Министерство образования и науки РФ
Севастопольский государственный университет
Кафедра информатики и управления в технических системах

РАБОТА С СОСТАВНЫМИ ТИПАМИ ДАННЫХ ЯЗЫКА PASCAL

Методические указания

к выполнению лабораторных работ по дисциплине
"Алгоритмизация и программирование"
для студентов очной формы обучения
по направлениям подготовки
27.03.04 "Управление в технических системах" и
09.03.01 " Информатика и вычислительная техника"

Севастополь
2017

УДК 681.5

Работа с составными типами данных языка Pascal: Методические указания к выполнению лабораторных работ №№6-7, входящих в блок № 3 лабораторного практикума по дисциплине «Алгоритмизация и программирование»/ Сост. Д.Н. Старинская, А.А. Кабанов, В.В Захаров. – Севастополь: Изд-во СевГУ, 2017. – 20 с.

Целью методических указаний является оказание помощи студентам при выполнении лабораторных работ, целью которых является изучение возможностей языка Pascal для работы с такими составными типами данных как файлы, строки и записи.

Методические указания предназначены для студентов дневной и изаочной форм обучения по направлениям подготовки 27.03.04 "Управление в технических системах" и 09.03.01 "Информатика и вычислительная техника"

Методические указания рассмотрены и утверждены на заседании кафедры информатики и управления в технических системах (протокол № 7 от 30.09.2017 г.)

Допущено учебно-методическим центром СевГУ в качестве методических указаний.

Рецензент
Кабанов А.А., канд. техн. наук, доцент

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Лабораторные работы данного цикла входят в блок №3 дисциплины «Алгоритмизация и программирование». Для успешной сдачи тем блока необходимо выполнить и защитить лабораторные работы *не позднее 8-ой недели* семестра, а также написать контрольную работу на оценку не менее "удовлетворительно". Образец задания контрольной работы представлен в приложении А.

1. ЛАБОРАТОРНАЯ РАБОТА №6. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ ОБРАБОТКИ МАССИВОВ ЗАПИСЕЙ НА ЯЗЫКЕ PASCAL

1.1. Цель работы

Исследование возможностей языка Pascal для работы с такими составными типами данных как файлы, строки и записи.

1.2. Задание на работу

1) Работа выполняется в среде Turbo Pascal 7.0. Описание лабораторного стенда приведено в методических указаниях к лабораторной работе № 1.

2) Изучите пример программы, приведенный в разделе 1.3.

3) Получите у преподавателя номер варианта индивидуального задания на лабораторную работу. Варианты заданий приведены в таблице 1.1.

4) Создайте программу, осуществляющую формирование и обработку файла (текстового или типизированного file of record…), предназначенного для хранения информации, структура которой задана в таблице 1.1 в соответствии с номером варианта. Программа должна удовлетворять таким требованиям:

– программа должна иметь простое меню

а) формирование файла;

б) просмотр файла;

в) выход из программы;

– при выборе пункта меню «формирование файла» программа должна запрашивать ввод имени файла с клавиатуры. Также должен быть предусмотрен ввод исходных данных с клавиатуры. Программа должна запрашивать у пользователя подтверждение для продолжения или окончания ввода данных;

– при выборе пункта меню «просмотр файла» программа должна запрашивать ввод имени файла с клавиатуры. В программе должен быть реализован вывод содержимого файла на экран в виде таблицы. Также, должна вызываться дополнительная подпрограмма обработки файла. Подпрограмма должна решать задачу, указанную в последнем столбце таблицы 1.1;

– при выборе пункта меню «выход из программы» программа должна завершать работу.

Представляемый при защите отчета файл должен содержать 10 записей.

