Файловый тип данных.

Операции для работы с файлами

Тема: Операции для работы с файлами

План урока:

Основные сведения о файлах.

Виды файлов.

Операции над файлами.

Домашнее задание.

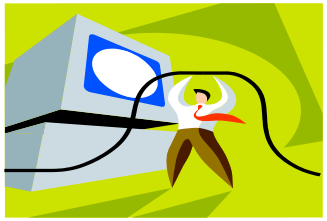
- Ранее мы рассматривали задачи, в которых во время выполнения программ данные поступали с клавиатуры, а результаты выводились на экран дисплея. Поэтому ни исходные данные, ни результаты не сохранялись. Всякий раз при выполнении одной и той же программы, особенно во время ее отладки, приходится заново вводить исходные данные. А если их очень много? В языке Qbasic и Turbo Pascal 7.0 есть возможность записать их на диск. Для этого необходимо оформить исходные данные и результаты в виде файлов, которые хранятся на диске точно так же, как и программа.

- У понятия файл есть две стороны.
- С одной стороны, ***файл — это область памяти на внешнем носителе, в которой хранится некоторая информация.*** В него можно поместить данные или извлечь их оттуда***. Файл*** в таком понимании ***называют физическим файлом,*** то есть существующим физически на некотором материальном носителе информации.
- С другой стороны, файл – это одна из многих структур данных, используемых в программировании. **Файл** в таком понимании **называют логическим файлом,** то есть существующим только в нашем логическом представлении при написании программы.

- **Структура физического файла** представляет собой **простую последовательность байт** памяти носителя информации – ЖМД или ГМД.

Байт	**Байт**	**Байт**	**…**	**Байт**	**Байт**	**Байт**

- **Структура логического файла –** это способ восприятия файла в программе. Образно говоря, это “шаблон” (“окно”), через который мы смотрим на физическую структуру файла. В языках программирования таким “шаблонам” соответствуют типы данных, допустимые в качестве компонент файлов. Например:



FILE OF BYTE

Байт	Байт	Байт	…	Байт	Байт	EOF

FILE OF CHAR

Код Символа	Код Символа	Код Символа	…	Код Символа	Код Символа	EOF

FILE OF INTEGER

Целое со знаком	Целое со знаком	…	Целое со знаком	EOF

Логическая структура файла в принципе очень похожа на структуру массива. Различия между массивом и файлом:

- У массива количество элементов фиксируется в момент распределения памяти, и он целиком располагается в оперативной памяти. Нумерация элементов массива выполняется соответственно нижней и верхней границам при его объявлении.

- У файла количество элементов в процессе работы может изменяться, и он располагается на внешних носителях информации. Нумерация элементов файла выполняется слева направо начиная от нуля (кроме текстового в Turbo Pascal 7.0, а в Qbasic с 1). Количество элементов в каждый момент времени неизвестно. Зато известно, что в конце файла располагается специальный символ конца файла, определяемый функцией **EOF.**

КЛАССИФИКАЦИЯ ФАЙЛОВ

Файлы классифицируются по двум признакам:

- **По методу доступа** – п**оследовательный, прямой** доступ (Turbo Pascal 7.0, в Qbasic)
- **По типу (логической структуре**) (Turbo Pascal 7.0) – типизированные, текстовые, нетипизированные.

- Файл последовательного доступа можно сравнить с довольно длинной магнитофонной лентой в кассете, на которой записаны песни (или какая-то информация). Для того, чтобы найти конкретную песню, надо перемотать кассету на начало и прослушивать песню за песней до тех пор, пока не будет найдена нужная.
- Зачем нужны файлы? Дело в том, что количество элементов файла может быть любым: число компонент файла может изменяться (увеличиваться или уменьшаться), то есть заранее не фиксируется. Поэтому в нем можно хранить достаточно большое количество данных. После каждого элемента автоматически ставится признак конца элемента, а в конце файла ставится признак конца файла.

- Переменные файлового типа могут быть описаны в программе либо явно в разделе переменных Var, либо с использованием раздела типов Type.
- Объявление файлов в разделе переменных имеет вид:
- Var <имя файла>: File Of <базовый тип элементов>;
- Например,
- Var Ft: File Of Integer; {файл целых данных}
- M: File Of Char; {файл символьных данных}

- Описание файлов с помощью раздела типов имеет такой вид:
- Type Fil1 = File Of Integer;
- Fil2 = File Of Char;
- Var F1: Fil1;
- F2: Fil2;
- Элементами файла F1 могут быть целые числа, а F2 — символы.
- У таких файлов указывается тип их элементов, они называются типизированными. Все компоненты имеют общее имя, а каждый еще и имеет свой номер. Начальный элемент имеет нулевой номер.

ОПЕРАЦИИ НАД ФАЙЛАМИ В QBASIC

Любые действия с файлами, выполняемыми в программе, включают в себя следующие обязательные шаги:

- открытие файла;
- чтение и запись обрабатываемых данных;
- закрытие файлов.

Можно открыть несколько файлов одновременно, чтобы читать информацию из одного файла, обрабатывать ее и затем сохранять в другом. Но нельзя использовать открытый файл и для чтения, и для записи одновременно, потому что эти действия связаны с различными режимами открытия файла.

ОТКРЫТИЕ ФАЙЛА

Для открытия файла используется оператор OPEN, имеющий следующий общий формат:

OPEN имя файла$ FOR режим AS #номер файла%

файл$ - имя файла

режим - режим работы с данным файлом

- необязательный знак, предваряющий номер файла

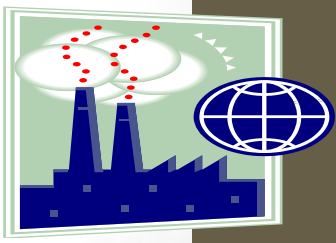
номер файла% - целое число от 1 до 255.

Например:

OPEN "имя" FOR OUTPUT AS #1

OPEN "имя" FOR INPUT AS #2

OPEN "имя" FOR APPEND AS #5



ИМЯ ФАЙЛА

Имя файла должно включать не только непосредственно название, но и наименование дискового устройства и директории, где находится или будет создан файл.

Например

"a:\fiele.bas"

"c:\DOS\fiele.bas"

"c:\DOS\QRUS\fiele.bas"

Простейший способ работы с файлами - сделать текущей ту поддиректорию, в которой находятся интересующие нас файлы, а затем просто указывать их имена.

В качестве параметра ФАЙЛ$ можно использовать переменную текстового типа, которой присвоено имя файла.

Например:

```
a$="имя файла"
OPEN a$ FOR INPUT AS #1
```

РЕЖИМЫ РАБОТЫ С ФАЙЛАМИ

Параметр «режим» определяет режим работы с файлом:. указывает, как вы собираетесь этот файл использовать: для чтения или записи в него информации. Этот параметр может иметь одно из следующих значений:

OUTPUT - открывает новый файл для записи в него информации, (если вы открываете в этом режиме уже существующий файл, его содержимое будет стерто)

APPEND - открывает существующий файл для добавления в него новой информации.

В таком режиме новая информация всегда помещается в конец файла, после последней записи. Если Вы указали имя файла, которого еще не существует, то будет открыт новый файл, как это делается в режиме OUTPUT.

INPUT - открывает существующий файл для чтения хранимой в нем информации.

