
ФАЙЛЫ

Файлы

Файлом называется совокупность данных, записанная во внешней памяти под определенным именем. Целесообразность применения файлов диктуется следующими причинами:

1. Ввод больших объемов данных, подлежащих обработке, утомителен и требует большого времени. Гораздо удобнее создать отдельный файл данных, который может быть подготовлен заранее и, самое главное, применяться неоднократно.
2. Файл данных может быть подготовлен другой программой, становясь, таким образом, связующим звеном между двумя разными задачами, а также средством связи программы с внешней средой.
3. Программа, использующая данные из файла, не требует присутствия пользователя в момент фактического исполнения.

Любой файл имеет три характерные особенности. Во-первых, у него есть имя, что дает возможность программе работать одновременно с несколькими файлами. Во-вторых, он содержит компоненты одного типа. В-третьих, длина вновь создаваемого файла никак

ОПРЕДЕЛЕНИЕ ПОНЯТИЙ

- **Физический Файл** – это поименованная область на диске, содержащая какую-либо информацию.

- **Логический файл** – это одна из структур данных, используемых в программировании.

СТРУКТУРА ЛОГИЧЕСКОГО ФАЙЛА

Это способ восприятия файла в программе, т.е. «шаблон», через который мы смотрим на физическую структуру файла на диске. В ЯП таким шаблонам соответствуют типы данных, допустимые в качестве компонент файлов.

File of byte:

байт	байт	. . .	байт	**Eof**

File of char:

код символа	**код символа**	. . .	**код символа**	**Eof**

File of integer:

целое со знаком	**целое со знаком**	. . .	**целое со знаком**	**Eof**

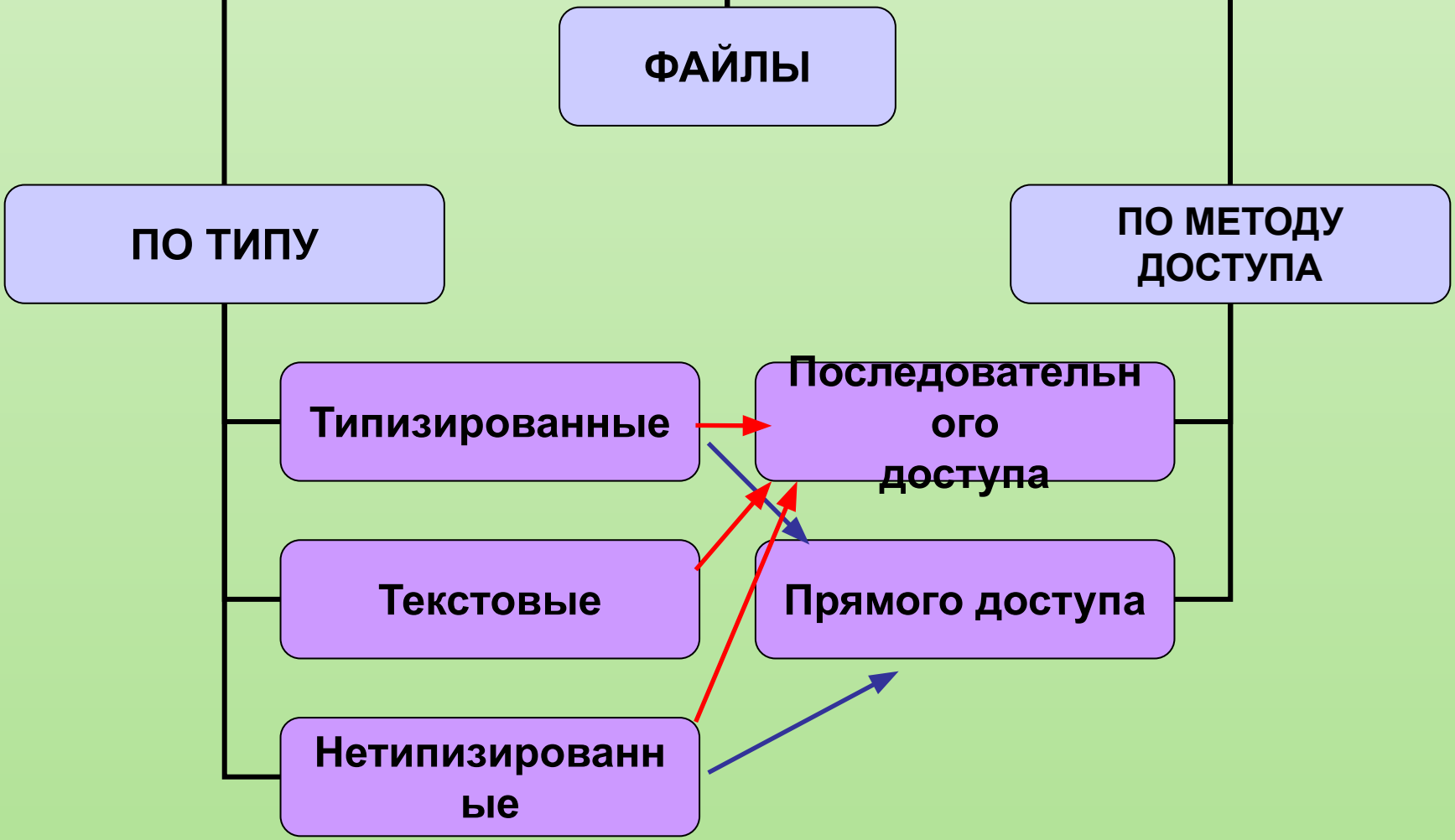
И другие

Логическая структура файла в принципе очень похожа на структуру массива.

Различия:

- У массива количество элементов фиксировано, а у файлов количество элементов может изменяться в процессе работы.(Количество в каждый момент времени неизвестно, но в конце файла стоит символ Eof)
- Массив целиком располагается в ОП, а файл находится на диске.
- Нумерация элементов массива выполняется соответственно значений нижней и верхней границ, указанных при его объявлении. Нумерация элементов файла выполняется слева направо, начиная с нуля

Классификация Файлов в ПАСКАЛЕ



ИСПОЛЬЗОВАНИЕ

Файлы используются для хранения данных. Из них можно считывать начальные данные, записывать результаты, изменять информацию в файле.

РАБОТА С ТЕКСТОВЫМИ ФАЙЛАМИ

var список имен файлов : text;

Текстовый файл может состоять из любых символов (в том числе и цифр)

Для работы с каким-либо физическим файлом (тем, который существует на диске) его необходимо связать с файловой переменной

Assign (имя файла, ‘путь к файлу’);

НАПРИМЕР:

На диске есть файл:

D:\MyFile.dat

```
. . .
Var f : text;
. . .
Begin
. . .
Assign (f; ‘D:\MyFile.dat’ );
```

Или:

```
Var f : text;
Name: string;
. . .
Begin
. . .
Name : = ‘D:\MyFile.dat ‘ ;
Assign ( f, name);
```

ПРИНЦИПЫ РАБОТЫ С ФАЙЛАМИ

1. Открытие
2. Чтение из файла или запись в файл
3. Закрытие

ПРИНЦИПЫ РАБОТЫ С ФАЙЛАМИ

RESET (название файла) – открытие файла для чтения из него информации в ОП. Если файла не существует, то будет выведена ошибка

REWRITE (название файла) – открытие файла для записи данных в файл. Если файла не существует, то он будет создан

APPEND (название файла) – открытие с целью дополнения данных. Если файла не существует, то он будет создан

CLOSE (название файла) – закрытие файла с сохранением

ИСПОЛЬЗОВАНИЕ ДАННЫХ ИЗ ФАЙЛА

- Для считывания данных из файла в ОП используют **read** и **readln**.
- Запись в файл осуществляется процедурами **write** и **writeln**

ИСПОЛЬЗОВАНИЕ ДАННЫХ ИЗ ФАЙЛА

- **read(f,a,b);** — читать из файла f две переменные a и b. После выполнения этой процедуры указатель в файле передвинется за переменную b;
- **readln(f,a,b,c);** — читать из файла f три переменные a, b и c, а затем перевести указатель (курсор) на начало следующей строки; если кроме уже считанных переменных в строке содержалось еще что-то, то оно будет проигнорировано.
- **write(f,a,b,c);** — записать в файл f переменные a,b и c;
- **writeln(f,a,b);** — записать в файл f переменные a и b, а затем записать туда же символ "конец строки".

- Если в списке переменных есть числовая переменная (integer или real), то считываются символы, которые трактуются как цифры до ближайшего пробела.
- Т. О. особенностью текстового файла является то, что происходит автоматическое преобразование числовых данных в цепочку символов при записи в файл и обратное преобразование символов в цифры при чтении из файла.