Таблица 1.1 – Варианты исходных данных

№ вар.	Назначение программы	Поля	Назначение подпрограммы
1.	Регистрационный журнал гостиницы	Фамилия жильца Занимаемый номер Дата поселения (день, месяц, год)	Вывод на экран списка жильцов из данного номера
2.	Список сотрудников предприятия	Фамилия Инициалы Должность Год приема на работу Оклад	Вывод на экран информации о служащих, принятых в данном году
3.	Расписание электричек	Номер электрички Пункт назначения Время отправления (часы, минуты) Время прибытия (часы, минуты)	Вывод на экран информации о поездах, отправляющихся после введенного часа
4.	Список товаров	Код товара Название Цена Количество	Вывод на экран информации о товаре, код которого введен с клавиатуры
5.	Прайс-лист мониторов	Модель Производитель Диагональ (дюймы) Время отклика Цена	Вывод на экран информации о мониторах, время отклика которых не превышает заданного
6.	База данных автовладельцев	Фамилия владельца Марка автомобиля Регистрационный номер Год выпуска	Вывод на экран фамилии владельца самого старого автомобиля
7.	Список книг	Фамилия автора Название книги Год издания Жанр	Вывод на экран названий и авторов всех книг данного года издания
8.	Список продуктовых товаров	Название товара Производитель Цена Сорт Срок хранения	Вывод информации о самом дешевом продукте

Таблица 1.1 – Варианты исходных данных

№ вар.	Назначение программы	Поля	Назначение подпрограммы
9.	Список учебных дисциплин	Название дисциплины Курс Семестр Фамилия преподавателя Наличие экзамена	Вывод списка дисциплин для заданного курса
10.	Библиографический список	Название литературного источника Автор Год издания Категория (книга, статья)	Вывод на экран списка статей, опубликованных после заданного года
11.	Список приглашенных на премьеру фильма	Фамилия Ряд Место Наличие приглашения на банкет	Вывод на экран списка гостей, размещенных в заданном ряду
12.	Прайс-лист фотоаппаратов	Модель Цена Разрешение (в мегапикселях) Объем карты памяти Вес	Вывод на экран списка фотоаппаратов, цена которых не превышает заданной
13.	Телефонный справочник	Фамилия Имя Отчество Номер телефона	Вывод на экран фамилии человека, телефонный номер которого введен с клавиатуры
14.	Список фильмов	Название Жанр Дата выхода в прокат (день, месяц, год)	Вывод на экран информации о фильмах, вышедших после указанной даты

1.3. Краткие теоретические сведения

При решении некоторых задач бывает удобно объединять значения различных типов под одним именем. Такая структура данных называется в `Pascal` *записью*. Запись состоит из фиксированного числа полей. В отличие от массива эти поля могут быть разного типа.

Описание записи в языке Pascal осуществляется с помощью служебного слова record, вслед за которым описываются поля записи. Завершается описание записи служебным словом end.

Например, сводная ведомость с итогами экзаменационной сессии содержит фамилии и имена студентов, а также оценки по учебным дисциплинам (математика, физика, программирование). Отдельную строку ведомости с информацией об одном студенте удобно описать так:

```
type tStudent = record
                    Surname: String[20];
                    Name: String[10];
                    Math: Integer;
                    Phys: Integer;
                    Prog: Integer;
                  end;
 var Stud:   tStudent;
```

Чтобы обратиться к отдельной компоненте записи, необходимо задать имя записи и через точку указать имя нужного поля, например:

```
Stud.Surname := 'Иванов';
Stud.Math := 5;
Stud.Phys := 4;
```

Обращение к компонентам записей можно упростить, если воспользоваться оператором присоединения with. Его структура выглядит следующим образом

```
With <список имен> do <оператор>
```

Приведенный выше фрагмент программы можно переписать так:

```
with Stud do
    begin
        Surname := 'Иванов';
       Math := 5;
       Phys := 4;
   end;
```

Тип записи может быть использован в качестве типа элементов массива. Например, необходимо описать экзаменационную ведомость, содержащую информацию о 30 студентах. Для этого следует объявить массив, элементами которого будт записи типа tStudent.

```
const NMAX=30;
.........................
tGroup = array[1..NMAX] of tStudent;
var Group: tGroup;
```

Каждый элемент массива Group представляет собой запись, содержащую фамилию и имя студента, а также его 3 оценки. Чтобы занести в элемент массива, содержащий данные второго студента, отличную оценку за экзамен по математике можно использовать оператор:

```
Group[2].Math  :=  5;
```

А чтобы i-тому студенту поставить хорошую оценку по физике, можно написать

```
Group[i].Phys  :=  4;
```

1.3.1. Пример программы

Пусть требуется составить программу, создающую и обрабатывающую текстовый файл, содержащий ведомость результатов экзаменационной сессии студенческой группы. Исходные данные приведены в таблице 1.2.