НОМЕР ФАЙЛА

- Последний параметр в операторе OPEN присваивает открываемому файлу определенный номер для более удобного использования в программе, например:
- OPEN "имя" FOR OUTPUT AS #3
- Этот оператор присваивает файлу номер 3. Теперь данный номер нельзя будет присвоить никакому другому файлу, пока то будет открыт.
- Номер файла - любое число от 1 до 255.
- Как много файлов можно открыть? Это зависит от операционной системы, в которой задается максимальное число одновременно открываемых файлов. Для избежания хаоса, не открывайте слишком много файлов одновременно. Закончили работу с файлом - закройте его.

ЗАКРЫТИЕ ФАЙЛА

- Для закрытия файла используется простой оператор CLOSE [#номер файла]
- Если вы указываете номер файла, то будет закрыт именно этот файл.
- Оператор CLOSE без параметра закрывает все файлы, открытые в данный момент в программе.

ОПЕРАЦИИ НАД ФАЙЛАМИ В TURBO PASCAL 7.0.

- **1. Связь переменной файлового типа с конкретным внешним файлом**
- В Turbo Pascal 7.0 файл - некоторая переменная, как и любая другая переменная, поэтому ему можно присвоить имя. С другой стороны, существует операционная система, которая в свою очередь использует имена файлов, например A: MM.Dat, B: Test.Pas. Для установления связи между переменной-файлом в Turbo Pascal 7.0 и именем файла, присваиваемого операционной системой, имеется стандартная процедура ***Assign*.**
- Общий вид:
- **Assign(<имя переменной-файла>,'<имя внешнего файла>');**
- Процедура устанавливает соответствие между файловой переменной и внешним файлом. Например,
- **Assign(F1,'a:int.dat');**
- Такое соответствие обозначает, что все операции, выполняемые над переменной F1, будут выполняться над файлом, хранящимся на диске A и имеющим имя 'Int.dat'.

2. **Открытие, чтение, запись**

Перед выполнением каких-либо операций чтения и записи в файлах, эти файлы должны быть открыты.

Открытие файлов выполняется процедурами RESET(f) и REWRITE(f), а закрытие – процедурой CLOSE(f).

Процедура RESET(f) открывает существующий физический файл, который был связан с файловой переменной f.

Если f – текстовый файл, то он доступен только для чтения при последовательном доступе к элементам.

Если f – типизированный файл, то он будет открыт и для чтения, и для записи как при последовательном доступе, так и при прямом. При открытии указатель текущей позиции файла устанавливается в его начало.

Процедура REWRITE(f) создает новый физический файл, имя которого связано с файловой переменной f.

Если такой файл уже существует, то он удаляется, и на этом месте создается новый пустой файл. При открытии указатель текущей позиции в файле устанавливается в его начало.

ОПЕРАЦИИ НАД ФАЙЛАМИ В TURBO PASCAL 7.0.

Под чтением файла понимается ввод данных из внешнего файла, находящегося на диске, в оперативную память машины. Данные внешнего файла становятся доступными программе.

Чтение из типизированных файлов выполняется только процедурой ***READ*** **(<имя переменной-файла>, <элемент>), а запись только процедурой WRITE(<имя переменной-файла>, <элемент >).** Элемент должен быть того же типа, что и компоненты файла. Выполняя запись в файл следует помнить, что при записи каждой переменной указатель текущей позиции в файле, так же как и при чтении, перемещается на следующий элемент, если указатель текущей позиции файла находится за последним элементом, то есть в конце файла, то при выполнении процедуры WRITE файл расширяется.

Если оператор ввода имеет вид **Read** (<элемент>), то данные вводятся с клавиатуры, а если **Read** (<имя переменной - файла>, <элемент>), то данные вводятся из файла, хранящегося на диске.

Если оператор вывода имеет вид **Write** (<элемент>), то данные выводятся на экран дисплея, а если **Write** (<имя переменная-файла>,<элемент>), то данные записываются в файл, который хранится на диске. После работы с файлом его закрытие обязательно.

3. Закрытие файла

Процедура **Close (<имя переменной-файла>).**

Признак конца файла

- **4.** Так как, по определению, число элементов файла не задается заранее, то в языке Turbo Pascal 7.0 введен признак конца файла. Это логическая функция: EOF(<имя переменной-файла>). Она используется для определения, достигнут ли конец файла или еще нет (принимает истинное значение (TRUE), если достигнут конец файла, и ложное (FALSE) — в противном случае). Для определения конца файла используется оператор цикла, например, (пока не достигнут конец файла...):

- **While Not EOF**(<имя переменной-файла>) **Do**...

Пример.

Прочитаем файл целых чисел и выведем их на экран:

```
Assign(F1, 'a'.int.dat'); {связываем с внешним файлом}
ReSet(F1);  {открываем его для чтения}
While Not EOF (F1) Do {пока не достигнут конец файла F1}
Begin
Read(F1,n);  {считываем очередное число}
Write(n, ' ');   {выводим его на экран}
End;
Close(F1);     {закрываем файл}
End.
```

Примечания:

1. После выполнения процедур открытия файла для чтения или для записи (ReSet или ReWrite) текущий указатель "смотрит" на первый элемент, например, ReSet(Fl):

После выполнения процедуры Read, например Read(F1,x), если указатель стоял на k-м элементе, то после выполнения он будет стоять на следующем, а значение переменной x будет равно **k**-му элементу.

Текстовые файлы

- **Текстовые файлы — это файлы, содержащие символы, разделенные на строки. Причем в конце каждой строки стоит признак конца строки. Для их описания используется служебное слово Text.**

  Var A: Text.

- Надо заметить, что этот тип не равен типу File Of Char, который является типизированным.
- Отличия текстовых файлов от типизированных:

1. Элементами этих переменных являются символы, и они имеют строковую структуру.
2. При чтении и записи числа преобразуются автоматически.
3. Текстовые файлы не имеют прямого доступа.
4. Наличия признаков конца строки.
5. К ним применима процедура Append (<имя переменной текстового файла>). Она открывает внешний файл, с которым связана данная переменная, текущий указатель помещает в конец для добавления новой информации

Нетипизированные файлы

- При объявлении нетипизированного файла указывается только ключевое слово, например:

  Var f:file

- Нетипизированные файловые переменные предназначены для низкоуровневой работы с файлами.
- С их помощью можно обратиться к файлу любого типа и логической структуры. Необходимо только учитывать, что нетипизированные файлы не имеет жестко установленной единицы чтения/записи, как типизированные файлы.

В нетипизированных файлах за одно обращение считывается/записывается число байт, приблизительно равное величине буфера ввода/вывода, что способствует увеличение скорости работы с файлами. В качестве буфера ввода\вывода нетипизированных файлов может выступать любая переменная.

Для работы с нетипизированными файлами можно применять почти все процедуры и функции, что и для типизированных. Исключение составляет то, что вместо процедур **READ** и **WRITE** используются процедуры **BlockRead** и **BlockWrite**, а процедуры Reset и Rewrite могут имеет второй параметр типа Word, который определяет размер записи, используемый при передачи данных. Если этот параметр опущен, то размер записи принимается по умолчанию равным 128 байтам.

