Структура записи: название группы, сокращенное название факультета, количество студентов в группе, общее количество часов пропусков занятий, количество часов пропусков по уважительной причине. Вычислить для каждого факультета: общее количество студентов, суммарные значения времени пропусков, среднее количество часов пропусков по неуважительной причине на одного студента. Напечатать сведения о трех группах, которые имеют максимальное количество пропусков занятий по неуважительной причине. Корректируемые реквизиты: количество студентов, количество часов пропусков.

---

42. Структура записи месячной ведомости по цеху имеет следующий вид: табельный номер (5 цифр), фамилия и инициалы работника, год рождения, пол, номер цеха, стаж работы, разряд рабочего, сумма заработной платы. Нужно для заданного цеха определить количество рабочих и среднюю заработную плату, а также эти же реквизиты по стажу работы в интервалах: до 3 лет включительно, от 4 до 6 лет, от 7 до 10 лет, свыше 10 лет. Определить и напечатать номера трех цехов, по которым средняя зарплата максимальная. Корректируемые реквизиты: стаж, разряд, зарплата.

---

43. Структура записей в сведениях по деканату имеет вид: название группы, шифр дисциплины (3 цифры), количество студентов, количество оценок ("5, "4, "3" и "2"), количество часов пропусков занятий (в отдельности лекционных и практических). Для заданной группы вычислить: средний балл, количество каждой из оценок в %, общее количество пропусков занятий, в том числе отдельно лекционных и практических. Напечатать сведения по трем лучшим группам. Корректируемые реквизиты: оценки и часы пропусков.

---

44. Дан массив записей с информацией о годовом плане работы сотрудников научно-исследовательского института. Структура записи: номер отдела, фамилия и инициалы сотрудника, табельный номер, номер темы (3 цифры), продолжительность работы по теме (в месяцах), код должности, размер зарплаты (в гривнах). Для заданного отдела напечатать: номер отдела, количество тем, количество сотрудников, общий фонд зарплаты. При этом учесть, что один и тот же сотрудник может работать одновременно в нескольких темах (суммарная продолжительность его работы не может превышать 12 мес.). Определить три темы с

максимальным количеством участников. Корректируемые реквизиты: номер темы, продолжительность работы по теме, зарплата.

---

45. Информация о продаже товаров имеет следующую структуру: номер магазина, номер секции, номер чека (буква и 4 цифры), название товара, артикул товара (6 цифр), цена товара (грн, коп.), количество товара, дата продажи. Нужно определить товарооборот (общую сумму выручки) по заданному магазину. Выбрать две секции с максимальным товарооборотом. Корректируемые реквизиты: цена и количество товара.

---

46. Для обработки на ЭВМ поступили сведения о производстве деталей за прошлую неделю (неделю считать шестидневной). Структура записи: шифр наряда (5 цифр), день недели, номер цеха, номер участка, табельный номер, код операции, разряд работы, количество изготовленных и количество принятых деталей. Для заданного цеха и заданного дня недели определить количество бракованных деталей. Определить также день недели, в который количество бракованных деталей было наибольшее. Корректируемые реквизиты: разряд и количество деталей.

---

47. Заданы записи следующей структуры: название группы, номер зачетной книжки в виде 09/34101, где 09 - год поступления в вуз, фамилия и инициалы студента, оценки по пяти экзаменам. Для заданной группы напечатать: название группы, количество студентов, которые сдали экзамены на "4" и "5", процент студентов, которые имеют хотя бы одну неудовлетворительную оценку, процент студентов, которые имеют 2 и больше неудовлетворительных оценок. По результатам сдачи экзаменов определить трех лучших студентов. Корректируемые реквизиты: оценки.

---

48. Записи, содержащие информацию о книгах в библиотеке, имеют следующую структуру: регистрационный номер книги (8 цифр), фамилия и инициалы автора, название книги, шифр тематики (3 цифры), год издания, издательство, количество экземпляров. Выполнить печать сведений о книгах заданного автора. Определить шифры тематик, по которым имеется максимальное и минимальное количество книг. Корректируемые реквизиты: шифр тематики и количество экземпляров. При формировании исходных данных считать, что для каждой книги задан лишь один автор. Название книги можно задавать в сокращенном виде.

---

49. Шахматный чемпионат проводится по круговой системе. В чемпионате принимают участие n спортсменов ($n \le 10$). Для каждой игры составляются сведения по форме: номера первого и второго спортсменов (по две цифры), их фамилии и инициалы, результат игры (1:0, 0:1, 0.5:0.5), время игры первого и второго спортсменов (часы, минуты). Нужно для каждого спортсмена напечатать: номер, фамилия и инициалы, количество побед, ничьих и поражений, количество набранных очков, среднее время одной игры. Сведения упорядочить по уменьшению набранных очков. Корректируемые реквизиты: результаты и время.

---

50. На каждого из спортсменов, заявленных на соревнование по легкой атлетике, составлена карточка со следующими данными: регистрационный номер (три цифры), город, фамилия и инициалы, возраст (годы), рост (см), коды видов соревнований (не более четырех видов). Нужно напечатать для заданного города: город, количество спортсменов, в том числе по интервалам возраста (до 18 лет, от 18 до 20 лет, от 20 до 25 лет, свыше 25 лет), средний возраст спортсменов (в дробной части числа - одна цифра), количество заявленных видов соревнований. Напечатать сведения о трех наиболее высоких спортсменах. Корректируемые реквизиты: возраст, рост, коды.

---

51. По каждому из автомобилей, имеющихся на автобазе, представлены следующие сведения: регистрационный номер (3 буквы и четыре цифры), тип, заводской номер, год выпуска, количество пройденных километров, оценка технического состояния (по четырехбальной системе). По каждому типу автомобиля определить среднюю оценку

технического состояния. Напечатать сведения о трех автомобилях с максимальным пробегом. Корректируемые реквизиты: количество километров, оценка состояния.

---

52. На АТС для каждого междугороднего разговора составлена карточка со следующими данными: код регистрации (2 буквы и 4 цифры), фамилия и инициалы абонента, телефон, домашний адрес (улица, дом, квартира), заказанный город, расстояние в км, телефон заказа, дата и время разговора (в мин.). Напечатать сведения для оплаты переговоров, считая, что по Украине стоимость одной минуты составляет: 10 коп. - до 200 км, 20 коп. - до 500 км, 30 коп. - свыше 500 км. Определить три города, для которых время переговоров максимальное. Корректируемые реквизиты: расстояние, дата, время.

---

53. Для текущей четверти в учебной части школы имеются сведения об изучаемых дисциплинах: класс, шифр (3 цифры) и название дисциплины (одно или два слова), преподаватель (фамилия и инициалы), количество часов за неделю, количество часов за четверть. Для каждого класса напечатать: шифр и название преобладающей дисциплины, количество часов за четверть по этой дисциплине. Определить две дисциплины с максимальным количеством часов. Поиск записи в архиве вести по двум реквизитам: класс и шифр дисциплины. Корректируемые реквизиты: преподаватель (фамилия и инициалы) и количество часов.

---

54. Имеются списки концертов и театральных спектаклей: код регистрации (буква и 3 цифры), театр, вид представления (концерт, спектакль), дата (число и месяц), название, начало представления (час), стоимость билетов (партер, амфитеатр, бельэтаж). Сформировать репертуар театра им. Артема на март месяц. Составить список концертов со стоимостью билетов в амфитеатр не более 20 грн. Корректируемые реквизиты: вид спектакля, название, стоимость билетов.

---

55. В фирменном магазине радиоэлектроники ведется учет продажи телевизоров: дата продажи (год, месяц, число), номер кассового чека, тип телевизора, стоимость, страна-изготовитель, дата рекламации покупателя, отметка об обмене телевизора или стоимость гарантийного ремонта (при наличии рекламации). Напечатать список телевизоров, изготовленных в Гонконге, Сингапуре и Малайзии, на которые поступили рекламации на протяжении года после их продажи, а также список трех типов телевизоров с наибольшей суммой выручки. Корректируемые реквизиты: стоимость, рекламация, обмен, гарантийный ремонт.

---

56. Имеются списки жителей дома, составленные в таком порядке: номер квартиры, фамилия и инициалы квартиросъемщика, количество членов семьи, количество детей до 16 лет, число комнат, общая площадь. Сформировать список многодетных семей (более трех детей) и список перенаселенных квартир по норме 13 кв.м/чел. Корректируемые реквизиты: количество членов семьи и количество детей.

---

57. В семейной учетной книге содержится следующая информация: дата (год, месяц, день), приход, расход, статья расходов (код), название расхода, остаток. Учесть, что бывают дни с приходом без расходов и наоборот. Составить список приходов, расходов и остатков за каждый месяц. Подвести итоги за полугодие. Определить месяц с максимальным приходом. Корректируемые реквизиты: приход, расход, код и название расхода, остаток.

---

58. На заочную школьную олимпиаду поступили решения опубликованных задач. После их проверки создана база данных с информацией об участниках: учетный номер в базе (2 буквы и 3 цифры), фамилия и инициалы, адрес (город, улица, дом, квартира), предмет (информатика, физика, математика, химия), баллы. Отпечатать данные о трех победителях по каждому из предметов. Определить, какой город взял наиболее активное участие в олимпиаде. Корректируемые реквизиты: баллы.

---

59. В студенческой поликлинике регистратура ведет учет посещаемости врачей: учетный номер регистрации (2 буквы и 4 цифры), дата (год, месяц, день), фамилия и инициалы студента, факультет, группа, специализация врача, время посещения (часы, минуты). Для заданной даты составить список посещений по специализациям врачей. Определить, на каком факультете наиболее низкая заболеваемость. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: дата, специализация, время.

---

60. Имеется информация о ходе сессии по факультету: группа, дисциплина, количество студентов, сдавших экзамен на "5", "4", "3" и "2" (недопуск к экзамену и неявку на экзамен считать эквивалентным оценке "2"). Сформировать список из трех групп, в которых относительное количество студентов (в %), не сдавших сессию, максимально. Определить по среднему баллу лучшую и худшую группы. Корректируемые реквизиты: количества студентов.

---

61. В бухгалтерии предприятия ведется учет зарплаты по форме: год, месяц, отдел (две или три буквы), фамилия и инициалы работника, зарплата, премия, сумма к выдаче (принять налог 20 % от зарплаты + премии). Составить по каждому отделу список рабочих, которым выплачена зарплата, превышающая среднюю по отделу. Определить три отдела с общей максимальной выплаченной суммой. Корректируемые реквизиты: зарплата, премия.