Функции и директивы для работы с файлами

- **Функция Eof (имя файла) – true, если достигнут конец файла и False – иначе.**

- **Функция FileExists(‘имя_файла’) определяет наличие файла на диске (true – есть, false – нет).**

ПРИМЕРЫ ПРОГРАММ

Задача № 1

```
var
a,b,sum:integer;
f1,f2: text;
begin
assign(f1,'input.txt');
assign(f2,'res.txt');
reset(f1);
read(f1,a,b);
sum:=a+b;
rewrite(f2);
writeln(f2,sum);
close(f2);
close(f1);
end.
```

ПРИМЕРЫ ПРОГРАММ

Задача № 2

Пусть на диске (в текущем каталоге) есть файл myfile.dat, который состоит из некоторого числа целых чисел, разделенных пробелами. Написать программу, вычисляющую сумму этих элементов.

```
Program Files;
uses Crt;
var f :text;
  x: integer;
  Summa:longint;
begin
clrscr;
assign(f, 'myfile.txt');
if not
   FileExists('myfile.txt')
then
writeln('ошибка
   открытия файла')
else
begin
reset(f);
Summa:=0;
  while not Eof(f) do
  begin
    read(f,x);
    Summa:=Summa+x;
    end;
    Writeln('Summa= ',
Summa:8);
    Close(f);
    end;
    readln;
  end.
```

СОЗДАНИЕ ФАЙЛОВ

1 способ – с помощью текстового редактора, например Блокнот или Pascal.

2 способ – программными средствами. Открыть файл процедурой Rewrite (f)

Процедурой REWRITE нельзя открыть запись информации в уже существующий файл. При выполнении этой процедуры старый файл с таким же именем уничтожается и никаких сообщений в программу не передается.

Задача № 3

Написать программу, в которой в текстовый файл записываются данные про 10 учеников: имя, вес и рост.

Перед созданием файла программа должна проверять наличие файла с таким именем на диске и спрашивать, что ей делать в случае обнаружения такого файла – прекратить работу или перезаписать файл.

```
Program Zapfile;
Uses Crt;
Const db='deti.txt';
var f: text;
    name: string [10];
    ves: real;
    ROST: real;
    Otvet: char;
procedure zapf;
begin Rewrite(f);
For var i:=1 to 10 do Begin
Writeln('введите имя, вес и рост:');
Readln(name, ves, rost);
Write(f,name:10,' ',ves:4,' ');
writeln(f,rost:5:2)
End;
Close(f);
end;
Begin
Clrscr;
Assign(f, db) ;
{проверяем, существует ли
 такой файл}
If FileExists(db) then
Begin
writeln('файл  deti.txt
существует. Заменить его?
(y/n)');
Readln(otvet);
if otvet = 'n' then halt
Else if otvet = 'y' then
zapf;
End
Else
zapf;
Readln;
End.
```

Задача № 4

Написать программу, которая считывает слова из одного текстового файла и записывает их в столбик в другой текстовый файл.

Пояснение: слова разделяются символом пробел. Поэтому мы будем считывать символы из первого файла и «складывать» их в слово до тех пор, пока не встретиться пробел. Потом это слово запишем во второй файл и опять начнем формировать следующее слово. И так до тех пор, пока не достигнем конца первого файла.

```
Program slovo;
{Запись слов из файла f
в столбик в файл h}
uses Crt;
var f,h:text;
    bukva:char;
    clovo:string;
begin  clrscr;
assign(f,'f.pas');
assign(h,'h.pas');
 reset(f);
 rewrite(h);
 clovo:=' ';
 while not eof(f) do
begin
  read(f, bukva);
  if bukva<>' ' then
   begin
       clovo:=clovo+bukva;
   end
  else
    begin
    writeln(h,clovo);
     writeln(clovo);
    clovo:=' ';
     End ;
end;
  readln;
  Close (f);   Close (h);  end.
```

Задание на урок:

- Написать программу, которая на диске компьютера создает файл nambers.txt и записывает в него 5 введенных с клавиатуры целых чисел. При помощи текстового редактора (например, БЛОКНОТА) просмотрите файл и убедитесь, что запись в файл произошла.

- Написать программу, которая дописывает в файл nambers.txt 5 введенных с клавиатуры целых чисел. При помощи текстового редактора (например, БЛОКНОТА) просмотрите файл и убедитесь, что запись в файл произошла.

Типизированные файлы

Типизированные файлы

- это файлы, все элементы в которых одного типа

```
var  f: file of integer;
       f1: file of real;
       f3: file of string[10];
```

При работе с типизированным файлом можно сначала определить тип данных, а затем определить файл:

```
type massiv=array[1..5] of integer;
 var f : file of massiv;
```

В типизированном файле можно использовать любой тип кроме файлового.