Таблица 1.2 – Исходные данные

Назначение программы	Поля	Назначение подпрограммы
Сводная ведомость результатов экзаменационной сессии студенческой группы.	Фамилия студента Имя Оценка по математике Оценка по физике Оценка по программированию	Вывод на экран списка студентов, имеющих тройки по математике

Кроме того, к программе предъявляются следующие требования. Программа должна предлагать простое меню

1) формирование файла;
2) обработка файла;
3) выход из программы;

– при выборе пункта меню «формирование файла» программа должна запрашивать ввод имени файла с клавиатуры. Далее должен осуществляться ввод данных о студентах с клавиатуры. Программа должна запрашивать у пользователя подтверждение для продолжения или окончания ввода данных;

– при выборе пункта меню «обработка файла» программа должна запрашивать ввод имени файла с клавиатуры. В программе должен быть реализован вывод содержимого файла на экран в виде таблицы. Кроме того, должна быть реализована вспомогательная процедура, осуществляющая вывод на экран списка студентов, имеющих тройки по математике;

– при выборе пункта меню «выход из программы» программа должна завершать работу.

Ниже представлен пример такой программы.

```
program lab7;
const NMAX=30;   {максимально возможное количество студентов
в группе}
type tStudent = record    {тип-запись: данные студента}
          Surname: String[20];     {фамилия}
          Name: String[10];        {имя}
```

```
          Math: Integer;      {оценка по математике}
          Phys: Integer;      {оценка по физике}
          Prog: Integer;  {оценка по программированию}
end;
tGroup = array[1..NMAX] of tStudent;   {тип-массив:    данные
группы}
var Group: tGroup;        {переменная-массив: данные группы}
  f:     Text;   {переменная для работы с текстовым файлом}
  fname: String[30];{переменная-строка  для  хранения  имени
файла}
  n:     Integer; {фактическое количество студентов в группе}
  m:     Integer; {переменная для организации основного меню:
    1- ввод данных, 2 – обработка файла, 3 – выход}
  i:     Integer;          {счетчик}
  cont: Char;         {переменная  для  организации  диалога:
‘y’–продолжать ввод данных, любая другая клавиша – завершить
ввод}
    tmp:   tStudent; {вспомогательная переменная-запись для
организации ввода данных об одном студенте}
{*********************************************}
procedure Screen3Math(var gr: tGroup; ng: integer);
var i: integer;
begin
     writeln('Math=3:');
     for i:=1 to ng do
               if gr[i].Math=3 then
                  writeln(gr[i].Surname, gr[i].Name:10);
end;
{*********************************************}
begin
     writeln('Programma "Examen Vedomost" ');
     writeln('1 - Sozdat novy file');
     writeln('2 - Obrabotat gotovy file');
     writeln('3 - Zavershit raboty programmy');
     m:=1;
     while  m<> 3 do {пока пользователь не ввел 3 для выхода
из программы}
       begin
         write('Vvedite 1,2 ili 3: '); readln(m);  {читаем m
с клавиатуры}
         case m of
             1: {********Ввод данных в файл (m=1)*******}
              begin
                  write('Vvedite imya faila ');
                  readln(fname);
                  assign(f,fname);
                  rewrite(f);
```

```
                  i:=1; cont:='y';
                  while cont ='y' do
                    with tmp do
                       begin
                     writeln('vvedite   dannye  ',i,  '-togo
studenta');
{читаем данные i-того студента с клавиатуры в поля перемен-
ной tmp}
                     write('Familia: ');  readln(Surname);
                     write('Imya: ');      readln(Name);
                     write('Matem: ');   readln(Math);
                     write('Physic: ');    readln(Phys);
                     write('Programm: ');   readln(Prog);
{записываем в текстовый файл поля переменной tmp отдельной
строкой}
                     writeln(f,Surname:20,Name:10,Math:5,
Phys:5,Prog:5);
                          i:=i+1;
{просим пользователя нажать «y» для продолжения, нажатие лю-
бой другой клавиши завершит цикл ввода данных (while cont
='y' do...)}
      write('klavisha y – prodolzit, drugaya  – stop : ');
      readln(cont);
      end;
      close(f);
     end;
       2:    {************обработка файла (m=2)**********}
     begin
               write('Enter file name ');
               readln(fname);
               assign(f,fname);
               reset(f);          {открываем файл на чтение}
               i:=1;
               while not eof(f) do    {пока не конец файла}
                  begin
                   with group[i] do {читаем из файла строку
с данными студента }
readln(f,Surname,Name,Math,Phys,Prog);
                   i:=i+1;
                  end;
     close(f);{закрываем файл}
     n:=i-1; {вычисляем фактическое количество студентов}
     for i:=1 to n do{выводим данные из массива на экран по-
строчно}
     with group[i] do
         writeln(Surname:20,Name:10, Math:5, Phys:5,Prog:5);
         Screen3Math(group,n); { вывод списка троечников}
```

```
     end;
     end;     {case}
     end;     { while m<>3 do}
end.
```