---

62. На железнодорожном вокзале имеется информация о проданных билетах: номер поезда, станции отправления и назначения, время прибытия и отправления (часы, минуты). Определить список поездов, которые прибывают в Донецк с 8 до 12 часов, а также список поездов, для которых время между отправлением и прибытием составляет больше 5 часов. Корректируемые реквизиты: время прибытия и отправления.

Примечание. Предусмотреть случаи, когда отправление и прибытие поезда происходят не в один день. В любом случае время между этими событиями не должно превышать 24 часа. Пример: отправление в 16.25, прибытие в 02.15.

---

63. Врач-гомеопат ведет учет посещений пациентов: учетный номер регистрации (2 буквы и 3 цифры), дата (год, месяц, день), фамилия и инициалы больного, назначенное лекарство, обращение к врачу (первичное, плановое, внеплановое), дата следующего посещения. Напечатать сведения по каждому месяцу о количестве первичных обращений к врачу. Определить месяц с максимальным количеством обращений. Корректируемые реквизиты: назначенное лекарство, обращение к врачу, дата следующего посещения.

---

64. Имеются данные об абитуриентах на специальность КС: учетный номер (4 цифры), фамилия и инициалы, адрес (город, улица, дом, квартира), оценки по трем вступительным экзаменам. Задан также проходной балл (в дробной части числа одна цифра). Сформировать список отличников, а также список абитуриентов г.Донецка, которые сдали все экзамены и прошли по конкурсу. Корректируемые реквизиты: оценки.

---

65. В прокуратуре имеются сведения о заключенных: фамилия и инициалы, номер дела (буква и 4 цифры), дата рождения, статья уголовного кодекса, срок заключения (годы и месяцы), дата заключения. Составить список заключенных, которые освобождаются менее чем через 6 месяцев, а также список пяти наиболее молодых заключенных. Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: срок, дата заключения.

---

66. Для прошедшей четверти в учебной части школы имеются сведения об изучаемых дисциплинах: учетный номер записи (2 буквы и 3 цифры), класс, название дисциплины, преподаватель (фамилия и инициалы), ученик (фамилия и имя), результаты выполнения трех

итоговых заданий (оценки). Для заданного класса сформировать три списка, в которые включить пять учеников, имеющих наиболее высокий средний балл (две цифры в дробной части числа) соответственно по информатике, математике и иностранному языку. Определить преподавателя с наиболее высоким общим средним баллом по всем итоговым занятиям. Корректируемые реквизиты: результаты проведения итоговых занятий.

---

67. Имеются сведения о кинотеатрах г.Донецка: учетный номер регистрации (2 буквы и 3 цифры), название кинотеатра, район города, фильм, времена сеансов (4 цифры в виде hhmm, где hh – часы, mm - минуты), цена билета. Напечатать список трех самых дорогих кинотеатров. Для заданного района сформировать список фильмов, которые можно посмотреть с 17.00 часов. Считать, что цена билета не зависит от места в зале. Корректируемые реквизиты: фильм, цена билета.

---

68. Имеется список театральных представлений: учетный номер записи (2 буквы и 3 цифры), театр, дата (месяц, день), название и вид представления (концерт, спектакль, опера, балет), начало представления (час), цена билетов (предусмотреть три разных цены), наличие билетов для каждой стоимости. Составить репертуар театра на заданный месяц. Определить наличие и стоимость билетов на определенное представление. Корректируемые реквизиты: наличие билетов и цены.

---

69. На железнодорожном вокзале имеется информация о наличии билетов на текущий день: номер поезда, станция назначения, время отправления (часы, минуты), наличие билетов (СВ, купейные, плацкартные, общие). Выдать информацию о поездах, на которые имеются плацкартные билеты не более чем за час до отправления поезда (задать текущее время), а также о поездах, на которые все билеты распроданы. Корректируемые реквизиты: наличие билетов.

---

70. В деканате факультета ВТИ сформированы данные о студентах-дипломниках: номер зачетной книжки в виде 09/17514, где 09 - год поступления в вуз, фамилия и инициалы, группа, средний балл по каждому из 9 семестров (в дробной части - две цифры). Составить список студентов, которые сдали на "3" не менее двух сессий, а также список студентов, у которых общий средний балл за весь период обучения не меньше, чем 4,75 (кандидаты на диплом с отличием). При этом считать, что на каждой сессии одинаковое количество экзаменов. Корректируемые реквизиты: баллы.

---

71. Частный врач-терапевт ведет учет посещений пациентов: учетный номер записи (2 буквы и 3 цифры), дата (год, месяц, день), фамилия и инициалы больного, назначенное лекарство, диагноз, дата следующего посещения. Напечатать сведения по каждому месяцу о количестве назначенных лекарств: название лекарства и сколько раз выписан. Определить три лекарства, которые используются чаще всего. Корректируемые реквизиты: назначенное лекарство, дата следующего посещения.

---

72. На складе ведется учет товаров по форме: номер корпуса, номер секции, шифр товара (три буквы и три цифры), цена, количество, минимальная норма. Составить список товаров, количество которых ниже минимальной нормы. Определить также корпус, в котором стоимость товаров наибольшая. Корректируемые реквизиты: цена, количество, минимальная норма.

---

73. Запись ведомости учета товаров имеет вид: номер склада, шифр товара (3 буквы и 3 цифры), цена, количество, дата изготовления (год, месяц, день), допустимый срок хранения (годы и месяцы). Для заданной даты составить список товаров (по складам), для которых срок хранения заканчивается менее чем через три месяца. Сформировать также сведения о пяти наиболее дешевых товарах. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: цена, количество, срок хранения.

---

74. Имеются данные о животных на молочной ферме: учетный номер (2 буквы и 3 цифры), кличка и порода коровы, среднесуточный надой, возраст, фамилия и инициалы доярки. Составить таблицу надоя молока по породам: общий надой и средний на одну корову. Определить трех доярок, которые добились наилучшего результата по надоям. Корректируемые реквизиты: порода, надой, возраст.

---

75. В ЖЭУ имеется информация о жителях: учетный номер записи (2 буквы и 4 цифры), адрес (улица, дом, квартира), количество комнат, метраж, фамилия и инициалы квартиросъемщика, численность семьи. Напечатать данные о перенаселенных квартирах при норме 13 кв.м/чел., а также список пяти наименее заселенных квартир. Корректируемые реквизиты: метраж, численность семьи.

---

76. В обменном бюро имеется информация о квартирах: учетный номер (2 буквы и 4 цифры), адрес (район города, улица, дом, квартира), количество комнат (общее и смежных), площадь, телефон. Сформировать список квартир, которые имеют не более двух смежных комнат. Определить для каждого района количество квартир (общее и в %), имеющих телефоны. Корректируемые реквизиты: количество комнат, телефон.

---

77. По сети кабельного телевидения имеется следующая информация: учетный номер (2 буквы и 3 цифры), улица, дом, квартира, фамилия и инициалы квартиросъемщика, тип телевизора, стоимость обслуживания в месяц, оплата за каждый из предшествующих 6 месяцев ("Да", "Нет"). Сформировать таблицу полученной арендной платы по каждому дому за 6 мес., а также список должников за 2 и больше месяцев. Корректируемые реквизиты: стоимость обслуживания, данные об оплате.

---

78. Форма учета переговоров на МТС: учетный номер (2 буквы и 4 цифры), дата (год, месяц, день), номер телефона, адрес абонента (улица, дом, квартира), заказанный город, тариф (коп/мин), продолжительность (мин), оплата ("Да", "Нет"). Для заданной даты сформировать список неплательщиков за срок свыше одного месяца. Подсчитать общую сумму неплатежей. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: тариф, продолжительность, оплата.

---

79. Имеются данные об абитуриентах на факультет ВТИ: учетный номер (4 цифры), фамилия и инициалы, возраст, год окончания школы, служба в армии ("Да" или "Нет"), оценки по трем вступительным экзаменам. Составить список абитуриентов, которые поступают в год окончания школы, а также список пяти абитуриентов, которые отслужили в армии и имеют наиболее высокий балл по экзаменам. Значение текущего года ввести с клавиатуры. Корректируемые реквизиты: оценки и сведения о службе в армии.

---

80. В ЖЭУ ведется учет внесения квартплаты: учетный номер записи (2 буквы и 3 цифры), улица, дом, квартира, общая площадь квартиры, фамилия и инициалы квартиросъемщика, тариф за 1 кв.м, сведения об оплате за каждый из предшествующих 6 месяцев ("Да","Нет"). Составить таблицу жителей, которые полностью внесли квартплату, а также список пяти жителей-должников в порядке уменьшения задолженности. Определить общую сумму полученной квартплаты. Корректируемые реквизиты: тариф, сведения об оплате.

---

81. В записной книжке имеются следующие сведения: учетный номер записи (2 буквы и 3 цифры), фамилия и инициалы, адрес (город, улица, дом, квартира), телефон, дата рождения. Напечатать список именинников, чей день рождения наступит не позже чем через месяц после заданной даты, а также трех наиболее молодых именинников, которые не имеют телефона. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: телефон, дата рождения.

---

82. В отделе сбыта завода имеется следующая информация: сокращенное название заказчика, шифр товара, название товара, цена, количество, плановый и фактический сроки поставки (год, месяц, день). Сформировать список заказов, по которым задержка поставки превышает 10 дней, а также список заказчиков с указанием общей суммы заказа. Корректируемые реквизиты: цена, количество, сроки поставки.

---

83. По заводу имеются данные о производстве за год: номер бригады, код изделия, себестоимость, общее количество, количество брака. Одно изделие могут вырабатывать разные бригады (но в полном объеме, то есть без кооперации). Составить обобщенные списки для бригад и для изделий. Определить бригаду, которая имеет наименьшее относительное количество брака (в стоимостном выражении). Корректируемые реквизиты: себестоимость и количества изделий.

---

84. На АТС имеются заявки на установку телефона: учетный номер заявки (2 буквы и 3 цифры), адрес (улица, дом, квартира), фамилия и инициалы заявителя, дата заявки (год, месяц, день), сведения о каблировании дома ("Да","Нет"). Составить список клиентов, которые подали заявки в последние 30 дней по отношению к заданной дате. Напечатать сведения о пяти первоочередниках, проживающих в каблированных домах. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: дата заявки, сведения о каблировании.