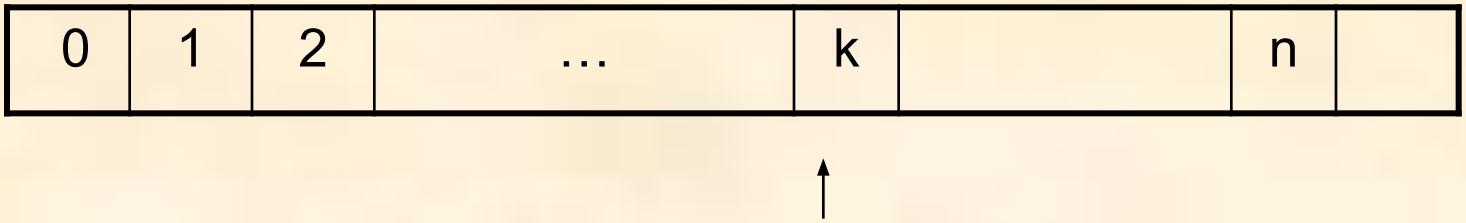
- Типизированные файлы допускают как последовательный, так и прямой доступ. При использование прямого доступа следует помнить, что элементы файла всегда нумеруются, начиная с нуля!

- Чтение из типизированного файла осуществляется **только** процедурой **Read**, а запись-**Write**

- **Read**(файловая переменная, список переменных);

- **Write**(файловая переменная, список переменных);

При считывании в каждую переменную из списка переменных указатель текущей позиции в файле перемещается на следующий элемент.



указатель текущей позиции

Если указатель текущей позиции находится за последним элементом, т.е в позиции EOF(f)=true, то выполнение процедуры Read приводит к ошибке.

При записи в файл, указатель перемещается на следующий элемент.

При достижении конца файла-файл расширяется

Процедуры и функции для работы с типизированными файлами

название	действие
FilePos функция	-определяет номер текущей позиции указателя в файле (начиная с нуля)
Seek процедура	-перемещает указатель текущей позиции в файле на элемент с заданным номером (начиная с нуля!)
FileSize функция	-определяет текущий размер файла (число элементов файла, начиная с единицы)
Truncate процедура	-усекает размер файла до текущей позиции. Все элементы, расположенные после текущей позиции, усекаются и она становится его концом EOF(f)=true;

Недостатки типизированных файлов:

- Нельзя создать или просмотреть в текстовом редакторе.
- В него нельзя дописывать данные в режиме append.

Пример.

Составить программу, которая создает типизированный файл, состоящий из случайного числа случайных целых чисел, выводит этот файл на экран компьютера. Затем сортирует элементы файла и выводит на экран уже отсортированный файл.

*чтобы получить данные из типизированного файла в читаемом виде, необходимо вывести их на экран не через стандартный редактор, а средствами вывода PascalABC.

```
Program Filesort;
uses crt;
var f: file of integer;
    x, y  : integer;
    i, j    : integer;
begin
     {$I-}
Clrscr;
assign (f,  'sort_dat.pas');
     {$I+}      { Создаем файл случайных чисел}
Rewrite ( f );
Randomize;
J :=random( 100 ); { количество элементов файла }
For i:=1 to j do
     begin
       x:=random(100);
       write( f, x );
     End;
close ( f );
writeln ('исходный файл');
reset ( f );
for i:=1 to fileSize ( f ) do
    begin
      read ( f, x );
      write ( x:4 );
    end;
writeln;
close ( f );         { сортировка }
reset ( f );
for I :=fileSize(f)-1 downto 1 do
{ всплывание очередного максимального
элемента на 1-ую  позицию}
    for j:=0  to  I -1 do
       begin
         seek ( f, j );
         Read (f, x, y);
         if x>y then
            begin
              seek(f, j);
              write ( f, y, x );
            end;
       end;
close ( f );
reset ( f );
writeln ('отсортированный файл');
for i:=1 to fileSize ( f ) do
    begin
       read( f, x );
       write ( x:4 );
    end;
readln;
close ( f );
end.
```

Задачи на урок:

1) Написать программу, которая создает типизированный файл, состоящий из случайного числа целых случайных чисел, и затем выводит содержание этого файла на экран компьютера.
2) Используя файл, созданный в предыдущей задаче, найти:

а) самое большое число в этом файле.

б) количество одинаковых чисел в нем.

в) порядковые номера чисел, кратных 3.

1. Что такое типизированные файлы?

2. Чем они отличаются от текстовых файлов?

3. В чем преимущества и недостатки типизированных файлов?