1.3.2. Рекомендации и замечания

Для правильной интерпретации данных типа *string* при их вводе из текстового файла ограничивайте их размер при описании (в примере, Surname: String[20];), а формируя этот файл задавайте формат вывода равным этому ограничению (в примере, writeln(f,Surname:20,...)). Формат вывода для *числовых* данных следует подбирать так, чтобы перед каждым числом в текстовом файле был, по меньшей мере, один пробел (в примере, writeln(f,...,Math:5,...)).

– В примере программы размер массива определяется значением константы NMAX=30. Однако, при работе программы могут быть задействованы не все ячейки массива. Пользователь вводит в файл произвольное количество записей с данными о студентах. При чтении данных из файла в массив подсчитывается количество фактически заполненных элементов массива и запоминается в переменную n. Это значение и используется для дальнейшей обработке массива в процедуре Screen3Math. Такой прием работы с массивом используется весьма часто. Поскольку размер массива должен быть зафиксирован в описании типа, задается заведомо большое количество элементов массива. Далее в программе определяется количество фактически используемых ячеек (оно должно быть меньше или равно размеру массива), и все действия осуществляются с нужными ячейками. Этот подход имеет недостаток – нерациональное использование памяти. *Рекомендация:* подумайте, как осуществить контроль за тем, чтобы пользователь не ввел более 30 записей в файл, а также, чтобы программа не попыталась считать более 30 записей из файла.

– В примере значения оценок по дисциплинам должны быть в пределах от 2 до 5, значение переменной m, определяющей выбор пункта меню, должно быть в пределах от 1 до 3. *Рекомендация*: попробуйте использовать интервальный тип данных для представления целочисленных значений из ограниченного диапазона. Также можно попробовать использовать перечисляемый тип данных.

1.4. Содержание отчета и порядок защиты работы

Выполнение и защита лабораторной работы производится каждым студентом индивидуально. Защита результатов лабораторной работы осуществляется при наличии работающей программы и полностью оформленного отчета.

Отчет должен включать в себя следующие разделы

- титульный лист;
- цель работы,
- постановка задачи. Этот раздел должен содержать вариант задания;

– схему программы и схему подпрограммы.

– текст программы на языке Pascal;

– результаты работы программы (вид информации, сохраненной в файле, и результат работы подпрограммы).

– выводы.