Обработка текстовых файлов

- Для текстовых файлов применимы те же процедуры и функции, что и для обычных файлов. Сначала они должны быть связаны с конкретным внешним файлом при помощи процедуры Assign. Затем они могут быть открыты для чтения процедурой ReSet или для записи процедурой ReWrite.
- Кроме этих процедур так же применяется процедура Read, которая считывает очередной элемент строки, но если необходимо прочитать и перейти на следующую строку, то применяется процедура *Readln* (<имя переменной текстового файла>,<элемент>). Если просто перейти к следующей строке, то используется процедура *Readln* (<имя переменной текстового файла>), которая переносит текущий указатель на первый элемент следующей строки.

- Процедура Write записывает элемент в текущую строку.
- Если надо записать и перейти к следующей строке, используется процедура Writeln (<имя переменной текстового файла>, <элемент>).
- Если требуется перейти для записи на новую строку, то применяется процедура Writeln(<имя переменной текстового файла>), которая ставит в конце данной строки признак ее конца и перемещает текущий указатель на начало следующей строки.

Т. к. в разных строках может быть разное количество элементов, то используется логическая функция **Eoln**(<имя переменной текстового файла>), которая определяет признак конца данной строки, если достигнут конец строки, то имеет истинное значение, а если нет — ложное. Ее так же, как и функцию определения конца файла, лучше всего применять в цикле, например, (пока не достигнут конец строки...):

While Not Eoln(Fl) Do...

К текстовым файлам применима процедура **Append**(<имя переменной текстового файла>). Она открывает внешний файл, с которым связана данная переменная, текущий указатель помещает в конец для добавления новой информации.

	program fil86;
	uses crt;
	var f:file of integer;
DEFINT M-N,I,S	*n,m,i,S:integer;*
DEFSTR A	*a:string*
	begin
CLS	*clrscr;*
A=”D:\UROKI\F86.TXT”	*a:= 'd:\uroki\f86.txt'*
Открываем файл А для записи как 2.	*Связываем переменную f с внешним файлом и открываем файл для записи.*
OPEN A FOR OUTPUT AS #2	*assign(f,a); rewrite(f);*
RANDOMIZE TIMER	*randomize;*

Вводим с клавиатуры N – число элементов в файле.

INPUT”N=”:N	*write('n=');readln(n);*
FOR I=1 TO N	*for i:=1 to n do begin*

Получаем случайные числа

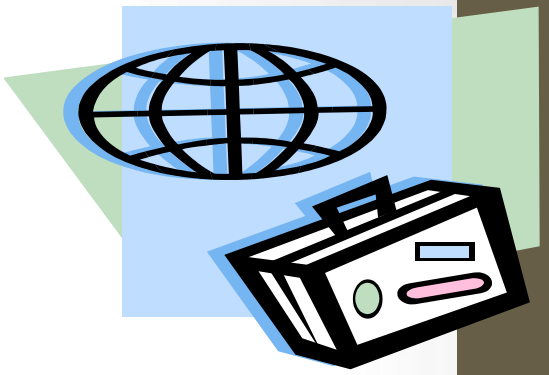
M=RND*25+65	*m:=trunc(random(25))+65;*
	Записываем их в файл.
WRITE #2, M	*write(f,m);*
	Одновременно выводим их на экран.
PRINT M; " ";	*write(m,' ');*
NEXT	*end;*
	Закрываем файл.
CLOSE #2	*close(f);*
PRINT	*writeln;*

Продолжение программы

Открываем файл А для чтения как 2	*Связываем переменную f с внешним файлом и открываем файл для чтения*
OPEN A FOR INPUT AS #2	*assign(f,a);reset(f);*
Пока не достигнут конец файла	
WHILE NOT EOF (2)	*while not eof(f) do begin*
Считываем очередное число.	
INPUT#2,N	*read(f,n);*
Выводим его на печать	
PRINT N;" ";	*write(n,' ');*
Находим сумму элементов файла.	
S=S+N	***s:=s+n***
WEND	*end;*
Закрываем файл.	
CLOSE	*close(f);*
PRINT"S=";S	***writeln('s=',s)***
	readln;
END	*end.*

QB TP

Домашнее задание



Ответить на вопросы.

1. Формат открытия файла
2. Какие операции возможны над файлами последовательного доступа.
3. Роль оператора CLOSE?
4. Что показывает функция EOF?
5. В чем различие между структурой логического и структурой физического файла.
6. В чем сходство и различие между массивом и файлом?

Урок 28

Тема: Операции для работы с файлами.

План урока:

- Проверка домашнего задания.
- Заполнение файлов
- Чтение из файлов.
- Нахождение максимальных элементов.
- Домашнее задание.

Проверка домашнего задания

Ответить на вопросы.

1. Формат открытия файла.
2. Какие операции возможны над файлами последовательного доступа.
3. Роль оператора CLOSE?
4. Что показывает функция EOF?
5. В чем различие между структурой логического и структурой физического файла.
6. В чем сходство и различие между массивом и файлом?

*Создать файл, элементы которого вычисляются по формуле m:=i*i+4*i; i=1..N Определить число элементов файла, делящихся на 4.*

	program fil87;
	uses crt;
	var f:file of integer;
DEFINT N,M,I,K	*n,m,i,k:integer;*
DEFSTR A	*a:string;*
	begin
CLS	*clrscr;*
A=”D:\UROKI\file\FIL87.TXT	*a:='d:\uroki\fiele\fil87.txt';*

Открываем файл А для записи как 1	***Связываем переменную f с внешним файлом и открываем файл для записи.***
OPEN A FOR OUTPUT AS #1	***assign(f,a);rewrite(f);***
INPUT"N=";N	***write('n=');readln(n);***
FOR I=1 TO N	***for i:=1 to n do begin***
M=I*I+4*I	***m:=i*i+4*i;***
PRINT M;" ";	***write(m,' ');***
	Записываем в файл.
WRITE #1, M	***write(f,m);***
NEXT	***end;***
CLOSE	***close(f);***
PRINT	***writeln;***

Продолжение программы

Открываем файл А для чтения как 1

Связываем переменную f с внешним файлом и открываем файл для чтения

OPEN A FOR INPUT AS #1

assign(f,a);reset(f);

Пока не достигнут конец файла.

WHILE NOT EOF(1)

while not eof(f) do Begin

Считываем очередное число.

INPUT #1, M

read(f,m);

Проверяем его на кратность 4 и считаем их количество.

IF M MOD 4 =0 THEN K=K+1

if m mod 4 =0 then k:=k+1;

Выводим его на печать

PRINT M;” “;

write(m,' ');

WEND

end;

CLOSE

close(f);

Печатаем число элементов кратных 4.

PRINT”K=”;K

writeln('кратных четырем =',k); readln;

END

end.

QB TP

Самостоятельно решите следующие задания:

А) определить число четных элементов в файле;

Б) определить число нечетных элементов в файле;

В) определить число элементов в файле, которые больше 32;

Г) определить число элементов в файле, которые меньше 32.