4. Какие вы знаете процедуры и функции для обработки типизированных файлов?

5. Какими командами осуществляется ввод и вывод информации в типизированный файл?

Тип данных Record -Запись

Назначение записей.

Тип данных Запись (Record) используется в тех случаях, когда необходимо обрабатывать структурированные данные, которые описывают несколько различных свойств объекта.

Например, нам надо использовать следующие данные про наших друзей:

1. **Фамилия**
2. **Имя**
3. **Адрес**
4. **Телефон**

Эти данные имеют разный тип. Но из них можно составить структурированный тип данных – запись.

Описание типа данных Record

```
type   имя записи = record
          имя  поля 1 : тип поля1;
                    - - -
          имя поля n : тип поля n ;
  end;
```

Например:

```
Структура   Друзья

Фамилия  :  строка  [ 12 ]
Имя          :  строка  [ 12 ]
Адрес        :  строка  [ 25 ]
Телефон   :  строка  [ 9 ]
```

```
type  friends = record

Fam     :  string [ 12 ];
Name   :  string [ 12 ];
Adress  :  string [ 25 ];
Telef     :  string [ 9 ];
end;
```

Составные имена полей

С полями, входящими в запись, можно выполнять те же действия, что и с обычными переменными соответствующего типа.
Для обращения к полям записи используют составные имена, части которых разделены точкой:

имя записи.имя поля

Friends.Fam - фамилия друга
Friends.Telef - телефон друга

Составные имена могут участвовать в выражениях как обычные переменные:

```
Friends.Telef:='123-45-67';
```

Работа с элементами записи: Использование команды присоединения With

Составные имена довольно громоздки.

Чтобы иметь возможность обращаться непосредственно к самому полю в записи, используют команду **With**

Например:

```
With  имя записи  do
   begin
     действия с полями;
   end;
```

```
With  drug  do
 begin
     writeln ('фамилия');
     readln ( fam );
     writeln ('имя');
     readln ( name);
     tel := '276-90-90';
end;
```

Использование записей в типизированных файлах.

1. В разделе описания типов задать тип данных Record

2. В разделе описания переменных объявить переменную этого типа и файл такого типа.

3. В программе сформировать запись в заданную переменную и потом целиком записать ее в файл.

4. При чтении из файла информацию считывать в эту переменную.

Задание № 1

Написать программу, которая создает типизированный файл содержащий информацию о друзьях, и потом выводит эту информацию на экран из файла. Количество друзей должно запрашиваться в начале программы.

```
Program Druzya;
uses crt;
type friends = record
        Fam, Name : string[ 12 ];
        Date : string[ 8 ];
        Adress : string[ 50 ];
        Tel : string[ 9 ]
   end;
Var  drug : friends;
        f : file of friends;
        i,n : integer;
Begin
Clrscr;
assign (f,  'friends.pas');
{ Создание файла }
Rewrite ( f );
write ('Укажите количество друзей  ');
readln (n);
```

```
for i:=1 to n do
   begin
      with drug do
         begin
           writeln ('Введите фамилию ',i,'-го друга ');
           readln (Fam);
           writeln ('Введите имя ',i,'-го друга ');
           readln (name);
           writeln ('Введите адрес ',i,'-го друга ');
 readln (Adress);
           writeln ('Введите телефон ',i,'-го друга ');
           readln (Tel);
         end;
      write (f, drug);
   end;
close ( f );
```

```
writeln(' Вывод из файла: ');
reset ( f );
for i:=fileSize(f)-1 downto 0 do
    begin
      seek ( f, i);
      read (f, drug);
      with drug do
           begin
             writeln (Fam);
             writeln (Name);
             writeln (Adress);
             writeln (Tel);
           end;
     writeln;
     end;
   end;
close ( f );  readln;  end.
```

Задачи на урок:

1) Написать программу, которая при первом запуске создает файл F записей, а при повторных запусках эта программа дописывает в этот файл следующую информацию: фамилия, имя, год рождения, рост и вес.

2, а) Написать программу, которая считывает файл F созданный в предыдущей программе и выводит на экран данные по фамилиям, первые буквы которых А, Б или В.

2, б) Написать программу, которая сортирует записи в файле F по алфавиту по фамилиям.

2, в) Написать программу, которая в файле находит самого высокого человека и выводит его данные.

Домашнее задание

Составьте программы:

1. Создания типизированного файла с данными про учеников. Поля записей в файле: фамилия, имя, рост и вес.
2. Используя файл из первой программы, определите самого высокого и самого легкого учеников. Выведите их фамилии, имя, вес и рост.