**Типизированные файлы**

**Описание типизированных**[**файлов**](http://pascal.net.ru/file)

В разделе [var](http://pascal.net.ru/var) файловые переменные, предназначенные для работы с *типизированными файлами*, описываются следующим образом:

var <файловая\_перем>: file of <тип\_элементов\_файла>;

Никакая файловая переменная не может быть задана константой.

**Назначение типизированного файла**

С этого момента и до конца раздела под словом «файл» мы будем подразумевать «*бинарный* *типизированный файл*» (разумеется, если специально не оговорено иное).

Команда [Assign](http://pascal.net.ru/Assign)(f, '<имя\_файла>'); служит для установления связи между файловой переменной f и именем того файла, за работу с которым эта переменная будет отвечать.

Строка '<имя\_файла>' может содержать полный путь к файлу. Если путь не указан, файл считается расположенным в той же директории, что и исполняемый модуль программы.

**Открытие и закрытие типизированного файла**

В зависимости от того, какие действия ваша программа собирается производить с открываемым файлом, возможно двоякое его открытие:

[Reset](http://pascal.net.ru/Reset)(f); — открытие файла для считывания из него информации и одновременно для *записи* в него (если такого файла не существует, попытка открытия вызовет ошибку). Эта же команда служит для возвращения указателя на начало файла. Возможность записи в файл зависит от значения переменной [FileMode](http://pascal.net.ru/FileMode) на момент открытия файла;

[Rewrite](http://pascal.net.ru/Rewrite)(f); — открытие файла для *записи* в него информации; если такого файла не существует, он будет создан; если файл с таким именем уже есть, вся содержавшаяся в нём ранее информация исчезнет.

Закрываются *типизированные файлы* процедурой [Close](http://pascal.net.ru/Close)(f), общей для всех типов файлов.

**Считывание из типизированного файла**

Чтение из файла, открытого для считывания, производится с помощью команды [Read](http://pascal.net.ru/Read)(). В скобках сначала указывается имя файловой переменной, а затем — список ввода[1](http://pascal.net.ru/%D0%97%D0%B0%D0%BF%D0%B8%D1%81%D0%B8.+%D0%91%D0%B8%D0%BD%D0%B0%D1%80%D0%BD%D1%8B%D0%B5+%D1%84%D0%B0%D0%B9%D0%BB%D1%8B+(4)#x_footnote_1):

[Read](http://pascal.net.ru/Read)(f, a, b, c); — читать из файла f три однотипные переменные a, b и c.

Вводить из файла можно только переменные, соответствующего объявлению типа, но этот тип данных может быть и структурированным. Cкажем, если вернуться к примеру, приведённому в начале п. «*Типизированные файлы*», то станет очевидным, что использование *типизированного файла* вместо текстового позволит значительно сократить текст программы:

**type** toy = **record**  
    name : **String**[20];  
    price : Real;  
    age : **set** **of** 0..18; {задано границами}  
 **end**;  
  
**var** f : **file** **of** toy;  
  a : **array**[1 .. 100] **of** toy;  
  
**begin**  
  Assign(f, input);  
  FileMode := 0; { 0 = только чтение, 2 = чтение и запись }  
 Reset(f);  
 **for** i :=1 **to** 100 **do**  
   **if** **not** Eof(f) **then**  
     Read(f, a[i]);  
 Close(f);  
 {...}  
**end**.

**Поиск в типизированном файле**

Уже знакомая нам функция [Eof](http://pascal.net.ru/Eof)(f : [File](http://pascal.net.ru/file)) : [Boolean](http://pascal.net.ru/Boolean) сообщает о достигнутом конце файла. Все остальные функции «поиска конца» ([Eoln](http://pascal.net.ru/Eoln)(), [SeekEof](http://pascal.net.ru/SeekEof)() и [SeekEoln](http://pascal.net.ru/SeekEoln)()), свойственные текстовым файлам, нельзя применять к файлам *типизированным*.

