




Файлы



**Физический файл** – область памяти на внешнем носителе, в которой хранится некоторая информация

**Логический файл** – представление физического файла в программе



Для работы с файлами определяется так называемый **файловый тип**

**Файловый тип** – произвольная последовательность элементов, длина которой заранее не определена, а конкретизируется в процессе выполнения программы

На физическом уровне существуют **файлы**, но для работы с ними на уровне программ определяются т.н. файловые переменные, имеющие файловый тип (сорри за тавтологию).



## Способы доступа к файловой переменной:

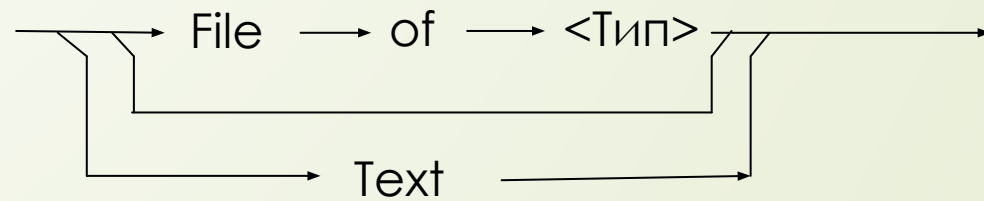
- Последовательный
- Прямой

В первом случае для того, чтобы обратиться к  $n$ -ному элементу файловой переменной, необходимо последовательно обработать  $(n - 1)$  предшествующих ему элементов.

Во втором – возможно прямое обращение к конкретному элементу (почти как в массиве).

## Виды переменных файлового типа:

- Текстовые (Text)
- Файлы с типом (File of <type>)
- Файлы без типа (File)



# Указатель файла

- Связан с каждым открытым файлом
- Другие названия: окно файла, текущая позиция файла
- Определяет позицию доступа (элемент файла, с которым выполняется следующая операция ввода/вывода)
- Конец файла никак не помечается!



## Операции над переменными файлового типа:

- Их нет
- Их нет
- Их нет

Работа с файлами осуществляется с помощью **процедур и функций ввода-вывода**



File

Бинарный файл





```
procedure AssignFile (var F : File; FileName : string);
```


Связывает файловую переменную F с файлом FileName

```
function FileExists (FileName : string) : boolean;
```

Проверяет, существует ли файл

```
AssignFile(F, 'C:\Мои документы\Мои рисунки\FileName.ini');  
AssignFile(F, 'FileName.ini');
```

```
If FileExists('FileName.ini') then  
begin  
    bla-bla-bla;  
end;
```



```
procedure Reset (var F : File; [RecSize : word]);
```

открывает файл для чтения

```
procedure Rewrite (var F : File; [RecSize : word]);
```

открывает файл для записи

- Две последние процедуры связывают файловую переменную F с файлом FileName
- RecSize – необязательный параметр, задаёт размер элемента файла в байтах
- Если файл до этого был открыт, то он закроется и откроется заново в заданном режиме



```
procedure CloseFile (var F : File);
```

Закрывает файл

```
function Eof (var F : File) : boolean;
```

Проверяет, является ли текущая позиция концом файла

```
while (not EOF(F)) do
```

```
begin
```

```
    ...
```

```
end;
```



procedure **Read** (*F : File, V1..Vn*);

Считывает компоненты файла в соответствующие переменные

procedure **Write** (*F : File, V1..Vn*);

Записывает в файл компоненты из соответствующих переменных

Var

F : file of integer;

value : integer;

begin

...

read(F, value);

write(F, value);

end;



```
procedure BlockRead (F : File, var Buf;  
count : integer [; var Done : integer]);
```

Считывает не более count компонентов в переменную buf из файла F. Необязательный параметр Done – реально прочитанное количество.

```
procedure BlockWrite ( - // - );
```

Записывает не более count компонентов в файл F из переменной buf. Необязательный параметр Done – реально записанное количество.



procedure **Seek** (*var F : File; N : longInt*);

Перемещает указатель файла F в позицию N

function **FilePos** (*var F : File*) : LongInt;

Возвращает текущее положение указателя в файле F

function **FileSize** (*var F : File*) : integer;

Возвращает размер файла в компонентах

```
const
    FILE_NAME = 'source.txt'
var
    F : file of integer;
    value : integer;
    currPosition : longInt;
begin
    AssignFile (F, FILE_NAME);
    if fileExists (FILE_NAME) then
        begin
            Reset (F)
            while not EOF(F) do
                begin
                    currPosition := filePos (F);
                    read (F, value);
                    seek (F, currPosition)
                    value = value * (-1);
                    write (F, value);
                end;
            end;
            CloseFile (F);
        end;
end;
```

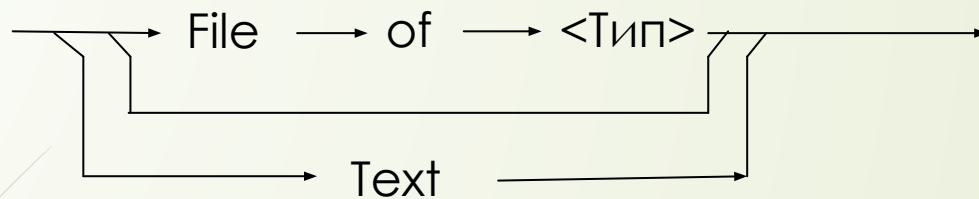
- В файле хранятся 4-байтовые числа.
- Программа меняет знак каждого на противоположный и перезаписывает число в файл



Text

Текстовый файл





- Файл считается последовательностью символов, интерпретируемых как текст
- Не эквивалентно file of char (также, как string не эквивалентно array of char)




## **Доступно:**

- AssignFile
- Reset (запись невозможна, только чтение)
- Rewrite (только запись)

## **Появляется:**

procedure **Append** (*var F : Text*);

- Открывает существующий файл для добавления
- Помещает указатель на конец файла

- 
- Read (Reset only)
  - Write (Rewrite only)
  
  - Eof – то же самое, что и в бинарном
  - Eoln ([var F : Text]) – определяет, является ли концом строки
  
  - Flush() – записывает данные, которые остаются в буфере в файл



Var

**Input** : Text;

**Output** : Text;

Файлы объявлены в стандартной библиотеке и связаны непосредственно с консолью