Создать файл F, элементы которого вычисляются по формуле m:=3*i*i-2*i+6. Получить два новых файла G, H, в один из них поместить четные элементы файла F, в другой нечетные. Убедитесь, что у вас это получилось.

```
DEFINT M-N,I,K
DEFSTR A,G,H

CLS
A=”FIL88.DAN”
```

```
program fil88;
uses crt;
var f:file of integer;
fg:file of integer;
fh:file of integer;
n,m,i,k:integer;
a,g,h:string;
begin
clrscr;
a:='fil88.dan';
```

Открываем файл А для записи. как 1	*Связываем переменную f с внешним файлом и открываем файл для записи.*
OPEN A FOR OUTPUT AS #1	*assign(f,a);rewrite(f);*
INPUT"N=";N	*write('n=');readln(n);*
FOR I=1 TO N	*for i:=1 to n do begin*
M=3*I*I-2*I+6	*m:=3*i*i-2*i+6;*
PRINT M;" ";	*write(m,' ');*
Записываем в файл.	
WRITE #1,M	*write(f,m);*
NEXT	*end;*
Закрываем файл.	
CLOSE	*close(f);*
PRINT	*writeln;*

G=”FIL88G.DAN”	*g:='fil88g.dan';*
H=”FIL88H.DAN”	*h:='fil88h.dan';*
Открываем файлы G, H для записи как 2,3	*Связываем переменные fg и fh с внешним файлом и открываем файлы для записи.*
OPEN G FOR OUTPUT AS #2	*assign(fg,g); rewrite(fg);*
OPEN H FOR OUTPUT AS#3	*assign(fh,h); rewrite(fh);*
Открываем файл A для чтения как 1.	*Связываем переменную f с внешним файлом и открываем файл для чтения*
OPEN A FOR INPUT AS #1	*assign(f,a);reset(f);*

Пока не достигнут конец файла.

WHILE NOT EOF(1)	***while not eof(f) do Begin***

Считываем очередное число.

INPUT#1,N	***read(f,n);***
PRINT N;" ";	***write(n,' ');***

Проверяем его на четность.

IF N MOD 2 =0 THEN WRITE	***if n mod 2 =0 then write (fg,n)***
#2,N ELSE PRINT #3,N	***else write(fh,n);***

Четные записываем в файл G, а нечетные в файл H.

NEXT	***end;***
CLOSE	***close(f);close(fg);close(fh);***
PRINT	***writeln;***

Открываем файл G для чтения как 2.	***Связываем переменную fg с внешним файлом G и открываем файл для чтения***
OPEN G FOR INPUT AS #2	**assign(fg,g);reset(fg);**
WHILE NOT EOF(2)	**while not eof(fg) do Begin**

Считываем очередное число и выводим на экран.

INPUT #2,N	read(fg,n);
PRINT N;” “;	write(n,' ');
NEXT	end;
CLOSE	close(fg);
PRINT	writeln;

Открываем файл H для чтения как 3.

Связываем переменную fh с внешним файлом H и открываем файл для чтения

OPEN H FOR INPUT AS #3
WHILE NOT EOF(3)

assign(fh,h);reset(fh);
while not eof(fh) do
Begin

Считываем очередное число.

INPUT#3,N
PRINT N;” “;
NEXT
CLOSE

END

read(fh,n);
write(n,' ');
end;
close(fh);
readln;
end.

QB TP

В предыдущем примере, среди нечетных элементов в файле H, найдите наибольший.

Так как мы определили тип элементов в файлах как INTEGER, следовательно наименьшее значение, которое может находиться в файлах это **- 32768.** Зададим начальное значение **MAX =-32768** перед считыванием из файла H и сравнивая с каждым элементом найдем максимальное значение.

Самостоятельно в предыдущую программу добавьте строки:

DEFINT MAX	***max:integer;***
MAX =-32768	***max:=-32768;***
IF MAX<= N THEN MAX=N	***if max<=N then max:=n;***
PRINT”MAX=”;MAX	***writeln(‘max=’,max);***

Домашнее задание.

Найдите первый (последний) максимальный элемент, среди элементов кратных трем.

Урок 29

Тема: языки программирования Qbasic и Turbo Pascal 7.0. Операции для работы с файлами.

План урока:

Проверка домашнего задания.

Использование процедур и файлов.

Домашнее задание.

Проверка домашнего задания.

Найдите первый (последний) максимальный элемент, среди элементов кратных трем.

Самостоятельно в предыдущую программу добавьте строки:

DEFINT MAX	***max:integer;***
MAX =-32768	***MAX:=-32768;***
IF MAX< N AND N MOD 3 =0	***if (max<N) and***
THEN MAX=N	***(n mod 3 =0 then max:=n;***
PRINT”MAX=”;MAX	***writeln(‘max=’,max);***

Напишите программу, которая обрабатывает файл, содержащий имена друзей, их адреса, номера телефонов и другую дополнительную информацию.

Программа должна предоставить пользователю следующие возможности:

- создание файла, содержащего эту информацию.
- просмотр всей информации, содержащейся в файле
- поиск информации об отдельных людях
- добавление в файл новых данных
- уничтожение записей
- изменение информации

Все эти действия независимы друг от друга и могут быть оформлены в виде отдельных подпрограмм (процедур). Выбор режима работы, т.е. вызов конкретной процедуры, оставим в основной программе.

Составим процедуру создания файла, содержащего информацию.

SUB REWRITE (A AS STRING)	***procedure rewrit(a:string);***
DEFSTR A,D,I,T,W	***var w, im, adr, tel, dop:string;***
‘	***begin***
Открываем файл А для записи.	***Связываем переменную f с***
как 2	***внешним файлом и открываем файл для записи.***
OPEN A FOR OUTPUT AS #2	***assign(f,a);rewrite(f);***
DO	***repeat***
INPUT” Для выхода введите 0”;W	***write('Для выхода введите 0 ');readln(w);***
IF W=”0” THEN EXIT DO	***if w='0' then break;***

Вводим информацию.

INPUT” Введите имя”;IM	***write('Введите имя '); readln(im);***
INPUT” Введите адрес ”;ADR	***write('Введите адрес '); readln(adr);***
INPUT” Введите телефон ”;TEL	***write('Введите телефон '); readln(tel);***
INPUT” Введите доп инф. ”;DOP	***write('Введите доп инф.'); readln(dop);***

Записываем в файл.

WRITE #2,IM,ADR,TEL,DOP	***write(f,im,adr,tel,dop);***
LOOP UNTIL W=”0”	***until (w='0');***
PRINT	***writeln;***
CLOSE #2	***close (f);***
END SUB	***end;***

Составим процедуру считывания информации из файла.

SUB PRINTT (A AS STRING)	***procedure print(a:string);***
DEFSTR A,D,I,T	***var im,adr,tel, dop:string;***
	begin
Открываем файл A	***Связываем переменную f***
для чтения как 2.	***с внешним файлом и открываем файл для чтения.***
OPEN A FOR INPUT AS #2	***assign(f,a);reset(f);***
WHILE NOT EOF(2)	***while not eof(f) do***
	Begin

Считываем данные и выводим на экран.

INPUT #2, IM,ADR,TEL,DOP	read(f,im,adr,tel,dop);
PRINT IM,ADR,TEL,DOP	write('имя ',im,' адрес ', adr,' тел ',tel,' доп инф',dop);
PRINT	writeln;
WEND	end;
CLOSE #2	close(f);
END SUB	end;

Составим процедуру поиска информации об отдельных людях

```
SUB POISK(A AS STRING)
DEFSTR A,D,F,I,P,T
```

Открываем файл А
для чтения как 2.

```
40 OPEN A FOR INPUT AS #2
INPUT” Введите имя ”;P
WHILE NOT EOF(2)
INPUT#2,IM,ADR,TEL,DOP
```

```
procedure poisk(a:string);
var p,im,adr,tel, dop,
fl:string;
label 40;
begin
```

Связываем переменную f
с внешним файлом и
открываем файл для чтения.

```
40: assign(f,a);reset(f);
write('Введите имя ');readln(p);
while not eof (f) do begin
read(f,im,adr,tel,dop);
```

Сравниваем введенное имя с именами хранящимися в файле.