Зато существуют специальные подпрограммы, которые позволяют работать с *типизированными файлами* как со структурами прямого доступа:

1. Функция [FilePos](http://pascal.net.ru/FilePos)(f : [Fie](http://pascal.net.ru/file)) : [LongInt](http://pascal.net.ru/Integer) сообщит текущее положение указателя в файле f. Если он указывает на самый конец файла, содержащего N элементов, то эта функция выдаст результат N. Это легко объяснимо: элементы файла нумеруются, начиная с нуля, поэтому последний элемент имеет номер N - 1. А номер N принадлежит, таким образом, «несуществующему» элементу — признаку конца файла.
2. Функция [FileSize](http://pascal.net.ru/FileSize)(f : [File](http://pascal.net.ru/file)) : [LongInt](http://pascal.net.ru/Integer) вычислит длину файла f.
3. Процедура [Seek](http://pascal.net.ru/Seek)(f : [File](http://pascal.net.ru/file); n : [LongInt](http://pascal.net.ru/Integer)) передвинет указатель в файле f на начало *записи* с номером N. Если окажется, что n больше фактической длины файла, то указатель будет передвинут и за реальный конец файла.
4. Процедура [Truncate](http://pascal.net.ru/Truncate)(f : [File](http://pascal.net.ru/file)) обрежет «хвост» файла f: все элементы, начиная с текущего и до конца файла будут из него удалены. На самом же деле произойдет лишь переписывание признака «конец файла» в то место, куда указывал указатель, а физически «отрезанные» значения останутся на прежних местах — просто они станут «бесхозными».

**Запись в типизированный файл**

Сохранять переменные в файл, открытый для *записи*, можно при помощи команды [Write](http://pascal.net.ru/Write)(). Так же, как и в случае считывания, первой указывается файловая переменная, а за ней — список вывода:

[Write](http://pascal.net.ru/Write)(f, a, b, c); — записать в файл f (предварительно открытый для *записи* командами [Rewrite](http://pascal.net.ru/Rewrite)(f) или [Reset](http://pascal.net.ru/Reset)(f)) переменные a,b,c.

Выводить в *типизированный файл* можно только переменные соответствующего описанию типа данных. Неименованные и нетипизированные константы нельзя выводить в *типизированный файл*.

*Типизированные файлы* рассматриваются как структуры одновременно и прямого, и последовательного доступа. Это означает, что *запись* возможна не только в самый конец файла, но и в любой другой его элемент. Записываемое значение заместит предыдущее значение в этом элементе (старое значение будет «затёрто»).

Например, если нужно заместить пятый элемент файла значением, хранящимся в переменной а, то следует написать следующий отрывок программы:

[Seek](http://pascal.net.ru/Seek)(f, 5); {указатель будет установлен на начало 5–го элемента}

[Write](http://pascal.net.ru/Write)(f, a); {указатель будет установлен на начало 6–го элемента}

**Замечание**: Поскольку и чтение из файла, и *запись* в файл сдвигают указатель на следующую позицию, то в случае, когда необходимо сначала прочитать значение, хранящееся в каком–то элементе, а затем вписать на это место новые данные, необходимо производить поиск дважды:

[Seek](http://pascal.net.ru/Seek)(f, 5); {указатель — на начало 5–го элемента}

[Read](http://pascal.net.ru/Read)(f, a); {указатель — на начало 6–го элемента}

[Seek](http://pascal.net.ru/Seek)(f, 5); {указатель — на начало 5–го элемента}

[Write](http://pascal.net.ru/Write)(f, b); {указатель — на начало 6–го элемента}

А что произойдёт, если в файле содержится всего N элементов, а *запись* производится в (N+k)–й? Тогда начало файла останется прежним, затем в файл будет включён весь тот «мусор», что оказался между его концом и записываемой переменной, и, наконец, последним элементом нового файла станет записанное значение.

Литература: <http://pascal.net.ru/Записи.+Бинарные+файлы+(4)>