---

85. В почтовом отделении имеются сведения о подписчиках: учетный номер регистрации (буква и 5 цифр), адрес (улица, дом, квартира), фамилия и инициалы, индекс и название издания (одно или два слова), цена за месяц, начало и конец подписки в месяцах. Напечатать список подписчиков газеты "Вечерний Донецк" на срок не менее 6 месяцев, а также таблицу изданий с указанием количества и суммы подписки. Корректируемые реквизиты: цена, срок подписки. Для названия издания использовать кодификатор.

---

86. В расписании занятий студенческой группы указано: учетный номер регистрации (2 буквы и 3 цифры), день недели, номер пары (I, II, III, IV, V), вид занятия (лекция, лабораторные или практические), номер корпуса и аудитории, фамилия и инициалы преподавателя. Составить таблицу с указанием по дням количества часов занятий (общее, лекций, лабораторных и практических занятий), а также список дней, в которых есть "окна". Корректируемые реквизиты: номера пары, корпуса, аудитории.

---

87. В плане нагрузки кафедры на семестр указано: сокращенное название дисциплины, название специальности, курс, количество групп в потоке, шифр записи (2 буквы и 3 цифры), количество часов за неделю (лекций, лабораторных, практических). Составить таблицу нагрузки кафедры по III курсу специальности КС с указанием часов за семестр по дисциплинам (с учетом количества групп и разделения на подгруппы), а также список трех дисциплин с максимальным общим объемом часов. Корректируемые реквизиты: количество групп и часов.

---

88. В таблице имеются сведения о выпускаемых цифровых платах: название (3 буквы и 4 цифры), длина и ширина в мм, количество микросхем, цена. Напечатать список плат с площадью меньше, чем 160 кв.см, а также сведения о двух платах с максимальной плотностью компоновки микросхем. Корректируемые реквизиты: размеры платы, количество микросхем.

---

89. Информация о продаже товаров в магазине имеет следующий вид: номер секции, название товара, артикул (7 цифр), цена, количество проданных единиц, дата продажи (год, месяц, день). Напечатать общую таблицу проданных товаров, а также список трех товаров, общий объем продажи которых за 7 дней перед заданной датой максимальный. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: цена, количество, дата продажи.

---

90. Каталог микропроцессоров для систем автоматики содержит такие сведения: тип процессора (2 буквы и 4 цифры), сокращенное название завода-изготовителя, параметры (тактовая частота, объем ОЗУ, объем ПЗУ, разрядность), стоимость. Напечатать список микропроцессоров с объемом ОЗУ не менее 64 Мбайт и частотой не менее 266 Мгц, а также список трех наиболее дорогих микропроцессоров. Корректируемые реквизиты: параметры, стоимость.

---

91. Результаты участия игроков футбольной команды в матчах чемпионата заданы в следующей таблице: учетный номер (буква и 4 цифры), фамилия и инициалы, дата рождения, дата включения в основной состав, количество сыгранных матчей, количество забитых мячей. Напечатать таблицу игроков моложе 25 лет и играющих в основном составе не менее двух лет, а также список трех наиболее результативных игроков (по отношению количество мячей/количество матчей). Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг – год, мм – месяц, дд – день. Корректируемые реквизиты: количество матчей и забитых мячей.

---

92. Таблица отборочных игр чемпионата мира по футболу содержит такую информацию: учетный номер записи (2 буквы и 3 цифры), номер отборочной группы, страна, фамилия и инициалы тренера и капитана, количество очков, количество забитых и пропущенных мячей. Напечатать сведения по командам заданной группы, а также список стран, которые заняли первое место в каждой группе. Корректируемые реквизиты: очки, количество мячей.

---

93. Годовой отчет швейной фабрики о выпуске продукции содержит такие данные: номер цеха, название изделия, артикул (7 цифр), объем выпуска плановый и фактический по товарам I, II и III сорта (в стоимостном выражении). Часть изделий выпускается только I или только I и II сортов. Напечатать таблицу изделий I сорта, по которым выполнен план их выпуска, а также список трех цехов, которые в наибольшей мере не выполнили план. Корректируемые реквизиты: объемы выпуска продукции.

---

94. Имеются следующие сведения об успеваемости студенческих групп: учетный номер записи (2 буквы и 3 цифры), факультет, группа, количество студентов в группе, средний балл сессии, количество пересдач, количество отчисленных студентов. Напечатать таблицу успеваемости по заданному факультету, а также информацию о пяти лучших группах вуза (максимальное отношение средний балл/среднее количество пересдач при отсутствии отчислений). Корректируемые реквизиты: результаты сессии.

---

95. Опросный лист референдума содержит три вопроса. Результаты референдума по городам отображены в таблице: учетный номер (2 буквы и 3 цифры), город, количество избирателей, количество проголосовавших по каждому вопросу ("За", "Против", "Воздержались"). Составить список городов, в которых количество проголосовавших избирателей превышает 50 %, а также список трех городов, в которых количество "За" по каждому вопросу референдума превышает количество "Против". Корректируемые реквизиты: количество избирателей и количество проголосовавших по каждому вопросу.

---

96. На Уимблдонском турнире имеются следующие сведения о теннисистах: учетный номер(2 буквы и 3 цифры), имя и фамилия, страна, дата рождения, рейтинг, количество выступлений на Уимблдоне. Составить список теннисистов моложе 26 лет и не менее трех раз выступавших на Уимблдоне, а также список пяти теннисистов с максимальным рейтингом. Текущую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг - год, мм – месяц, дд – день. Корректируемые реквизиты: рейтинг, количество выступлений.

---

97. В кассах предварительной продажи железнодорожных билетов имеются такие сведения: учетный номер (2 буквы и 4 цифры), номер поезда, станция назначения, дата отправления (год, месяц, день), количество мест и количество проданных билетов (мягких,

купейных, плацкартных и общих). В некоторых поездах мягкие или общие места могут отсутствовать. Для заданной даты составить список поездов, по которым количество проданных билетов превышает 90 % от их общего количества, а также сведения о трех поездах, в которых остались лишь общие места. Задаваемую дату ввести с клавиатуры как 8-значное число ггггммдд, где гггг –год, мм – месяц, дд – день. Корректируемые реквизиты: количество мест и билетов.

---

98. На кафедре ведется журнал учета рубежного контроля выполнения курсовой работы: учетный номер (2 буквы и 3 цифры), курс, группа, фамилия и инициалы студента и преподавателя, вариант задания (2 цифры), процент выполнения плановый и фактический, причина задержки при отставании от плана (уважительная или неуважительная). Напечатать список студентов, для которых отставание от графика по неуважительной причине составляет свыше 10 %, а также список трех студентов, которые в наибольшей мере опередили график выполнения курсовой работы. Корректируемые реквизиты: проценты, причина задержки.

---

99. Турнирная таблица кубка обладателей кубков по футболу содержит такие сведения: учетный номер (2 буквы и 3 цифры), страна, название команды, фамилия и инициалы тренера и капитана, количество побед, поражений и ничьих. Напечатать список трех команд-призеров, а также сведения о командах, которые имеют больше поражений, чем побед. Корректируемые реквизиты: результаты игры.

---

100. На ипподроме имеются сведения по скачкам: имя лошади, ее возраст, вес, рейтинг, количество выигрышей. Составить список лллошадей моложе 4 лет и имеющих не менее пяти выигрышей, а также сведения о трех лошадях с максимальным рейтингом. Корректируемые реквизиты: вес, рейтинг, количество выигрышей.

Лабораторная работа №8

ОБРАБОТКА СПИСКОВ

Методические указания

В программе `Labor7` обрабатываемый архив документов представлен в памяти ЭВМ в виде массива `Products`, каждый компонент которого - это запись типа `ProductType`, имеющая длину 41 байт.

С точки зрения машинной обработки использование массива для представления группы обрабатываемых записей, хранящихся в архивном файле, имеет два существенных недостатка.

1. Массив всегда имеет фиксированный размер, который не может быть изменен в процессе работы программы. Следовательно, при объявлении массива приходится ориентироваться на его максимально возможный размер, что ведет к неэкономному использованию памяти ЭВМ.

2. При изменении текущего количества компонент в массиве (добавление новых или удаление существующих) обрабатываемый массив нужно полностью или частично перемещать в памяти ЭВМ (сдвигать влево или вправо компоненты массива). Это требует определенных затрат машинного времени, которые тем больше, чем длиннее поле памяти, занимаемое одним компонентом.

Эти недостатки можно устранить, если для представления группы обрабатываемых записей использовать списковые структуры (стек, очередь и т.п.). Для списка может быть выделено ровно столько памяти, сколько нужно для размещения данной группы записей,

причем эта память может быть выделена в любой момент работы программы и освобождена, когда в этом возникнет необходимость. При изменении количества компонент в списке перемещать в памяти эти компоненты не нужно; для этого достаточно изменить значения соответствующих указателей.

Тем не менее списковые структуры имеют и некоторые недостатки по сравнению с массивами.

1. Компонент однонаправленного списка занимает на 4 байта больше памяти, чем компонент массива (за счет размещения в нем указателя).

2. Список не предоставляет возможность прямого доступа к его компонентам (например, чтение компонента по его номеру в списке). Это, в частности, исключает возможность применения метода бинарного поиска для нахождения в сгруппированном списке компонента с заданным кодом.

Сравнивая преимущества и недостатки списковых структур, можно сделать вывод, что для задач АСУ, в которых объектами обработки в основном являются записи большого размера, списки более предпочтительны по сравнению с массивами.

В лабораторной работе № 8 решаются те же задачи, что и в лабораторной работе № 7, но с использованием списков вместо архивных массивов. Ниже приводится описание программы `Labor8`, реализующей решение задачи, условие которой приведено в методических указаниях к лабораторной работе № 7.

Содержание отчета по лабораторной работе №8 такое же, как и по работе №7, но в описании программы нужно отметить, чем она отличается от программы, которая реализует лабораторную работу № 7.

Пример выполнения задания

В программе `Labor8` вместо обрабатываемого массива используется очередь, компонентами которой являются записи типа `DynProduct`. Запись `DynProduct` разделена на две части: информационную `Inf` и указатель `Next`. Информационная часть `Inf` целиком совпадает с записью типа `ProductType`. Для работы с очередью используются три глобальных указателя: `Lp`, `Rp` и `Run` типа `PointerProduct`. Указатели `Lp` и `Rp` определяют адреса левого и правого элементов очереди, указатель `Run` используется в основном для перебора компонентов очереди.