Для простоты не учитываем регистры.

IF P=IM THEN PRINT IM,ADR,	***if p=im then begin write(im,' ',adr,'***
TEL,DOP	***',tel,' ',dop);end;***
PRINT	***writeln;***
WEND	***end;***
CLOSE #2	***close (f);***
INPUT" Повторим поиск Y/N";FL	***write('Повторим поиск ? ');readln(fl);***
IF UCASE(FL)="Y" THEN 40	***if (upcase(fl[1])='Y') goto 40;***
PRINT	***writeln;***
END SUB	***end;***

Самостоятельно осуществите поиск по адресу, телефону, дополнительной информации. Измените процедуру поиска по части имени, адреса и др.

Составим процедуру добавление в файл новых данных

SUB APP(A AS STRING) ***procedure app(a:string);***

DEFSTR A,D,I,T,W ***var im,adr,tel, dop,w:string;***

begin

Открываем файл А для добавления как 2.

Связываем переменную f с внешним файлом и открываем файл для записи.

Почему использовали reset а не append.

OPEN A FOR APPEND AS #2 ***assign(f,a);reset(f);***

{Устанавливаем указатель на конец файла}

while not eof(f) do read(f,im,adr,tel,dop);

DO ***repeat***

INPUT" Для выхода введите 0";W	write('Для выхода введите 0'); readln(w);
IF W="0" THEN EXIT DO	if w='0' then break;
LINE INPUT" Введите имя ";IM	write('Введите имя '); readln(im);
LINE INPUT" Введите адрес ";ADR	write('Введите адрес '); readln(adr);
LINE INPUT" Введите телефон ";TEL	write('Введите телефон'); readln(tel);
LINE INPUT" доп информ.";DOP	write(' доп информ.'); readln(dop);

Записываем в файл.

WRITE#2,IM,ADR,TEL,DOP	***write(f,im,adr,tel,dop);***
LOOP UNTIL W="0"	***until (w='0');***
PRINT	***writeln;***
CLOSE#2	***close (f);***
END SUB	***end;***

Составить процедуру уничтожение отдельных записей.

SUB DEL(A AS STRING)	***procedure del(a:string);***
DEFSTR A-B,D,I,T	***var b, im,adr,tel, dop:string;***
DEFINT C,K	***c,k:integer;***
	label 20;
	label30;
	begin

В файл B будем записывать оставшуюся информацию. Можно было бы записать и в старый файл (смотри следующий пример). Мы это сделали только для демонстрации такого подхода.

B=”FIL90B.TXT”	*b:='fil90b.txt';*
20:	*20:*

Открываем файл A для чтения как 2.

Связываем переменную f с внешним файлом и открываем файл для чтения.

OPEN A FOR INPUT AS #2

assign(f,a);reset(f);

Открываем файл B для записи как 3.

Связываем переменную f1 с внешним файлом и открываем файл для записи.

Почему взяли rewrite(f1) а не reset(f1)

OPEN B FOR OUTPUT AS #3

assign(f1,b);rewrite(f1);

Продолжение программы

30:	***30:***
INPUT”номер удаляемой записи ”;C	***write('номер удаляемой записи'); readln(c);***
IF C<0 THEN PRINT”C<0”:GOTO 30	***if c<0 then begin write ('C <0 ');goto 30;end;***
IF C=0 THEN	***if c=0 then***
CLOSE:CALL PRINTT(A):GOTO 20	***begin close(f); print(a); goto 20;***
END IF	***end;***
K=1	***k:=1;***
WHILE NOT EOF(2)	***while not eof(f) do begin***

Считываем информацию из файла

INPUT#2,IM,ADR,TEL,DOP	read(f,im,adr,tel,dop);

Если C<>K то записываем информацию в другой файл.

IF C<>K THEN	if c<> k then
WRITE#3,IM,ADR,TEL,DOP	begin write(f1,im,adr,tel,dop);end;
K=K+1	k:=k+1;
WEND	end;
IF C>=K-1 THEN	if c>k-1 then
PRINT" Такой записи нет"	write('Такой записи нет ');
END IF: PRINT	writeln;
CLOSE	close (f); close(f1);
END SUB	end;

Составить процедуру изменения информации в файле.

SUB RED(A AS STRING)	***procedure red(a:string);***
DIM IM(15) AS STRING	***Type massiv=array[1..15] of string;***
DIM ADR(15) AS STRING	***Var im, adr, tel, dop: massiv;***
DIM TEL(15) AS STRING	***c,i:integer;***
DIM DOP(15) AS STRING	***fl:string***
DEFINT C,I	***label 50;***
DEFSTR F	***begin***
OPEN A FOR INPUT AS #2	***assign(f,a);reset(f);***
C=1	***c:=1;***
WHILE NOT EOF(2)	***while not eof(f) do begin***

Считываем из файла и заносим в соответствующие массивы.

INPUT#2,IM(C),ADR(C), TEL(C),DOP(C)	read(f,im[c],adr[c],tel[c], dop[c]);

Выводим на экран считанную информацию.

PRINT IM(C),ADR(C),TEL(C), DOP(C)	writeln(im[c],adr[c],tel[c], dop[c]);
PRINT	writeln;
INPUT" Вы хотите изменить эту информацию Y/N ? ";FL	write('Вы хотите изменить эту информацию Y/N ? '); readln(fl);
IF UCASE(FL)<>"Y" THEN	if(upcase(fl[1])<>'Y') then 50 goto 50;
PRINT " Если вы не введете	writeln('Если вы не введете

Продолжение программы

INPUT”Введите новое имя ”;FL	write('Введите новое имя '); readln(fl);
IF FL<>”” THEN IM(C)=FL	if fl<>'' then im[c]:=fl;
INPUT” Введите новый адрес ”;FL	write('Введите новый адрес'); readln(fl);
IF FL<>”” THEN ADR(C)=FL	if fl<>'' then adr[c]:=fl;
INPUT”новый номер телефона ”;FL	write(' новый номер телефона '); readln(fl);
IF FL<>”” THEN TEL(C)=FL	if fl<>'' then tel[c]:=fl;
INPUT”Новая доп информация”;FL	write('Новая доп информация '); readln(fl);
IF FL<>”” THEN DOP(C)=FL	if fl<>'' then dop[c]:=fl;

50:	***50:***
C=C+1	***c:=c+1***
WEND	***end;***
CLOSE #2	***close(f);***
PRINT	***writeln;***

В отличие от предыдущего примера, здесь мы записываем в этот же файл.