Защита работы состоит в следующем:

– представление работающей программы на компьютере;

– предъявление отчета, оформленного в соответствии с указанными требованиями;

– ответы на вопросы преподавателя по теоретической и практической части работы. Примеры возможных вопросов приведены в подразделе 0.5.

–

1.5. Контрольные вопросы

1. Что собою представляет тип данных - запись?

2. Какие операции допустимы над переменными типа запись?

3. Как осуществляется обращение к полю записи? Для чего используется оператор присоединения with?

4. Что собою представляет запись с вариантной частью?

5. Что собою представляет тип данных – файл? Сопоставьте характеристики типов файл и массив.

6. Что собою представляет буферная переменная, связанная с файлом?

7. Что такое текстовый файл?

8. Для чего предназначены и как работают процедуры Assign и Close?

9. Для чего предназначены и как работают процедуры Reset и Rewrite?

10. Для чего предназначены и как работают процедуры Read и Readln?

11. Для чего предназначены и как работают процедуры Write и Writeln?

12. Для чего предназначены и как работают функции Eof и Eoln?

2. ЛАБОРАТОРНАЯ РАБОТА №7. ПРОГРАММИРОВАНИЕ АЛГОРИТМОВ РАБОТЫ С ДИНАМИЧЕСКИМИ ПЕРЕМЕННЫМИ ЯЗЫКА PASCAL

2.1. Цель работы

Исследование базовых возможностей языка Pascal для работы с динамическими переменными и указателями.

2.2. Задание на работу

Работа выполняется в среде Turbo Pascal 7.0. Описание лабораторного стенда приведено в методических указаниях к лабораторной работе № 1.

Необходимо решить задачу, поставленную в предыдущем разделе, разместив данные в динамической памяти, используя массив указателей на структуры.

В задаче лабораторной работы №7 следует отметить важную проблему. Поскольку размер массива должен быть зафиксирован в описании типа, задается заведомо большое количество элементов массива. Это приводит к нерациональному использованию памяти. Чтобы устранить этот недостаток, необходимо разместить данные в динамической памяти. Стандартными средствами Pascal динамический массив создать нельзя*. Для помещения данных в динамическую память можно использовать массив указателей на элементы данных, то есть массив, каждым элементом которого является указатель на переменную типа структуры. Графическое изображение примера такой структуры данных показано на рисунке 2.1.

