Уроки 1 (4 часа)

Тема урока. Типизированные файлы.

Цель урока. Познакомить учащихся с принципами работы типизированных файлов Turbo Pascal.

План урока.

1. Повторение.
2. Типизированные файлы.
3. Прямой доступ к файлам.
4. Перемещение по файлу.
5. Основные операции с файлами.
6. Примеры программ.

Повторение

1. Что такое файл? Для каких целей используется файл?
2. Как его описать?
3. Какие особенности текстового файла?
4. Зачем используется специальная файловая переменная?
5. Каковы требования к именам файлов?
6. Какие действия можно производить с файлами?
7. Что общего у процедуры Reset и Rewrite и чем они отличаются?
8. Зачем применяется процедура Close?

Типизированные файлы

Типизированный файл – последовательность записей определенной структуры. В качестве записи выступает компонента тип, которой задается при определении файловой переменной.

С каждым типизированным файлом связан *файловый указатель*, текущим значением которого является номер компоненты.
При вводе или выводе очередной компоненты его значение увеличивается на единицу.

Var
File_T : file of <*тип*>**;**

<*тип*> - тип компоненты (записи) файла; может быть любого типа, кроме файлового.

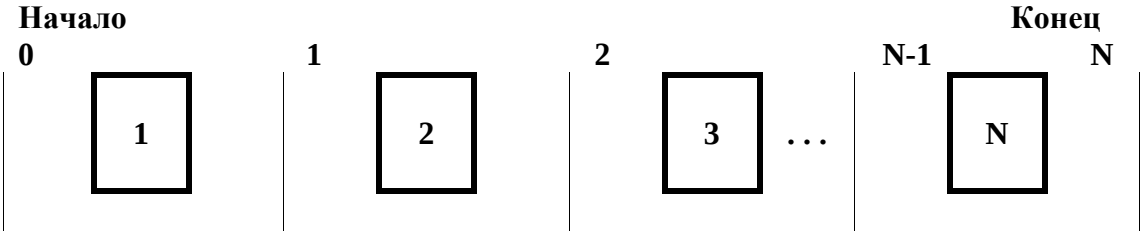
Прямой доступ к файлам

Прямой доступ - возможность установить *указатель* внутри файла на интересующую нас запись.
Это возможно, так как длина любого компонента типизированного файла строго постоянна.

- Файл заполняется последовательно от начала.
- Структура файла всегда линейна: запись следует за записью.
- В файле не может быть «дыр».
- Добавить запись можно лишь к конец.

Для индексации структуры файла используется нумерация его записей. Но нумеруются не записи, а границы между ними.

Самая первая граница (в начале файла) имеет номер 0.



Перемещение по файлу

Настройка на требуемую компоненту

Seek (<*имя файловой переменной*>, <*номер компоненты*>**)**

Процедура смещает файловый указатель на компоненту с указанным номером.

Удаление части файла

Truncate (<*имя файловой переменной*>**)**

Процедура удаляет часть файла, начиная с текущей позиции и до его конца.

Номер текущей компоненты файла

FilePos (<*имя файловой переменной*>**)**

Функция возвращает номер текущей компоненты файла, с которым связана файловая переменная. Возвращаемое значение имеет тип *LongInt.*

Текущий размер файла

FileSize (<*имя файловой переменной*>**)**

Функция возвращает текущий размер файла, с которым связана файловая переменная. Возвращаемое значение имеет тип *LongInt.* Для пустого файла возвращает 0.

Примеры

Seek (F, FilePos (F) + 1)	пропуск одного элемента
Seek (F, 0)	установка указателя на начало файла
Seek (F, FileSize (F))	установка текущего указателя непосредственно за последним элементом файла; это может служить исходной позицией для добавления элементов в «хвоста» файла

Открытие файлов

Типизированные файлы открываются одновременно на запись и на считывание.