OPEN A FOR OUTPUT AS #2 ***assign(f,a);rewrite(f);***

{Почему rewrite, а не reset}

FOR I=1 TO C-1	***for i:=1 to c-1 do begin***
WRITE #2,IM(C),ADR(C),	***write(f,im[i],adr[i],***
TEL(C),DOP(C)	***tel[i], dop[i]);***
PRINT:NEXT	***writeln;end;***
CLOSE	***close(f);***
END SUB	***end;***

Основная программа примера

Место декларации процедур

	program fil90;
	uses crt;
DEFSTR A,F	var a, fl, flag:string;
	f:file of string;
	f1:file of string;
	begin
CLS	clrscr;
A=”FIL90,TXT” ‘A=”FIL90B.TXT”	a:='fil90.txt'; {a:='fil90b.txt';}
DO	repeat
INPUT “Продолжим Y/N-выход ”;	write('Продолжим Y/N-выход ?');
FLAG	readln(flag);
IF UCASE(FLAG)<>”N” THEN	if (upcase(flag[1])<>'N') then
EXIT DO	exit;

INPUT” Создать новый файл Y/N ?”;FL	write('Создать новый файл Y/N ?'); readln(fl);
IF UCASE(FL)=”Y” THEN CALL REWRITE(A)	if (upcase(fl[1])='Y') then rewrit(a);
INPUT”Печать всей информации информации Y/N ?”;FL	write('Печать всей информации Y/N ? '); readln(fl);
IF UCASE(FL)=”Y” THEN CALL PRINTT(A)	if (upcase(fl[1])='Y') then print(a);
INPUT”Добавление информации информации Y/N ?”;FL	write('Добавление информации Y/N ? '); readln(fl);
IF UCASE(FL)=”Y” THEN CALL APP(A)	if (upcase(fl[1])='Y') then app(a);

Продолжение программы.

INPUT”Удаление информации Y/N ?”;fl	write('Удаление информации Y/N ? '); readln(fl);
IF UCASE(FL)=”Y” THEN CALL DEL(A)	if (upcase(fl[1])='Y') then del(a);
INPUT” Поиск информации Y/N ?”;FL	write('Поиск информации Y/N ? '); readln(fl);
IF UCASE(FL)=”Y” THEN CALL POISK(A)	if (upcase(fl[1])='Y') then poisk(a);
INPUT “ Редактирование информации Y/N ?”;FL	write('Редактирование информации Y/N?'); readln(fl);
IF UCASE(FL)=”Y” THEN CALL RED(A)	if (upcase(fl[1])='Y') then red(a);
LOOP UNTIL UCASE(FLAG)=”N”	until (upcase(flag[1])='N');
END	end.

Домашнее задание.

А) Как можно уничтожить отдельные записи в файле?

Б) Как изменить информацию в файле?

В) Как производиться поиск информации в файле?

QB TP

Урок 30

Тема: Операции для работы с файлами

План урока:

1. Проверка домашнего задания.
2. Файлы с произвольным доступом.
3. Упорядочение элементов файлов.
4. Домашнее задание.

Проверка домашнего задания.

А) Как можно уничтожить отдельные записи в файле?

Записать в новый или старый файл всю информацию, кроме уничтожаемой.

Б) Как изменить информацию в файле?

Считать информацию в массив, изменить информацию и записать элементы массива в старый или новый файл

В) Как производиться поиск информации в файле?

Считываем информацию из файла и сравниваем считанную запись (часть записи) с данной информацией.

Qbasic. Файлы с произвольным доступом

- Файлы с произвольным доступом состоят из записей, доступ к которым может быть осуществлен в любой последовательности.
- Данные хранятся точно в таком виде, в котором они представляются в памяти, обеспечивая, таким образом экономию времени при обработке (поскольку не требуется никакого преобразования), как при записи в файл, так и при чтении из него.
- Файлы с произвольным доступом - лучшее решение задачи организации базы данных, чем последовательные файлы, хотя и они не свободны от недостатков. Одним из них является то, что файлы с произвольным доступом не являются особо переносимыми. **Вы не можете проникнуть внутрь них с помощью редактора или распечатать их в осмысленном виде на экране.**
- Действительно, перенесение файла с произвольным доступом, созданного в Qbasic, на другой компьютер или язык, возможно, потребует того, чтобы вы написали программу-транслятор для чтения информации из файла с произвольным доступом и вывода ее в текстовый (последовательный) файл.

Ниже перечислены операторы и функции Qbasic, которые управляют чтением и записью в файлы с произвольным доступом:

Оператор/Функция	Действие
CLOSE	*Закрывает файл.*
FIELD	*Определяет переменные поля.*
GET	*Считывает запись.*
LOC	*Определяет номер последней считанной записи.*
LSET, RSET	*Присваивают значения переменным поля.*
MKI$, MKL$, MKS$, MKD$	*Преобразуют числа заданного типа в формат, в котором они могут быть присвоены переменным поля.*
CVI, CVD, CVS	*Специальные функции преобразования из символьной форме в числовую при считывание с диска*
OPEN	*Открывает файл с произвольным доступом.*
PUT	*Записывает запись в файл.*

Основное преимущество файлов с произвольным доступом:

каждая запись в файле доступна в любое время.

Например, в базе данных из 23000 элементов, программа может непосредственно обратиться к 22709-й или 11663-й записи без считывания всех предыдущих. Это свойство делает этот тип единственно возможным вариантом при работе с большими файлами, а может быть также и лучшим вариантом для небольших файлов, если они имеют относительно постоянную длину записей.

- Файлы с произвольным доступом расходуют лишнее пространство на диске, т.к. пространство под каждую запись выделяется из расчета на самую длинную запись.

Например, включение поля комментариев из 100 байтов делает каждую запись длиннее на 100 байтов, даже если это поле комментариев использует только одна запись из тысячи.

Последовательность шагов по созданию, записи и чтению файлов с произвольным доступом

1. Открыть файл оператором OPEN и задать длину каждой записи.

OPEN имя файла FOR RANDOM AS #№% LEN = длина

Параметр LEN показывает для Qbasic, что это файл с произвольным доступом. В отличие от последовательных файлов, в данном случае не надо объявлять, открываете ли вы файл для ввода или для вывода, т.к. в файл с произвольным доступом вы можете одновременно и писать и читать.

2. Выполнить оператор FIELD, чтобы определить соответствие между рядом символьных переменных (после выполнения этого оператора они становятся "переменными полей") и "буфером файла".

FIELD имя файла, ширина AS симв-пер [ширина AS симв-пер]...

Этот буфер является приемником для данных, записываемых или считываемых из данного конкретного файла. Оператор FIELD должен быть выполнен по крайней мере один раз для данного файла с произвольным доступом.

3. Чтобы записать запись в файл, используйте операторы LSET и RSET для загрузки переменных полей с данными, которые должны быть записаны. Числа должны быть преобразованы в символьную форму посредством соответствующей функции (например, MKS$ для обычной точности), а потом уже могут быть использованы операторы LSET и RSET. Наконец, используйте оператор PUT, чтобы записать запись в файл на место, которое вы укажете.

4. Чтобы прочитать запись из файла, используйте оператор GET.

Затем загрузите прочитанные значения в ваши символьные или числовые переменные из соответствующих позиций в буфере. Прежде, чем вы сможете что-нибудь с ними сделать, числовые данные должны быть преобразованы из символьного формата с помощью соответствующей функции (например, CVS для чисел обычной точности или CVI, CVL, CVD). Эти функции обратные MKS$, MKI$, MKL$, MKD$

5. Когда вы все сделаете, закройте файл оператором CLOSE.

В связи с тем, что при работе с файлами произвольного доступа придется определять длину записи, объем памяти, приведем типы переменных и размер занимаемой памяти.