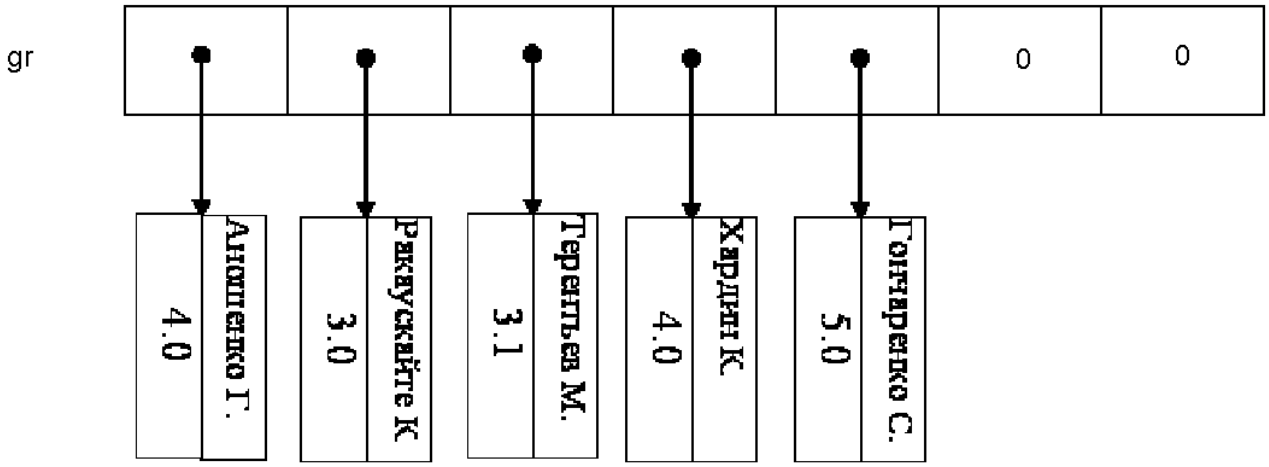

Рисунок 2.1 – Массив указателей на динамические переменные

* В Borland Pascal динамический массив создать все же можно. Для этого нужно объявить указатель на массив нужного типа, а память выделить под необходимое количество элементов процедурой getmem. См., например, учебник Т.А. Павловской «Паскаль. Программирование на языке высокого уровня», стр. 128. Или http://forum.sources.ru/index.php?showtopic=51192.

2.3. Краткие теоретические сведения. Пример программы

Основное свойство динамических переменных заключается в том, что они создаются и память для них выделяется во время выполнения программы. Размещаются динамические переменные в динамической области памяти (heap - области).

Динамическая переменная не указывается явно в описаниях переменных и к ней нельзя обратиться по имени. Доступ к таким переменным осуществляется с помощью указателей.

Указатель – это переменная, значением которой является адрес другой переменной определенного типа, называемого базовым типом.

Множество значений переменной-указателя состоит из адресов ячеек памяти и специального значения NIL, которое ни на что не указывает. NIL не является реальным адресом, а означает отсутствие ссылки на какой-либо объект. Не путайте со случаем, когда значение указателя не определено, NIL – вполне определенное значение.

Для обращения к значению переменной, адрес которой хранится в указателе, применяется операция разыменования (разадресации), обозначаемая с помощью символа ^ справа от имени указателя.

Работа с динамической областью памяти в Pascal реализуется с помощью процедур New и Dispose.

Процедура New(p) выделяет место в динамической области памяти для размещения динамической переменной p^ и ее адрес присваивает указателю p.

Процедура Dispose(p) освобождает участок памяти, выделенный для размещения динамической переменной процедурой New, и значение указателя p становится неопределенным.

Порядок работы с динамической переменной:

– создать (отвести для нее место в динамической памяти;

– работать с ней при помощи указателя (ее адреса в памяти);

– удалить (освободить занятое этой переменной место).

Приведенный ниже пример покажет работу с динамическими переменными и указателями:

```
var p1,p2:^real; {описание указателей на динамическую пере-
                  менную типа real (конкретный тип нужен для
                  определения размера)}
...
New(p1); {создать динамическую переменную и присвоить указа-
                  телю p1 значение ее адреса}
p1^:=3.1415; {заполнение динамической переменной}
p2:=p1; {теперь указатель p2 показывает на ту же переменную}
writeln(p2^); {можно в этом убедиться, распечатав ее значе-
                  ние}
dispose(p1); {удалить переменную, освободив динамическую па-
                  мять}
```

С целью наглядно проиллюстрировать дополнительные возможности, возникающие при программировании с применением динамических переменных и указателей, рассмотрим тот же пример, приведенный в пункте 1.3.1 описания лабораторной работы №6.

Пусть по-прежнему требуется составить программу, создающую и обрабатывающую текстовый файл, содержащий ведомость результатов экзаменационной сессии студенческой группы. Исходные данные приведены в таблице 2.1.

Таблица 2.1 – Исходные данные для программы

Назначение программы	Структура	Функция
Сводная ведомость результатов экзаменационной сессии студенческой группы.	**Данные о студенте:** Фамилия студента Имя Оценка по математике Оценка по физике Оценка по программированию	Вывод на экран списка студентов, имеющих тройки по математике