Выбор очереди вместо стека продиктован следующими соображениями. В различных режимах работы программы нужно читать записи из архивного файла, формируя при этом линейный список. Если в роли списка использовать стек, то в нем элементы будут расположены в обратном порядке по сравнению с архивом. В частности, если в архиве записи сгруппированы по возрастанию кода изделия, то в стеке они будут расположены по уменьшению этого кода. Это требует периодически выполнять операцию реверсирования стека, что не всегда удобно. В то же время очередь лишена указанного недостатка: порядок расположения ее элементов совпадает с порядком расположения записей в архивном файле.

В программе `Labor8` предусмотрены те же режимы работы, что и в программе `Labor7`. По сравнению с предыдущей программой в Labor8 внесено ряд изменений, описанных ниже.

1. Вместо массива записей `Products` используется очередь с типом указателя `PointerProduct`.
2. Удалена процедура `SearchArchive`, так как бинарный поиск в очереди не реализуется. В связи с этим изъяты также переменная `SignSort` и фрагменты программы, где эта переменная используется.
3. Добавлена процедура `DisposeProduct` для удаления очереди.
4. Изменены тексты процедур `ReadFileOut`, `WriteFileOut`, `SortArchive`, `AddArchive`, `DeleteArchive`, `ChangeArchive`, `WorkUpArchive`, что

связано с заменой массива записей на очередь записей. Суть выполненных изменений достаточно ясна из текста соответствующих процедур.

5. Все другие процедуры и функции остались без изменения. В приведенном ниже тексте программы Labor8 для этих процедур и функций указаны лишь их имена.

Текст программы Labor8 :

```
Program Labor8;
{ Создание, печать, коррекция и обработка архива сведений }
{ о продукции, которая выпускается цехами предприятия }
Uses Crt,Printer;
Const
  MaxKodif = 50;     { макс.кол-во компонент кодификатора }
  Enter = 13;        { код клавиши Enter }
  PressKey = 'Нажмите клавишу ENTER';
Type
  ProductType = record       { тип компонента архива }
    NumberShop : byte;       { номер цеха }
           Kod : longint;    { код изделия }
        Dimens : string[5]; { единица измерения }
         Price : real;       { цена изделия }
          Plan,              { план выпуска по полугодиям }
          Fact               { факт.выпуск по полугодиям }
               : array[1..2] of real;
  end;
  PointerProduct = ^DynProduct;
  DynProduct = record
     Inf : ProductType;
    Next : PointerProduct;
  end;
  KodIfType = record    { тип компонента кодификатора }
     Kod : longint;     { код изделия }
    Name : string[35]; { наименование изделия }
  end;
  KodifAr = array[1..MaxKodif] of KodifType;
  string80 = string[80];
  StringAr = array[1..10] of string80;
Var
   np : word;              { количество компонентов архива }
   nk,                     { количество компонентов кодификатора }
   KeyRegime,              { ключ выбора режима работы }
   Device : byte;          { устройство вывода результатов: }
                           { 0 - экран; 1 - экран и магн.диск; }
                           { 2 - экран и принтер }
   SignArchive : boolean; { признак создания архива }
   Reply : char;           { символ ответа на запрос программы }
   Product : ProductType; { компонент архива }
   Lp,Rp,                  { левый и правый указатели очереди }
   Run : PointerProduct;  { текущий указатель очереди записей }
   Kodif : KodIfType;      { компонент кодификатора }
   Kodifs : KodIfAr;       { массив компонентов кодификатора }
   St : StringAr;          { строки для печати таблиц }
   FileInput,              { файл входных документов }
   FileAdd,                { файл добавляемых документов }
```

```
   FileKodif,              { файл кодификатора изделий }
   FileRes : text;         { файл результатов }
   FileOut                 { архивный файл изделий}
          : file of ProductType;
{ ------------------------------------------------------ }
Procedure WaitEnter;
Procedure PrintString(X,Y:byte; S:string);
Procedure WritelnString(S:string80);
Procedure PrintKeyAndWaitEnter;
Procedure CheckPageScreen(Var j:byte);
Function  Space(S:string80; k:byte):byte;
Function  NotSpace(S:string80; k:byte):byte;
Function  FillString(S:string; len,p:byte):string;
Function  GetNumber(MinNumber,MaxNumber:real;m1,n1,m2,n2:byte):real;
Procedure UsesDevice;
Procedure PrintHat(n:byte);
Procedure ReadProduct(Var FileInput:text);
Procedure ReadKodif;
Procedure MakeKodifs;
Procedure SortKodIf;
Procedure CreateArchive;
{ ------------------------------------------------------ }
Procedure ReadFileOut;;
{ Чтение из архива и формирование очереди изделий }
Begin
  Reset(FileOut); np:=0;
  Lp:=nil; Rp:=nil;
  While not eof(FileOut) do
    Begin
      Inc(np);
      Read(FileOut,Product);
      New(Run);
      Run^.Inf:=Product;
      Run^.Next:=nil;
      If Lp=nil then
        Lp:=Run
      Else
        Rp^.Next:=Run;
      Rp:=Run;
    End;
  Close(FileOut);
End { ReadFileOut };
{ ------------------------------------------------------ }
Procedure DisposeProduct;
{ Удаление очереди изделий }
Begin
  While Lp<>nil do
    Begin
      Run:=Lp;
      Lp:=Lp^.Next;
      Dispose(Run);
    End;
End { DisposeProduct };
{ ------------------------------------------------------ }
Procedure WriteFileOut;
```

```
{ Пересылка очереди изделий в архив }
{ с последующим удалением очереди     }
  Rewrite(FileOut);
  Run:=Lp;
  While Run<>nil do
    Begin
      Product:=Run^.Inf;
      Write(FileOut,Product);
      Run:=Run^.Next
    End;
  Close(FileOut);
  DisposeProduct;
End { WriteFileOut };
{ ------------------------------------------------------ }
Function SearchKodif(Kod:longint):byte;
. . . . . .
End { SearchKodif };
{ ------------------------------------------------------ }
Procedure SortArchive;
{ Сортировка архива по возрастанию кода изделия }
Var  i,m : word;
  KodMax : longint;
    Pmax : PointerProduct;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  m:=np;
  While m>1 do
    Begin
      Run:=Lp;
      KodMax:=Run^.Inf.Kod;
      Pmax:=Run;
      For i:=2 to m do
        Begin
          Run:=Run^.Next;
          If Run^.Inf.Kod>KodMax then
            Begin
              KodMax:=Run^.Inf.Kod;
              Pmax:=Run;
            End;
        End;
      If Pmax<>Run then
        Begin
          Product:=Run^.Inf; Run^.Inf:=Pmax^.Inf;
          Pmax^.Inf:=Product
        End;
      Dec(m);
    End;
  WriteFileOut;
  Writeln('Сортировка архива закончена');
```

```
  PrintKeyAndWaitEnter;
End { SortArchive };
{ ------------------------------------------------------- }
Procedure PrintArchive;
. . . . . .
End { PrintArchive };
{ ------------------------------------------------------- }
Procedure PrintKodif;
. . . . . .
End { PrintKodif };
{ ------------------------------------------------------- }
Procedure AddArchive;
{ Добавление компонентов в архив изделий }
Label 10;
Var  Cond : boolean;
       Sr : string;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  Reset(FileAdd);
  While not eof(FileAdd) do
    Begin
      ReadProduct(FileAdd);
      Str(Product.Kod,Sr);
      Run:=Lp; Cond:=false;
      While Run<>nil do
        Begin
          If Run^.Inf.Kod=Product.Kod then
            Begin
              Cond:=true; Goto 10
            End;
          Run:=Run^.Next
        End;
      10:
      If not Cond then
        Begin
          New(Run); Inc(np);
          Run^.Inf:=Product;
          Run^.Next:=nil;
          If Lp=nil then
            Lp:=Run
          Else
            Rp^.Next:=Run;
          Rp:=Run;
        End
      Else
        WritelnString('В архиве уже есть изделие с кодом '+Sr);
    End;
  WriteFileOut;
  Close(FileAdd);
```

```
  Writeln('Дополнение архива закончено');
  PrintKeyAndWaitEnter;
End { AddArchive };
{ ------------------------------------------------------ }
Procedure DeleteArchive;
{ Удаление компонента из архива изделий }
Label 10;
Var  Kod : longint;
    Cond : boolean;
     Del : PointerProduct;
      Sr : string;
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  ClrScr;
  Writeln('Укажите код изделия удаляемого компонента');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);
  Cond:=false;
  If Kod=Lp^.Inf.Kod then
    Begin
      Cond:=true;
      Run:=Lp; Lp:=Lp^.Next;
      Dispose(Run)
    End
  Else
    Begin
      Run:=Lp;
      While Run^.Next<>nil do
        If Kod=Run^.Next^.Inf.Kod then
          Begin
            Cond:=true;
            Del:=Run^.Next;
            If Run^.Next=Rp then
              Rp:=Run;
            Run^.Next:=Del^.Next;
            Dispose(Del);
            Goto 10
          End
        Else
          Run:=Run^.Next;
    End;
  10:
  If Cond then
    Begin
      Dec(np);
      WriteFileOut;
      Writeln('Удаление компонента из архива закончено');
    End
  Else
```

```
    WritelnString('В архиве нет компонента с кодом '+Sr);
  PrintKeyAndWaitEnter;
End { DeleteArchive };
{ ------------------------------------------------------ }
Procedure ChangeArchive;
{ Изменение компонента в архиве изделий }
Label 10;
Var  Kod : longint;
    Cond : boolean;
      Sr : string;
{ ------------------------------------------------------ }
Procedure MakeComponent;
. . . . . .
End { MakeComponent };
{ ------------------------------------------------------ }
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  ReadFileOut;
  ClrScr;
  Writeln('Укажите код изделия изменяемого компонента');
  Kod:=Round(GetNumber(0,999999,1,0,6,0));
  Str(Kod,Sr); WritelnString('Kod = '+Sr);
  Run:=Lp; Cond:=false;
  While Run<>nil do
    If Kod=Run^.Inf.Kod then
      Begin
        Cond:=true;
        Product:=Run^.Inf;
        WritelnString('  Укажите такие реквизиты :');
        WritelnString(' ед.измерения цена план-1 план-2 '+
                      'факт-1 факт-2 ');
        MakeComponent;
        Run^.Inf:=Product;
        WriteFileOut;
        Writeln(' Изменение компонента в архиве закончено ');
       Goto 10
      End
    Else
      Run:=Run^.Next;
  10:
  If not Cond then
    WritelnString('В архиве нет компонента с кодом ',Kod);
  PrintKeyAndWaitEnter;
End { ChangeArchive };
{ ------------------------------------------------------ }
Procedure WorkUpArchive;
{ Обработка архива изделий }
Var  Shop,                  { номер цеха }
    j,k,m : byte;
      Kod : longint;        { код изделия }
```

```
  PlanSt1,PlanSt2,PlanGod, { суммарные плановые показатели }
  FactSt1,FactSt2,FactGod, { суммарные фактические показатели }
  ProcSt1,ProcSt2,ProcGod, { процент выполнения плана }
  BufSt : real;            { буферная переменная }
Begin
  If not SignArchive then
    Begin
      Writeln('Архивный файл не создан. Режим отменяется.');
      PrintKeyAndWaitEnter;
      Exit
    End;
  Writeln('Укажите номер цеха');
. . . . . .
  Run:=Lp; k:=0;
  While Run<>nil do
    Begin
      If Shop=Run^.Inf.NumberShop then
        Begin
          Inc(k);
          BufSt:=Run^.Inf.Plan[1]*Run^.Inf.Price;
          PlanSt1:=PlanSt1+BufSt;
          BufSt:=Run^.Inf.Plan[2]*Run^.Inf.Price;
          PlanSt2:=PlanSt2+BufSt;
          BufSt:=Run^.Inf.Fact[1]*Run^.Inf.Price;
          FactSt1:=FactSt1+BufSt;
          BufSt:=Run^.Inf.Fact[2]*Run^.Inf.Price;
          FactSt2:=FactSt2+BufSt;
        End;
      Run:=Run^.Next;
    End;
  If k=0 then
    Writeln('В архиве нет сведения о цехе ',Shop)
  Else
    Begin
      . . . . . .
      St[1]:='       СВЕДЕНИЯ О ВЫПОЛНЕНИИ ПЛАНА ПО ЦЕХУ '
              +St[1];
      . . . . . .
    End;
  . . . . . .
  St[2]:=' СПИСОК ИЗДЕЛИЙ, ПО КОТОРЫМ НЕ ВЫПОЛНЕН ГОДОВОЙ '+
         'ПЛАН ПРОИЗВОДСТВА';
  . . . . . .
   Run:=Lp; m:=0;
   While Run<>nil do
    Begin
      ProcGod:=100*(Run^.Inf.Fact[1]+Run^.Inf.Fact[2])/
               (Run^.Inf.Plan[1]+Run^.Inf.Plan[2]);
      If ProcGod<100 then
        Begin
          St[1]:=''; Inc(m);
          Kod:=Run^.Inf.Kod;
          k:=SearchKodIf(Kod);
          If k>0 then
            St[1]:=KodIfProducts[k].Name
```

```
          Else
            St[1]:='              ';
          Str(m:2,St[2]); Str(Kod:6,St[3]);
          Str(ProcGod:8:2,St[4]);
          St[2]:='|  '+St[2]+' |  '+St[3]+'  | '+St[1]+
                 '|'+ St[4]+' |';
          WritelnString(St[2]);
          CheckPageScreen(j);
        End;
      Run:=Run^.Next;
    End;
  St[6]:='-----------------------------------------------'+
         '-------------';
  WritelnString(St[6]);
  Writeln('Обработка архива закончена');
  PrintKeyAndWaitEnter;
End { WorkUpArchive };
{ ------------------------------------------------------- }
Begin
. . . . . .
End.
```