СВОДНАЯ ТАБЛИЦА ОПИСАНИЯ ТИПОВ ДАННЫХ

Суффикс	Тип переменной	Объявление	Описание	Занимаемый
		DEF_тип	AS_тип	Объем
%	Целая	DEFINT	INTEGER	2 байта
&	Длинная целая	DEFLNG	LONG	4 байта
!	Обычной точности	DEFSNG	SINGLE	4 байта
#	Двойной точности	DEFDBL	DOUBLE	8 байт
$	Строка перемен. Длины	DEFSTR	STRING	1 байт на каждый символ + 4 байта на описатель
$	Строка фиксированной . Длины		STRING*N	N байт
			Пользовательский тип	Равен сумме отдельных элементов

- Покажем несколько программ, использующей файлы с произвольным доступом.
- **В Turbo Pascal 7.0 все файлы, *кроме текстового,* могут рассматриваться как файлы с произвольном доступом.**
- **Установка указателя позиции.**
- **SEEK #1, N** ***Seek(F1,N)***
- Процедура Seek устанавливает текущий указатель на N-й элемент.
- **Qbasic N >0** **Turbo Pascal 7.0 *N>=0***
- Определение номера элемента, элемента, на который "смотрит" текущий указатель.
- **LOC**(Номер файла) ***FilePos****(<имя переменной-файла>)*

Для файлов прямого доступа, **LOC** возвращает номер последней записи/чтения.

Позиции нумеруются от нуля.

Определение количества элементов в файле.

Функция LOF(номер файла) возвращает длину файла в байтах.

*Функция **FileSize** (<имя переменной-файла>)*

Для определения числа *элементов в файле нужно*

max=lof(номер файла**)**

len(длину записи**)**

Возвращает текущий размер файла (число элементов файла при счете от единицы)

Создайте файл прямого доступа, элементами которого являются целые случайные числа в диапазоне от -35 до 30. Число элементов в файле нечетно. Установив указатель позиции на первый, средний и последний элемент считайте данные из файла и найдите их сумму.

	program fil97;
	uses crt;
	var f:file of integer;
DEFINT I, M-N,S	s,n,m,i:integer;
DEFSTR A	a:string;
	begin
CLS	clrscr;
A=”d:\fil97.txt”	a:= 'd:\fil97.txt';

Открываем файлы произвольного доступа.
Так как тип целый, то длина равна двум.

OPEN A FOR RANDOM	assign(f,a); rewrite(f);
AS #1 LEN=2	
RANDOMIZE TIMER	randomize;
INPUT”N=”;N	write('n=');readln(n);
FOR I=1 TO N	for i:=1 to n do begin
M=RND*65-35	m:=trunc(random(65))-35;
PRINT M;” ‘;	write(m,' ');

Записываем в файл данные.

PUT #1,I, M	write(f,m);
NEXT	end;

Закрываем файл

CLOSE #1	close(f);
PRINT	writeln;

Продолжение программы

Открываем файлы произвольного доступа.

FOR A FOR RANDOM	**assign(f,a);reset(f);**
AS #1 LEN=2	
MAX=LOF(1)\2	
PRINT”число записей =”; writeln('число записей=', MAX filesize(f));	

Считываем из файла первый элемент и суммируем значения.

SEEK#1,1:GET#1	**seek(f,0);read(f,n);**
,,N:PRINT N:S=S+N	**write(n:4);s:=s+n;**

Считываем из файла средний элемент и суммируем значения.

SEEK#1,MAX\2+1:GET#1, seek(f, filesize(f) div	
, N: PRINT N:S=S+N	**2);read(f,n);write(n:4); s:=s+n;**

QB TP

Считываем из файла последний элемент и суммируем значения.

SEEK#1,MAX:GET#1, , N:PRINT N:S=S+N	seek(f,filesize(f)-1);read(f,n); write(n:4); s:=s+n;
CLOSE #1	close(f);
PRINT “S=”;S	writeln(‘ s=’,s);
	readln;
END	end.

Создать файл элементами которого являются целые случайные числа. Упорядочить по убыванию, возрастанию элементы файла.

```
                                program fil98;
                                uses crt;
                                var f:file of integer;
DEFINT I-J,M-N,X-Y      n,m,i,j,x,y:integer;
DEFSTR A                        a:string;
                                begin
CLS                             clrscr;
A=”FIL98.DAN”                   a:='fil98.dan';
```

Создаем файл, содержащий целые случайные числа.

Так как тип целый, то длина записи равна двум.

OPEN A FOR RANDOM AS #1	assign(f,a);rewrite(f);
LEN=2	
RANDOMIZE TIMER	randomize;
INPUT “N=”;N	write('n=');readln(n);
FOP I=1 TO N	for i:=1 to n do begin
M=RND*65-22	m:=trunc(random*65)-25;
PRINT M;” “;	write(m,' ');

Записываем данные в файл.

PUT#1,I, M	write(f,m);
NEXT	end;
CLOSE	close(f);
PRINT	writeln;

Считываем данные из файла.

OPEN A FOR RANDOM AS #1 LEN=2	assign(f,a);reset(f);

Определяем число записей в файле.

MAX=LOF(1)\2	
PRINT”число записей =”;MAX	writeln('число записей=', filesize(f));
FOR I=1 TO MAX	for i:=1 to filesize(f) do begin
GET#1, I, N	Read(f,n);
PRINT N;” “;	write(n,' ');
NEXT	end;
CLOSE	close(f);
OPEN A FOR RANDOM AS #1 LEN=2	reset(f);
FOR I=MAX TO 1 STEP-1	for i:=filesize(f)-1 downto 1 do
FOR J=1 TO I-1	for j:=0 to i-1 do begin

Разница в начальных значениях цикла из-за того, что процедуры SEEK в Qbasic начинают счет с единицы, а в Turbo Pascal 7.0 с нуля.

SEEK #1,J:GET#1,J, X:SEEK #1,	seek(f,j); read(f,x);
I:GET#1,I,Y	seek(f,i); read(f, y);

Считываем из файла отдельные элементы стоящих на позициях I,J, сравниваем элементы и записываем обратно в файл.

IF X>=Y THEN SEEK#1,J:	if x>=y then begin
PUT#1,J,Y: SEEK#1,I :PUT$1,I,X	seek(f,j);write(f,y);
	seek(f,I);write(f,,x);end;
NEXT J,I	end;
CLOSE #1	close(f);
	writeln;

Открываем файл и считываем данные из файла.

OPEN A FOR RANDOM AS #1 LEN=2	*reset(f);*
MAX=LOF(1)\2	*writeln('размер=',filesize(f));*
FOR I=1 TO MAX	*for i:=1 to filesize(f) do begin* GET#1,I, X
	read(f,x);
PRINT X;” “;	*write(x,' ');*
NEXT	*end;*
CLOSE #1	*close(f*);
	readln;
END	*end.*

Домашнее задание.

1. *Назовите типы переменных.*
2. *Способ описания переменных.*
3. *Что производит процедура Seek в файлах прямого доступа?*
4. *Как объявляется нетипизированный файл?*
5. *В чем отличие типизированного файла от нетипизированного?*

QB TP

Урок 31

Операции для работы с файлами

План урока:

1. Проверка домашнего задания.
2. Файлы с произвольным доступом.
3. Упорядочение элементов файлов.
4. Домашнее задание.

Проверка домашнего задания

1. **Назовите типы переменных.**
2. **Способ описания переменных.**
3. **Что производит процедура Seek в файлах прямого доступа?**
4. **Как объявляется нетипизированный файл?**
5. **В чем отличие типизированного файла от нетипизированного?**

Текстовые файлы

Текстовые файлы — это файлы, содержащие символы, разделенные на строки. Причем в конце каждой строки стоит признак конца строки. Для их описания используется служебное слово Text.