Ниже приводится текст программы, соответствующий исходным данным таблицы 2.1 и наглядно иллюстрирующий возможности и способы применения динамических переменных и указателей при программировании.

```
program lab7;
const NMAX=30;   {максимально возможное количество студентов
в группе}
type tStudent = record    {тип-запись: данные студента}
                      Surname: String[20];
                      Name: String[10];
                      Math: Integer;
                      Phys: Integer;
                      Prog: Integer;
                end;
  pStudent = ^tStudent;   {Тип-указатель на запись}
  tGroup = array[1..NMAX] of pStudent; {тип-массив
указателей}
var Group: tGroup;        {переменная-массив указателей}
    f:     Text; {переменная для работы с текстовым файлом}
    fname:   String[30];
    m:   Integer; {переменная для организации основного меню:
           1- ввод данных, 2 – обработка файла, 3 – выход}
    i:   Integer;         {счетчик}
```

```
    cont: Char; {переменная для организации диалога: 'y'-
продолжать ввод данных, любая другая клавиша-завершить ввод}
    tmp:   pStudent;{вспомогательная переменная-указатель}
{*********************************************}
procedure Screen3Math(var gr: tGroup);
var i: integer;
begin
     writeln('Math=3:');
     i:=1;
     while gr[i] <> NIL do
        begin
           if gr[i]^.Math=3 then
                writeln(gr[i]^.Surname, gr[i]^.Name:10);
           i:=i+1;
        end;
end;
{*********************************************}
begin
     writeln('Programma "Examen Vedomost" ');
     writeln('1 - Sozdat novy file');
     writeln('2 - Obrabotat gotovy file');
     writeln('3 - Zavershit raboty programmy');
     m:=1;
     while  m<> 3 do {пока пользователь не ввел 3 для выхода
из программы}
       begin
         write('Vvedite 1,2 ili 3: '); readln(m);  {читаем m
с клавиатуры}
         case m of
             1:  {********Ввод данных в файл (m=1)*******}
              begin
               write('Vvedite imya faila ');
               readln(fname);
               assign(f,fname);
               rewrite(f);
               i:=1; cont:='y';
               while cont ='y' do
               begin
                New(tmp); {Создаем динамическую переменную}
                with tmp^ do
                begin
                  writeln('vvedite  dannye ',i, '-togo stu-
denta');
{читаем данные i-того студента с клавиатуры в поля перемен-
ной, на которую указывает tmp}
                  write('Familia: ');  readln(Surname);
                  write('Imya: ');      readln(Name);
```

```
                write('Matem: ');   readln(Math);
                write('Physic: ');    readln(Phys);
                write('Programm: ');   readln(Prog);
                writeln(f,Surname:20,Name:10,Math:5,
Phys:5,Prog:5);
                Dispose(tmp);{Удаляем  динамическую  пере-
менную}
                i:=i+1;
{просим пользователя нажать «y» для продолжения}
                write('klavisha y - prodolzit, drugaya  -
stop : ');
                readln(cont);
               end;
        end; close(f);
       end;
       2: {************обработка файла (m=2)**********}
        begin
        write('Enter file name ');
        readln(fname);
        assign(f,fname);
        reset(f);         {открываем файл на чтение}
        for i:=1 to NMAX do
        group[i] := NIL;  {Заполняем массив указателей значе-
ниями NIL}
        i:=1;
        while not eof(f) do       {пока не конец файла}
           begin
           New(group[i]); {выделение динамической памяти под
новую запись}
           with group[i]^ do {читаем из файла строку с дан-
ными студента }
           readln(f,Surname,Name,Math,Phys,Prog);
           i:=i+1;
           end;
           close(f);  {закрываем файл}
           i:=1;
           while group[i]<>NIL do {выводим данные  на экран}
              begin
              with group[i]^ do
              writeln(Surname:20,Name:10,            Math:5,
Phys:5,Prog:5);
              i:=i+1;
              end;
           Screen3Math(group); { вывод списка троечников}
           while group[i]<>NIL do
            begin
```

```
                Dispose(group[i]);     {Удаляем данные из дина-
мической памяти}
                i:=i+1;
              end;
            end;
      end;     {case}
      end;     { while m<>3 do}
end.
```

2.4. Содержание отчета и порядок защиты работы

Выполнение и защита лабораторной работы производится каждым студентом индивидуально. Защита результатов лабораторной работы осуществляется при наличии работающей программы и полностью оформленного отчета.