Результаты работы программы Labor8 не отличаются от результатов, полученных в программе Labor7.

Лабораторная работа №9

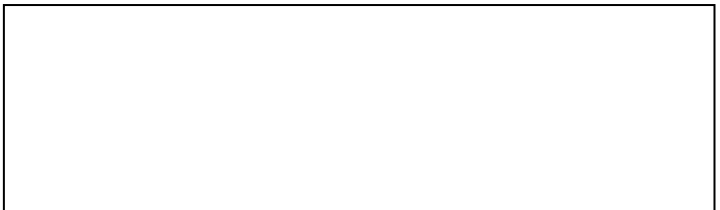
РАЗРАБОТКА МНОГОМОДУЛЬНОЙ ПРОГРАММЫ

Методические указания

Паскаль-програма не может занимать свыше 64 Кбайт оперативной памяти. Поэтому при разработке больших программ используют аппарат модулей пользователя. В этом случае программа разделяется на несколько программных модулей. Каждый модуль также не может иметь размер свыше 64 Кбайт, но их количество в программе не ограничивается. Важным свойством модуля является также то, что его можно компилировать и отлаживать автономно, независимо от других модулей. Это ускоряет процесс создания большой программы.

Модуль разделяется на три части: секция интерфейса, секция реализации и секция инициализации. Заголовок модуля состоит из зарезервированного слова **Unit** и следующего за ним идентификатора, который является именем модуля. Имя модуля должно быть уникальным. Предполагается, что имя модуля совпадает с именем файла, в котором хранится данный модуль. Например, модуль с заголовком **Unit** BasUnit должен находиться в файле с именем "BasUnit.pas". Следствием последнего обстоятельства есть то, что имя модуля, как и имя файла, должно состоять не более чем из восьми символов.

Общая структура модуля:



```
Unit UnitName;

  Interface
```

Описание видимых объектов	секция интерфейса

```
  Implementation
```

Описание скрытых объектов	секция реализации

```
  Begin
```

Операторы инициализации	секция инициализации

```
  End.
```

Интерфейсная часть начинается со служебного слова **Interface**. Если в данном модуле используются другие модули (стандартные или модули пользователя), то после слова **Interface** должна идти фраза **Uses** с именами используемых модулей. Интерфейсная часть может содержать описания констант, типов, переменных, процедур и функций. Эти описания считаются глобальными, их может использовать любой другой модуль, в фразе **Uses** которого указано имя данного модуля.

Как известно, заголовок подпрограммы содержит всю информацию, необходимую для ее вызова: имя подпрограммы, количество и типы параметров и, если это функция, тип результата. Тело подпрограммы - это блок, который определяет алгоритм ее работы. С точки зрения вызывающей программы вся необходимая и достаточная информация находится в заголовке подпрограммы, блок подпрограммы носит сугубо внутренний характер по отношению к вызываемой подпрограмме.

В связи с вышесказанным, в интерфейсной части размещают только заголовки процедур и функций, имеющие значение при их глобальном использовании. Полное описание процедур и функций переносится в секцию реализации, которая начинается со служебного слова **Implementation**.

В секции реализации могут быть свои описания констант, типов и переменных, которые используются только при работе этого модуля. Здесь могут размещаться также процедуры и функции, заголовки которых отсутствуют в секции интерфейса; эти процедуры и функции недоступны для других модулей и могут быть активизированы только при работе данного модуля.

Поскольку другие модули могут использовать только объекты, описанные в секции интерфейса, но не в секции реализации, то говорят, что секция интерфейса содержит видимые объекты, а секция реализации - скрытые объекты. Различие между секцией интерфейса и секцией реализации можно определить еще следующим образом: секция интерфейса указывает, что делает модуль, а секция реализации описывает, как он это делает.

Секция инициализации необязательна, ее наличие определяет слово **Begin**. Секции инициализации всех модулей вызываются перед запуском основного тела программы. Эту секцию обычно используют для установки начальных значений переменных данного модуля, для открытия файлов, автоопределения типа монитора и др. Секция инициализации может быть пустой, а при отсутствии слова **Begin** соответствующий модуль не содержит эту секцию.

Компиляция модуля возможна как из интегрированной среды Турбо Паскаля (`turbo.exe`), так и с помощью компилятора командной строки (`tpc.exe`). Так как модуль не

является непосредственно выполняемой единицей, то в результате его компиляции получается дисковый файл с расширением `tpu` (Turbo Pascal Unit), при этом имя этого файла копируется из имени файла с входным текстом модуля. Компиляция модуля возможна, если к началу компиляции созданы все нестандартные модули с расширением `tpu`, имена которых перечислены в фразе **`Uses`** данного модуля.

Структура многомодульной программы может быть самой различной в зависимости от назначения программы, ее размера, объединения процедур по группам и т.д. Ниже приводится один из возможных вариантов такой структуры.

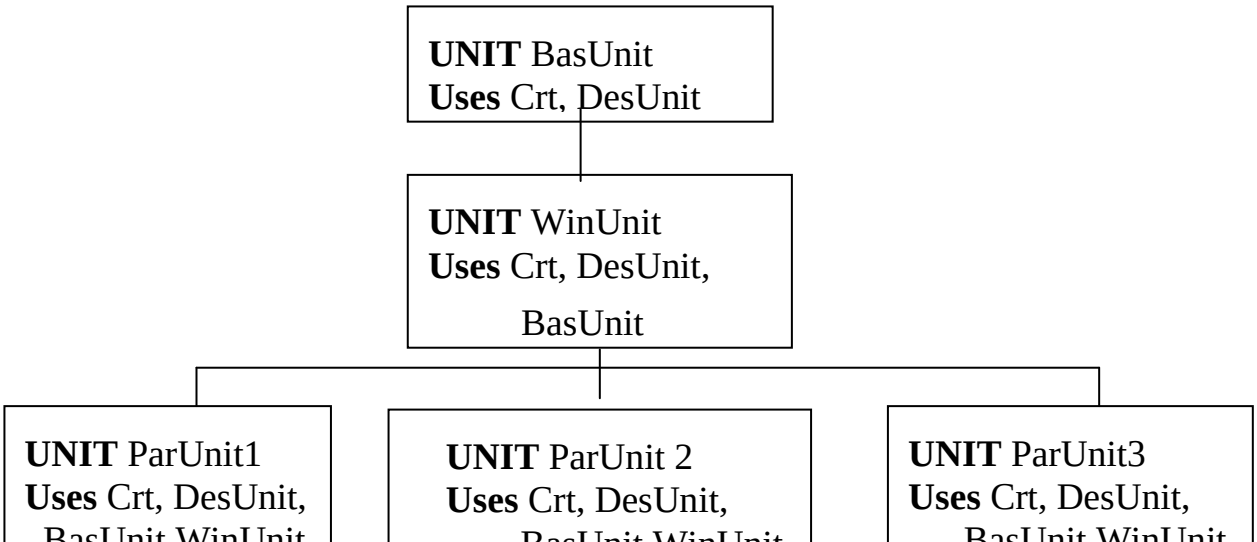
Здесь в модуле `DesUnit` размещены описания глобальных констант, типов и переменных (Des - от слова description, описание). Секция реализации этого модуля пустая. В модуле `BasUnit` сосредоточены базовые служебные процедуры и функции, которые используются во многих других модулях (функция знака, процедуры контроля наличия файлов, бинарного поиска, группировка массивов, вывод массива на экран, удаление стека и т.п.). В модуле `WinUnit` записаны процедуры для организации многооконного интерфейса (включение и отключение курсора, сохранение и восстановление экрана, формирование разного типа меню и т.п.). Модули `ParUnit1`, `ParUnit2`, `ParUnit3` содержат процедуры и функции, реализующие различные режимы работы программы. В основной программе `Example` осуществляется выбор режима ее работы путем активизации позиций меню и обращения к соответствующим процедурам.