1. Открытие файла уже существующего файла (если его нет, то возникнет сообщение об ошибке) для чтения:

Reset (< *имя файловой переменной* >**)**

2. Открытие файла, который может не существовать; в этом случае процедура создает заданный файл. Если же файл существует, происходит очищение его:

Rewrite (< *имя файловой переменной* >**)**

Пример

Открыть файл с помощью процедуры *Reset* с целью добавления записей.

```
Var     F       : File of LongInt;
        Code    : Word;
. . .
Begin
        Assign ( F,'C:\Myfile.dat');
        {$I-}
        Reset ( F );
        {$I+}
        Code := IOResult;
        If Code <> 0 Then
                Begin
                        Write ('Ошибка при открытии файла' );
                Case Code of
                2: Write ('Файл не найден');
                3: Write ('Маршрут не найден');
                4: Write ('Слишком много открытых файлов');
                12: Write ('Некорректный код доступа к файлу');
                Else
                . . .

                End
        Else { Старый файл открыт на считывание и запись}
                Begin
                        Seek ( F, FileSize ( F ) );
                . . .
                End;
                . . .
End.
```

Пример

Создание типизированного файла, содержащего записи с поляли.

```
Type Stud = Record
                        Name            : String [25];
                        Age             : Integer;
                        Sex             : Char;
                End;
Var
        File_Name       : String;
        Student         : Stud;
        Ch              : Char;
        F               : File of Stud;
Begin
        Write ('Введите имя файла:');
        ReadLn ( File_Name);
        Assign ( F, File_Name);
        Rewrite ( F );
        Repeat
                With Student do
```

```
                    Begin
                    Write (‘Введите фамилию:’); ReadLn (Name);
                    Write (‘Введите возраст:’); ReadLn (Age);
                    Write (‘Введите пол:’); ReadLn (Sex)
                    End;
                Write ( F, Student);
                Write (‘Будете продолжать работу?’); ReadLn (Ch)
        Until (Ch <> ‘Y’) and (Ch <> ‘y’);
        Close ( F );
End.
```

Пример

По номеру записи выдает информацию о студенте.

```
Type Stud = Record
                    Name        : String [25];
                    Age         : Integer;
                    Sex         : Char;
                End;
Var     F           : File of Stud;
        Code        : Word;
        Student     : Stud;
        Size, Num   : Integer;
Begin
        Assign ( F,’C:\Myfile.dat’);
        {$I-}   Reset ( F );    {$I+}
        Code := IOResult;
        If Code <> 0 Then
                Begin
                    Write (‘Ошибка при открытии файла’ );
                Case Code of
                2: Write (‘Файл не найден’);
. . .
                End
        Else { Старый файл открыт на считывание и запись}
                Begin
Size := FileSize ( F ); WriteLn (‘Число записей в файле -’, Size);
                    Repeat
        WriteLn (‘Введите номер записи:’); eadLn (Num);
                    If  ( Num < 1) or ( Num > Size ) Then
                            WriteLn (‘’, Num:5, ‘’)
                    Else    Begin
                    Seek ( F, Num - 1); Read ( F, Student );
                    With Student do Begin
                    Write (‘Фамилия:’, Name);
                    Write (‘Возраст:’, Age);
                    Write (‘Пол:’, Sex)
                            End
                End;
```

```
            Write (‘Будете продолжать работу?’); ReadLn (Ch)
        Until (Ch <> ‘Y’) and (Ch <> ‘y’);
        Close ( F );
End.
```

Задание

1. Познакомиться с программой обслуживания телефонных справочников (создание, изменение, добавление, поиск номера телефона абонента).
2. Составить программу, которая создает файл данных о химических элементах, отображая следующую информацию: название, символическое обозначение, массу атома, заряд атомного ядра, перечень основных химических свойств. Программа должна выполнять вывод данных о химическом элементе по указанному символическому обозначению, находить элемент с самой большой массой, с самым маленьким зарядом ядра (на оценку 4)
3. Дополнение для задачи п.2: информация вводится из текстового файла (на оценку 5).