Отчет должен включать в себя следующие разделы:

- титульный лист;
- цель работы;
- постановка задачи. Этот раздел должен содержать вариант задания;
- схема и текст программы на языке Pascal;
- результаты работы программы (вид информации, сохраненной в файле);
- выводы.

Защита работы состоит в следующем:

- представление работающей программы на компьютере;
- предъявление отчета, оформленного в соответствии с указанными требованиями;
- ответы на вопросы преподавателя по теоретической и практической части работы. Примеры возможных вопросов приведены в подразделе 1.5.

2.5. Контрольные вопросы

1. Что собою представляет тип данных «запись»?

2. Указатели в языке Pascal. Операции взятия адреса и разыменования.

3. Динамические переменные в языке Pascal. Динамическое выделение памяти с помощью процедуры new().

4. Динамические переменные в языке Pascal. Освобождение динамической памяти с помощью процедуры dispose().

5.Что собою представляет тип данных – файл? Сопоставьте характеристики типов файл и массив.

6. Что такое типизированный файл?

7. Что такое текстовый файл?

8. Для чего предназначены и как работают процедуры `Assign` и `Close`?

9. Для чего предназначены процедуры `Reset` и `Rewrite`?

10. Для чего предназначены и как работают процедуры `Read` и `Readln`?

11. Для чего предназначены процедуры `Write` и `Writeln`?

12. Для чего предназначены и как работают функции `Eof` и `Eoln`?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Пильщиков В.Н. Сборник упражнений по языку Pascal: Учебное пособие для вузов / В.Н. Пильщиков. – М.: Наука, 1989г. – 160с.

2. Павловская Т. А. Паскаль: программирование на языке высокого уровня: практикум [Текст] : учеб. пособие для студ. вузов / Т. А. Павловская. – М. и др.: Питер, 2006. – 408 с.

3. Павловская Т. А. Паскаль: программирование на языке высокого уровня [Текст] : учеб. для студ. вузов / Т. А. Павловская. – М. и др. : Питер, 2006. – 400 с.

ПРИЛОЖЕНИЕ А.
ОБРАЗЕЦ ЗАДАНИЯ КОНТРОЛЬНОЙ РАБОТЫ №3

(справочное)

Вариант № 30 – Образец

Задание №1 (5 баллов)

Процедура assign служит для

1) открытия существующего файла на чтение;
2) завершения работы с файлом;
3) чтения данных из текстового файла;
4) проверки, достигнут ли конец файла;
5) связывания файловой переменной с внешним файлом.

Номер правильного ответа: _____

Задание №2 (5 баллов)

Что из нижеследующего неверно:

1) Процедура New(p) создает новую динамическую переменную и записывает ее адрес в указатель p;
2) Процедура reset открывает файл на чтение.
3) Функция `eof(f)` закрывает файл, с которым связана переменная f.
4) Процедура writeln(f) выводит в текстовый файл f символ перехода на следующую строку.
5) Операция «.» (точка) служит для обращения к полю записи.

Номер правильного ответа: _____

Задание №3 (5 баллов)

Значение выражения

```
[5,7,11]<=[4..15]
```

равно

1) [];
2) 1;
3) False;
4) True;
5) -1.

Номер правильного ответа: _____

Задание №4 (Максимальная оценка 15 баллов)

Найдите ошибки в следующих операторах и запишите правильные варианты:

а) пусть описаны переменные `Var F : File Of Real; A : Real;`

Следующий оператор должен считать из файла F значение переменной A:

```
ReadLn(F,A);
```

б) Пусть описана переменная `Var F : File Of Real;`

Следующий цикл должен выполняться, пока не закончится файл `F:`

```
While Not Eof Do
.  .  .  .  .
```

```
в)rewrite(f,'new.txt');
```

```
г) read(x:10:2);
```

Задание №5 (Максимальная оценка 20 баллов)

Даны два *типизированных* файла целых чисел одинакового размера. Создать *текстовый* файл, содержащий эти числа, расположенные в два столбца шириной по 30 символов (в первом столбце содержатся числа из первого исходного файла, во втором – из второго файла).