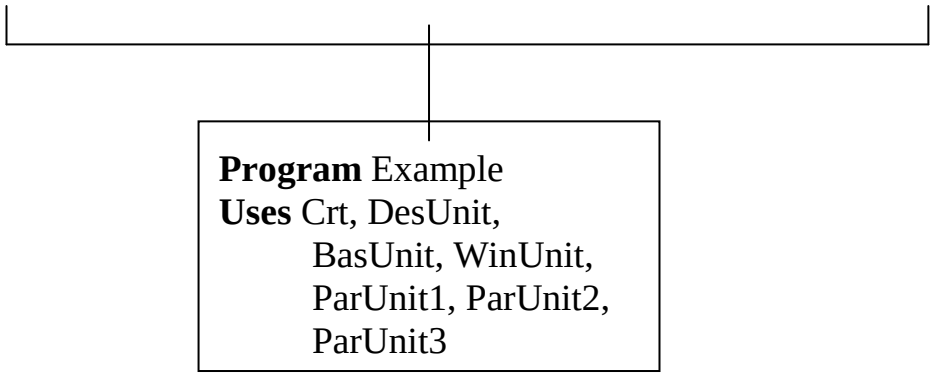
Как известно, все переменные, которые описаны в разделах **`Var`,** размещаются в двух сегментах памяти: в сегменте данных и в сегменте стека. При этом сегмент стека используется для локальных переменных, а сегмент данных – для глобальных переменных и типизированных констант.

Локальными считаются все переменные, которые описаны в блоках процедур и функций, глобальными – те, что описаны вне пределов этих блоков. Следовательно, сегмент данных используется не только для переменных, описанных в разделе **`Var`** основной программы, но и в разделах **`Var`**, расположенных в секциях интерфейса и реализации ее модулей. Хотя последние две группы переменных расположены в одном сегменте памяти, но между ними имеется существенная разница:

- переменные, описанные в секции реализации, могут быть использованы лишь при работе данного модуля;
- переменные, описанные в секции интерфейса, могут быть использованы в данном модуле и во всех других, включающих в себя фразу Uses с именем данного модуля.

Разумеется, эти ограничения обеспечиваются во время компиляции программы.





В лабораторной работе № 9 нужно оформить процедуры и функции, разработанные в работе № 8, в виде набора программных модулей. Дополнительное задание: разработать модуль проверки корректности входных текстовых файлов.

Содержание отчета по лабораторной работе №9 в основном такое же, как и по работе №8. При этом в отчете нужно отметить:

- разделение общей программы на модули;
- перечень контролируемых ошибок в текстовых файлах (в первую очередь содержательных ошибок, характерных для условия именно данной задачи).

Ниже приведен пример преобразования программы `Labor8` к многомодульной структуре. Поскольку программная реализация всех режимов, кроме режима "Контроль текстовых файлов", сохранена практически без изменений, то здесь приводится лишь описание модуля `InpUnit`, осуществляющего проверку параметров, которые вводятся из текстовых файлов.

В рассмотренной задаче используются три исходных текстовых файла: `FileInput`, `FileAdd` и `FileKodif`. Структура записей первых двух файлов одинаковая, их проверку осуществляют одни и те же процедуры. Для контроля параметров третьего файла применяются отдельные процедуры.

Для сообщений об ошибках используется типизированный файл `FileError`, в который последовательно заносится информация о выявленных ошибках. Индикатором наличия ошибок является булевская переменная `FatalError`. Если `FatalError=true`, то в последнюю строку файла `FileError` заносится фраза "*Скорректируйте исходные файлы*", в противном случае - фраза "*В исходных файлах ошибок не обнаружено*". В сообщениях об ошибках указывается имя проверяемого файла, тип ошибки и номер строки, содержащей ошибку. Чтение содержания файла `FileError` выполняет процедура `ReadFileError`. В конечной стадии работы режима контроля файл `FileError` уничтожается.

Для массива `Sf^`, куда с самого начала загружаются строки входного текстового файла, отводится память, необходимая для размещения `2*NfMax` строк, где `NfMax` – константа, которая означает максимально допустимое количество строк в таком файле. Удвоение необходимого размера памяти дает возможность читать избыточные строки файла без выхода за границы массива `Sf^`.

Ошибки в текстовых файлах разделяются на три группы:

- ошибки формата;
- ошибки диапазона;
- содержательные ошибки.

а) О ш и б к и ф о р м а т а.

Проверка осуществляется процедурами FormatFileProduct и FormatFileKodif.

Во входном текстовом файле могут быть пустые строки (их длина l=0) или строки, которые содержат только пробелы. При работе программ Labor7 и Labor8 это привело бы к получению неверных результатов. Тем не менее наличие таких строк нецелесообразно относить к критическим ошибкам, поскольку эти строки легко устранить программным путем. Для этого содержимое текстового файла вводится в динамический массив строк Sf^, из этого массива удаляются строки, которые не содержат значащих символов, после чего массив Sf^ записывается в тот же текстовый файл. Эта работа выполняется процедурой ReadAndCheckSpaces, обращение к которой осуществляется при старте процедур FormatFileProduct и FormatFileKodif.

В процедуре ReadAndCheckSpaces сначала проверяется наличие текстового файла. Отсутствие файла, естественно, блокирует дальнейшую работу по его проверке.

Если текстовый файл существует, то выполняется чтение его строк в массив Sf^. При этом проверяется, не является ли файл пустым (количество строк nf=0), а также, не превышает ли его размер максимально допустимый (nf>Nmax). В том и другом случае дальнейшая проверка файла не производится.

После успешного завершения работы процедуры ReadAndCheckSpaces выполняется непосредственная проверка форматов параметров. В процедуре FormatFileProduct проверяются:

- количество параметров в строке файла (больше или меньше 8);
- длина строки текстового параметра;
- корректность форматов числовых параметров (возможные ошибки: буква вместо цифры, точка в целочисленном параметре и др.).

Процедура FormatFileKodif работает аналогично, но с учетом структуры записи кодификатора.

При отсутствии форматных ошибок будут сформированы динамические массивы Products и AddProducts, а также статический массив KodifProducts.

Если обнаружена хотя бы одна ошибка формата, дальнейший контроль текстовых файлов не производится.

б) О ш и б к и д и а п а з о н о в.

Проверку допустимых диапазонов числовых параметров осуществляют процедуры CheckProdDiapason и CheckKodifDiapason. Здесь в цикле просматриваются массивы Products, AddProducts и KodifProducts. Если какой-нибудь числовой параметр меньше заданного минимального значения или больше заданного максимального значения, то формируется сообщение об ошибке.

Если обнаружена хотя бы одна ошибка диапазона, дальнейший контроль текстовых файлов не производится.

в) С о д е р ж а т е л ь н ы е о ш и б к и.

Проверка этого типа ошибок осуществляется процедурами ProdParameters (массивы Products и AddProducts) и KodifParameters (массив KodifProducts). В первой из них проверяется:

- имеет ли место дублирование кода изделия Kod в компонентах архива;
- имеется ли заданное значение параметра Kod в кодификаторе;
- соответствует ли единица измерения списку допустимых значений.

В процедуре KodifParameters проверяется лишь дублирование кода изделия в кодификаторе.

```
Unit  DesUnit;
```

```
{ Глобальные описания констант, типов и переменных }

Interface

Uses  Crt;

Const
  MaxKodif = 50;     { макс.количество компонент кодификатора }
  Enter = 13;        { код клавиши Enter }
  PressKey = 'Нажмите клавишу ENTER';
Type
  ProductType  = record { тип компонента архива изделий }
    NumberShop : byte;      { номер цеха }
    Kod : longint;          { код изделия }
    Dimens : string[5];     { единица измерения }
    Price : real;           { цена изделия }
    Plan,                   { план выпуска по полугодиям }
    Fact                    { факт.выпуск по полугодиям }
          : array [1..2] of real
  end;
  PointerProduct = ^DynProduct;
  DynProduct = record
    Inf : ProductType ;
    Next : PointerProduct;
  end;
  KodifType = record         { тип компонента кодификатора }
     Kod : longint;          { код изделия }
     Name : string[35];      { наименование изделия }
  end;
  KodifAr = array[1..MaxKodif] of KodifType;
  string80 = string[80];
  StringAr = array[1..10] of string80;

Var
   np : word;              { количество компонент архива }
   nk,                     { количество компонент кодификатора }
   KeyRegime,              { ключ выбора режима работы }
   Device : byte;          { устройство вывода результатов: }
                           { 0 - экран; 1 - экран и магн.диск; }
                           { 2 - экран и принтер }
   SignArchive : boolean;  { признак создания архива }
   Reply : char;           { символ ответа на запрос программы }
   Product : ProductType;  { компонент архива }
   Lp,Rp,                  { левый и правый указатели очереди }
   Run : PointerProduct;   { текущий указатель очереди архива }
   Kodif : KodifType;      { компонент кодификатора }
   Kodifs : KodifAr;       { массив компонентов кодификатора }
   Sr : string80;          { сообщение об ошибке }
   St : StringAr;          { строка для печати таблиц }

   FileInput,              { файл входных документов }
   FileAdd,                { файл добавляемых документов }
   FileKodif,              { файл кодификатора изделий }
   FileRes : text;         { файл результатов }
   FileOut                 { архивный файл изделий }
```

```
       : file of ProductType ;

Implementation

End.

{ ------------------------------------------------------------ }

UNIT BasUnit;
{ Сервисные процедуры и функции }
Interface

Uses Crt,DesUnit,Printer;

Procedure WaitEnter;
Procedure PrintString(X,Y:byte; S:string);
Procedure WritelnString(S:String);
Procedure PrintKeyAndWaitEnter;
Procedure CheckPageScreen(Var j:byte);
Function  Space(S:string; k:byte):byte;
Function  NotSpace(S:string; k:byte):byte;
Function  FillString(S:string; len,p:byte):string;
Function  GetNumber(MinNumber,MaxNumber:real;
                    m1,n1,m2,n2:byte):real;
Procedure UsesDevice;
Procedure PrintHat(n:byte);
Procedure DisposeProduct;
Procedure SortKodif;
Function  SearchKodif(Kod:longint):byte;


Implementation

{ ------------------------------------------------------- }
```
Тексты процедур и функций, которые приведены в секции реализации, полностью совпадают с соответствующими текстами программы Labor8

```
Procedure WaitEnter;
Procedure PrintString(X,Y:byte; S:string);
Procedure WritelnString(S:string);
Procedure PrintKeyAndWaitEnter;
Procedure CheckPageScreen(Var j:byte);
Function  Space(S:string; k:byte):byte;
Function  NotSpace(S:string; k:byte):byte;
Function  FillString(S:string; len,p:byte):string;
Function  GetNumber(MinNumber,MaxNumber:real;
                    m1,n1,m2,n2:byte):real;
Procedure UsesDevice;
Procedure PrintHat(n:byte);
Procedure DisposeProduct;
Procedure SortKodif;
Function  SearchKodif(Kod:longint):byte;

End.
{ ------------------------------------------------------------ }
```

```
UNIT InpUnit;
{ Ввод и контроль корректности текстовых файлов }

Interface

Uses Crt,DesUnit,BasUnit,Printer;

Procedure  CheckFiles;

Implementation

{ ------------------------------------------------------ }
Const
  NfMax = 200; {макс.количество строк в текстовом файле }
Type
  FileStringAr = array [1..2*NfMax] of string80;
  FileStringArPtr = ^FileStringAr;
  ProductAr = array [1..NfMax] of ProductType ;
  ProductArPtr = ^ProductAr;
Var
  nd : byte;  { количество строк в файле FileAdd }
  FatalError  { наличие ошибки в исходных данных }
           : boolean;
  Sf : FileStringArPtr; { массив строк текстового файла }
  Products,
  AddProducts : ProductArPtr;
  FileError   { файл протокола проверок }
       : file of string80;
{ ------------------------------------------------------ }

Procedure ReadAndCheckSpaces(Var F:text; FileName:string80;
                             Var n:byte; Nmax:byte);
{ Ввод текстового файла и удаление из него пустых строк }
Var i,j,k : byte;
    kf : integer;
    SignSpace : boolean;  { признак пустой строки в файле }
    S : string80;
Begin

{ Проверка наличия файла F с именем FileName }
  {$I-} Reset(F); {$I+}
  k:=IOResult;
  If k<>0 then
    Begin
      FatalError:=true;
      Sr:='Отсутствует входной файл '+FileName;
      Write(FileError,Sr);
      Exit;
    End;

{ Чтение текстового файла }
  n:=0;
  While not eof(F) do
    Begin
      Inc(n);
```

```
      Readln(F,Sf^[n]);
    End;
  Close(F);

{ Проверка: является ли файл F пустым }
  If n=0 then
    Begin
      FatalError:=true;
      Sr:='Входной файл '+FileName+' пустой';
      Write(FileError,Sr);
      Exit;
    End;

{ Проверка: превышает ли количество строк макс.значение }
  If n>Nmax then
    Begin
      FatalError:=true;
      Str(Nmax,S);
      Sr:='В исходном файле '+FileName+' свыше '+S+
          ' строк';
      Write(FileError,Sr);
      Exit;
    End;

{ Удаление пустых строк из состава файла F }
  SignSpace:=false;
  i:=1;
  While i<=n do
    Begin
      k:=NotSpace(Sf^[i],1);
      If k=0 then
        Begin
          For j:=i to n-1 do
            Sf^[j]:=Sf^[j+1];
          Dec(n);
          SignSpace:=true;
        End
      Else
        Inc(i);
    End;
  If SignSpace then
    Begin
      Rewrite(F);
      For i:=1 to n do
        Writeln(F,Sf^[i]);
      Close(F);
    End;
End { ReadAndCheckSpaces };
{ ------------------------------------------------------ }

Procedure FormatFileProduct(Var F:text; FileName:string80;
           Var n:byte; Nmax:byte; Var Prod:ProductArPtr);
{ Проверка форматов файлов FileInput и FileAdd }
Var k,k1,k2 : byte;
          i : word;
```

```
       Code : integer;
       Cond : boolean;
   Sa,Sb,S1,S2 : string80;
Begin
  ReadAndCheckSpaces(F,FileName,n,Nmax);
  If FatalError then
    Exit;
  For i:=1 to n do
    Begin
      Sa:=Sf^[i];
      With Product do
        Begin
          Cond:=true; k2:=0; k:=0;
          While Cond do
            Begin
              k1:=NotSpace(Sa,k2+1);
              If k1=0 then
                Cond:=false
              Else
                Begin
                  k2:=Space(Sa,k1+1);
                  If k2=0 then
                    Begin
                      k2:=length(Sa)+1;
                      Cond:=false
                    End;
                  Inc(k);
                  If k>8 then
                    Begin
                      FatalError:=true;
                      Str(i,S1);
                      Sr:='Файл '+FileName+': в строке '+S1+
                          'свыше 8 элементов';
                      Write(FileError,Sr);
                    End;
                  Sb:=Copy(Sa,k1,k2-k1);
                  Case k of
                    1 : Val(Sb,NumberShop,Code);
                    2 : Val(Sb,KodProduct,Code);
                    3 : Begin
                          If length(Sb)>5 then
                            Begin
                              FatalError:=true;
                              Str(i,S1);
                              Sr:='Файл '+FileName': в строке '+
                                 S1+' длина элемента 3 '+
                                 'свыше 5 символов';
                              Write(FileError,Sr);
                            End;
                          Dimens:=FillString(Sb,5,1);
                        End;
                    4 : Val(Sb,Price,Code);
                    5 : Val(Sb,Plan[1],Code);
                    6 : Val(Sb,Plan[2],Code);
                    7 : Val(Sb,Fact[1],Code);
```

```
                    8 : Val(Sb,Fact[2],Code);
                  end;
                  If (k<>3) and (Code<>0) then
                    Begin
                      FatalError:=true;
                      Str(i,S1); Str(k,S2);
                      Sr:='Файл '+FileName+': в строке '+S1+
                       ' неправильный формат элемента '+S2+
                       ' ('+Sb+')';
                      Write(FileError,Sr);
                    End;
                End;
            End;
          If k<8 then
            Begin
              FatalError:=true;
              Str(i,S1);
              Sr:='Файл '+FileName+': в строке '+S1+
                  'меньше 8 элементов';
              Write(FileError,Sr);
            End;
        End;
      Prod^[i]:=Product;
    End;
End { FormatFileProduct };
{ ------------------------------------------------------ }

Procedure FormatFileKodif;
{ Проверка форматов файла FileKodif }
Var  k,k1,k2 : byte;
           i : word;
        Code : integer;
        Cond : boolean;
    Sa,Sb,S1 : string80;
Begin
  ReadAndCheckSpaces(FileKodif,'Kodif.txt',nk,MaxKodif);
  If FatalError then
    Exit;
  For i:=1 to nk do
    Begin
      Sa:=Sf^[i];
      With KodifProduct do
        Begin
          Cond:=true; k2:=0; k:=0;
          While Cond do
           Begin
             k1:=NotSpace(Sa,k2+1);
             If k1=0 then
               Cond:=false
             Else
               Begin
                 k2:=Space(Sa,k1+1);
                 If k2=0 then
                   Begin
                     k2:=length(Sa)+1; Cond:=false
```

```
                End;
              Inc(k);
              If k>3 then
                Begin
                  FatalError:=true;
                  Str(i,S1);
                  Sr:='Файл Kodif.txt: в строке '+S1+
                      'свыше 2 элементов';
                  Write(FileError,Sr);
                End;
              Sb:=Copy(Sa,k1,k2-k1);
              Case k of
                1 : Begin
                      Val(Sb,Kod,Code);
                      If Code<>0 then
                        Begin
                          FatalError:=true;
                          Str(i,S1);
                          Sr:='Файл Kodif.txt: '+' в строке '+
                               S1+' неправильный формат '+
                              'элемента 1'+'('+Sb+')';
                          Write(FileError,Sr);
                        End;
                      End;
                2 : Begin
                      If length(Sb)>35 then
                        Begin
                          FatalError:=true;
                          Str(i,S1);
                          Sr:='Файл Kodif.txt: '+' в строке '+
                               S1+' длина элемента 2 '+
                              ' свыше 35 символов ';
                          Write(FileError,Sr);
                        End;
                       Name:=FillString(Sb,35,1);
                    End;
      3 : Begin
            While Name[length(Name)]=' ' do
              Delete(Name,length(Name),1);
            Sb:=Name+' '+Sb;
            If length(Sb)>35 then
              Begin
                FatalError:=true;
                Str(i,S1);
                Sr:='Файл Kodif.txt: в строке '+S1+
                 ' длина элемента 2 свыше 35 символов';
                Write(FileError,Sr);
              End;
            Name:=FillString(Sb,35,1);
          End;
    end;
  End;
End;
If k<2 then
  Begin
```

```
      FatalError:=true;
      Str(i,S1);
      Sr:='Файл Kodif.txt: в строке '+S1+
          'меньше 2 элементов';
      Write(FileError,Sr);
    End;
  End;
  KodifProducts[i]:=KodifProduct;
 End;
End { FormatFileKodif };
{ ------------------------------------------------------ }

Procedure CheckProdDiapason(FileName:string80; n:byte;
                                  Prod:ProductArPtr);
{ Проверка диапазонов параметров в FileInput и FileAdd }
Var i : byte;
  S1,S2,S3,S4 : string80;
{ ------------------------------------------------------ }
Procedure ReportError1(i,k:word; d1,d2:longint);
Begin
  FatalError:=true;
  Str(i,S1); Str(k,S2); Str(d1,S3); Str(d2,S4);
  Sr:='Файл '+FileName+': в строке '+S1+' элемент '+S2+
      ' вне границ '+S3+'..'+S4;
  Write(FileError,Sr);
End { ReportError1 };
{ ------------------------------------------------------ }
Procedure  ReportError2(i,k:word; d1,d2:real);
Begin
  FatalError:=true;
  Str(i,S1); Str(k,S2); Str(d1:8:1,S3); Str(d2:8:1,S4);
  While S3[1]=' ' do
    Delete(S3,1,1);
  While S4[1]=' ' do
    Delete(S4,1,1);
  Sr:='Файл '+FileName+': в строке '+S1+' элемент '+S2+
      ' вне границ '+S3+'..'+S4;
  Write(FileError,Sr);
End { ReportError2 };
{ ------------------------------------------------------ }
Begin
  For i:=1 to n do
    With Prod^[i] do
      Begin
        If (NumberShop<1) or (NumberShop>99) then
          ReportError1(i,1,1,99);
        If (Kod<100000) or (Kod>999999) then
          ReportError1(i,2,100000,999999);
        If (Price<0.1) or (Price>1000) then
          ReportError2(i,4,0.1,100);
        If (Plan[1]<10) or (Plan[1]>10000) then
          ReportError1(i,5,10,10000);
        If (Plan[2]<10) or (Plan[2]>10000) then
          ReportError1(i,6,10,10000);
        If (Fact[1]<10) or (Fact[1]>10000) then
```

```
          ReportError1(i,7,10,10000);
        If (Fact[2]<10) or (Fact[2]>10000) then
          ReportError1(i,8,10,10000);
       End;
End { CheckProdDiapason };
{ ------------------------------------------------------ }

Procedure  CheckKodifDiapason;
Var   i : byte;
     S1 : string80;
Begin
  For i:=1 to nk do
    With KodifProducts[i] do
      If (Kod<100000) or (Kod>999999) then
        Begin
          FatalError:=true;
          Str(i,S1);
          Sr:='Файл Kodif.txt: в строке '+S1+' элемент 1'+
              ' вне границ 100000..999999';
          Write(FileError,Sr);
        End;
End { CheckKodifDiapason };
{ ------------------------------------------------------ }

Procedure  ProdParameters(Prod:ProductArPtr; n:byte;
                          FileName:string80);
{ Проверка параметров записей в файлах FileInput и FileAdd }
Const Measurs : array[1..2] of string[5] = ('шт.  ','кг   ');
Var   i,j,k : byte;
        Kod : longint;
       Cond : boolean;
       Meas : string[5];
   S1,S2,S3 : string80;
Begin

{ Проверка дублирования параметра Kod }
  For i:=1 to n-1 do
    Begin
      Kod:=Prod^[i].Kod;
      For j:=i+1 to n do
        If Kod=Prod^[j].Kod then
          Begin
            FatalError:=true;
            Str(i,S1); Str(j,S2); Str(Kod,S3);
            Sr:='Файл '+FileName+': равные значения'+
                'Kod '+'в строках '+S1+' и '+S2+' ('+S3+')';
            Write(FileError,Sr);
          End;
    End;

{ Проверка наличия параметра Kod в кодификаторе }
  For i:=1 to n do
    Begin
      Kod:=Prod^[i].Kod;
      k:=SearchKodif(Kod);
```

```
      If k=0 then
        Begin
          FatalError:=true;
          Str(i,S1); Str(Kod,S2);
          Sr:='Файл '+FileName+': код изделия '+S2+
            ' (строка '+S1+ ') отсутствующий в кодификаторе';
          Write(FileError,Sr);
        End;
    End;

{ Проверка единиц размерности }
  For i:=1 to n do
    Begin
      Meas:=Prod^[i].Dimens;
      Cond:=false;
      For j:=1 to 2 do
        If Meas=Measurs[j] then
          Cond:=true;
      If not Cond then
        Begin
          FatalError:=true;
          Str(i,S1);
          While Meas[length(Meas)]=' ' do
            Delete(Meas,length(Meas),1);
          Sr:='Файл '+FileName+': в строке '+S1+
              ' неправильная '+'ед.измерения ('+Meas+')';
          Write(FileError,Sr);
        End;
    End;

End { ProdParameters };
{ ------------------------------------------------------ }

Procedure  KodifParameters;
{ Проверка дублирования параметра Kod в файле FileKodif }
Var   i,j,k : byte;
        Kod : longint;
   S1,S2,S3 : string80;
Begin
  For i:=1 to nk-1 do
    Begin
      Kod:=Kodifs[i].Kod;
      For j:=i+1 to nk do
        If Kod=Kodifs[j].Kod then
          Begin
            FatalError:=true;
            Str(i,S1); Str(j,S2); Str(Kod,S3);
            Sr:='Файл Kodif.txt: равные значения ' +
                'Kod '+'в строках '+S1+' и '+S2+' ('+S3+')';
            Write(FileError,Sr);
          End;
      End;
End { KodifParameters };
{ ------------------------------------------------------ }
```

```
Procedure  ReadFileError;
{ Чтение протокола контроля текстовых файлов }
Begin
  ClrScr;
  Seek(FileError,0);
  While not eof(FileError) do
    Begin
      Read(FileError,Sr);
      Writeln(Sr);
    End;
  PrintKeyAndWaitEnter;
End { ReadFileError };
{ ------------------------------------------------------ }

Procedure CheckFiles;
{ Контроль корректности текстовых файлов }
Var i : byte;
Begin
  Assign(FileError,'Error.dat');
  Rewrite(FileError);
  Sr:=' ';
  Write(FileError,Sr);
  Sr:='      ПРОТОКОЛ КОНТРОЛЯ ТЕКСТОВЫХ ФАЙЛОВ';
  Write(FileError,Sr);
  Sr:=' ';
  Write(FileError,Sr);
  FatalError:=false;
  New(Sf);
  New(Products); New(AddProducts);
  FormatFileProduct(FileInput,'Input.txt',np,Nfmax,Products);
  FormatFileProduct(FileAdd,'Add.txt',nd,Nfmax,AddProducts);
  FormatFileKodif;
  If not FatalError then
    Begin
      CheckProdDiapason('Input.txt',np,Products);
      CheckProdDiapason('Add.txt',nd,AddProducts);
      CheckKodifDiapason;
    End;

  If not FatalError then
    Begin
      SortKodif(nk);
      ProdParameters(Products,np,'Input.txt');
      ProdParameters(AddProducts,nd,'Add.txt');
      KodifParameters;
    End;

  Sr:=' ';
  Write(FileError,Sr);
  If FatalError then
    Sr:='Скорректируйте исходные файлы'
  Else
    Sr:='В исходных файлах ошибок не выявлено';
  Write(FileError,Sr);
  ReadFileError;
```

```
  Dispose(Sf); Sf:=nil;
  Dispose(Products); Products:=nil;
  Dispose(AddProducts); AddProducts:=nil;
  Close(FileError);
  Erase(FileError);
End { CheckFiles };
{ ------------------------------------------------- }

End.



UNIT WorkUnit;
{ Обрабатывающие процедуры и функции }

Interface

Uses Crt,DesUnit,BasUnit,Printer;

Procedure  CreateArchive;
Procedure  SortArchive;
Procedure  PrintArchive;
Procedure  PrintKodif;
Procedure  AddArchive;
Procedure  DeleteArchive;
Procedure  ChangeArchive;
Procedure  WorkUpArchive;

Implementation

{ ------------------------------------------------- }
```

Тексты процедур, заголовки которых приведены в секции реализации, полностью совпадают с текстами соответствующих процедур программы Labor8.

```
Procedure ReadProduct(Var FileInput:text);
Procedure ReadKodif;
Procedure MakeKodifs;
Procedure CreateArchive;
Procedure ReadFileOut;
Procedure WriteFileOut;
Procedure SortArchive;
Procedure PrintArchive;
Procedure PrintKodif;
Procedure AddArchive;
Procedure DeleteArchive;
Procedure ChangeArchive;
Procedure WorkUpArchive;

End.
{ ------------------------------------------------------- }


Program Labor9;
{ Создание, печать, коррекция и обработка архива сведений }
{ о продукции,  которая  выпускается  цехами  предприятия }
```

```
Uses Crt,DesUnit,BasUnit,InpUnit,WorkUnit,Printer;

Begin

  Assign(FileInput,'Input.txt');
  Assign(FileAdd,’Add.txt’);
  Assign(FileKodif,'Kodif.txt');
  Assign(FileRes,'Res.txt');
  Assign(FileOut,'Out.dat');

  ClrScr;
  UsesDevice;
  If Device=1 then
    Rewrite(FileRes);
  SignArchive:=false;
  Repeat
    ClrScr;
    PrintString(15, 7,'Укажите режим работы программы:');
    PrintString(17, 8,'0 - конец работы;');
    PrintString(17, 9,'1 - контроль текстовых файлов;');
    PrintString(17,10,'2 - создание архива изделий;');
    PrintString(17,11,'3 - сортировка компонентов архива;');
    PrintString(17,12,'4 - печать архива изделий;');
    PrintString(17,13,'5 - печать кодификатора изделий;');
    PrintString(17,14,'6 - добавление компонентов в архив;');
    PrintString(17,15,'7 - удаление компонента из архива;');
    PrintString(17,16,'8 - изменение компонента в архиве;');
    PrintString(17,17,'9 - обработка архива');
    Writeln;
    KeyRegime:=Round(GetNumber(0,9,1,0,1,0));

    Case KeyRegime of
      0 :              ;
      1 : CheckFiles   ;
      2 : CreateArchive;
      3 : SortArchive  ;
      4 : PrintArchive ;
      5 : PrintKodif   ;
      6 : AddArchive   ;
      7 : DeleteArchive;
      8 : ChangeArchive;
      9 : WorkUpArchive;
       Else KeyRegime:=0;
    end;
  Until KeyRegime=0;
  If Device=1 then
    Close(FileRes);
End.
```