Var A: Text.

Так как в Qbasic нет полной аналогии текстовым файлам в Turbo Pascal 7.0, то в данной задаче мы воспользуемся файлами последовательного доступа. Затем в каждой строке найдем количество пробелов и определим число элементов в каждой строке.

Создать текстовый файл, содержащий только целые числа, в каждой строке может быть несколько чисел, которые разделяются пробелами. Вывести на экран все числа с учетом разбиения на строки и подсчитать количество элементов в каждой строке.

Пусть в файле содержится следующая информация:

1 2 3 4 5 6 7

-1 -2 -3 -4

-1 -2

Этот файл можно создать:

1. В среде Qbasic или в Turbo Pascal таким образом:

- создайте новый файл (команда New меню File);
- записать все числа в строках через пробелы;
- сохранить его, например, “d:\fil99.txt” или 'd: fil99.txt'.

2. Или составить программу для создания текстового файлаю

Мы создадим в Turbo Pascal 7.0 первым способом, а в Qbasic пойдем вторым путем.

```
FIL99.TXT                                   program fil99;
CONST N=3                                   const n=3;
                                            uses crt;
DEFSTR A, M                                 var a,m;string;
DEFINT I, K, X                      i,k,x:integer;
                                            F:text;
                                            begin
CLS                                          clrscr;
A = "d:\fil99.txt"                          a = "d:\fil99.txt"
```

Открываем файл А последовательного доступа для записи как 1.

```
OPEN A FOR OUTPUT AS #1
DO
LINE INPUT "Введите строку="; m
IF UCASE$(m) = "N" THEN
EXIT DO
```

Записываем в файл данные.

```
PRINT #1, m
LOOP UNTIL UCASE$(m) = "N"
```

Закрываем файл

```
CLOSE #1
PRINT
```

Теперь этот текстовый файл будем использовать в программе.

Открываем файл А для чтения как 1	*Связываем с внешним файлом, открываем для чтения*
OPEN A FOR INPUT AS #1	**Assign(F, 'a: int1.dan');** **Reset(F);**
	Пока не конец файла
WHILE NOT EOF(1)	**While Not Eof(F) Do**
	Begin
	Начальное количество элементов строки
K=0	**k:=0;**
Считываем всю строку	***пока не конец строки***
LINE INPUT #1, M	**While Not Eoln(F) Do begin**

Печатаем ее на экран

PRINT M;	***считываем очередное число***
M=M+" "	***и выводим его на экран***
FOR I = 1 TO LEN(M)	**Read(F, x); Write(x, ' ');**
IF MID$(M, J, 1) = " " THEN	**k:=k+1; {увеличиваем счетчик}**
K=K + 1	
NEXT	**End;**
PRINT " K="; K	**Writeln(' в строке ', k, ' элементов');**

Переходим к следующей строке файла

	Readln(F);
WEND	**End;**

Закрываем файл

CLOSE(1)	**Close(F);**
	Readln;
END	**End.**

QB TP

Структурированный тип данных, так называемые записи (RECORD), позволяющие хранить вместе переменные, имеющие различные типы данных.

Тип данных RECORD представляет программисту возможность объединить в одну связную структуру различные по типу и смыслу элементы (поля). В качестве примера можно привести запись представляющую информацию о работнике: его табельный номер, фамилию и тарифную ставку.

Определим тип данных RECORD с помощью оператора TYPE

TYPE RABOTNIK	TYPE rabotnik=record
TABNOMER AS INTEGER	Tabnomer:integer;
FAMILY AS STRING*15	Family:strind[22];
STAWKA AS SINGLE	Stawka:real;
END TYPE	End;
DIM R AS RABOTNIK	Var R: rabotnik

Запись занимает в памяти столько

байт, сколько занимают в сумме каждый

из составляющих его элементов.

У нас 2+15+4=19

Обращение к полям записи выполняется с помощью указания имени всей записи и имени отдельного поля, причем они должны быть разделены точкой.

Например: R.TABNOMER или R.stawka

Для сокращения длины идентификаторов в Turbo Pascal 7.0 обеспечивает оператор WITH … DO

Дан список данных о группе баскетболистов с указанием имени, роста, числа забитых мяче
Создайте файл, содержащий информацию о баскетболистах. Проверьте, что вам это удалос
Определите число записей в файле.

‘FIL100	program fil100;
	uses crt;
CONST N=3	const n=3;
TYPE BASK	type bask=record
IM AS STRING*50	im : string[50];
ROST AS STRING	rost : real;
GOL AS INTEGER	gol : integer;
END TYPE	end;
DIM B(N) AS BASK	var b:array [1..n] of bask;
	f:file of bask;
DEFINT I,R	i,R:integer;
DEFSTR A	a:string;
DEFSNG S	s,s1:single;
	begin
CLS	clrscr;
A=”D:\F100.DAT”	a:='d:\f100.dat';

Открываем файл произвольного доступа. OPEN A FOR RANDOM AS #1
LEN=LEN(B(N)) assign(f,a);rewrite(f);

FOR I=1 TO N	for i:=1 to n do Begin
Вводим данные о баскетболистах с клавиатуры.	
INPUT”Введите имя “;B(I).IM	write(' Введите имя '); readln(b[i].im);
INPUT”Введите рост”;B(I).ROST	write(' Введите рост '); readln(b[i].rost);
INPUT” Введите число голов”;B(I).GOL	write(' Введите число голов '); readln(b[i].gol);
Записываем в файл данные.	
PUT #1,I, B(I)	write(f,b[i]);
NEXT	end;
CLOSE(1)	close(f);
PRINT	writeln;
Открываем файл произвольного доступа.	
OPEN A FOR RANDOM AS #1	*assign(f,'d:\f100.dat'); reset(f);*
LEN=LEN(B(N))	
Определяем число записей в файле.	
R= LOF(1)\LEN(B(N))	*r:=filesize(f));*
PRINT”Число записей “;R	*writeln('Число записей=',r;*

FOR I=1 TO R	for i:= 1 to r do Begin
Считываем из файла.	
GET#1,I,B(I)	read(f,b[i]);
PRINT I,B(I).IM,B(I),ROST,B(I),GOL	write(b[i].im,' ',b[i].rost:5:2,' ',b[i].gol);
Находим сумму.	
S=S+B(I).ROST:S1=S1+B(I).GOL	s:=s+b[i].rost;s1:=s1+b[i].gol;
PRINT	writeln;
NEXT	end;
CLOSE(1)	close(f);
PRINT”SR ROST=”;USING”##.##”;S/R	writeln('sr rost=',s/r:5:2)
PRINT” SR GOL=”;USING”##.##”; S1/R	writeln(' sr gol=',s1/r:5:2);
	readln;
END	end.

QB TP

Домашнее задание.

1. Что показывает функция EOF?
2. Каков результат действия операторов PUT, GET ?
3. Что производит процедура Seek в файлах прямого доступа?
4. В чем сходство и различие между массивом и файлом?

Контрольная работа.

Вариант 1.

1. Дан файл F, компоненты которого являются действительными числами. Найти сумму компонент файла.

2. Дан файл F, компоненты которого являются действительными числами. Найти наибольшее из значений компонент с нечетными номерами.

3. Даны символьный файлы F и G. Записать в файл H все начальные совпадающие компоненты файлов F и G.

Вариант 2.

1. Дан файл F, компоненты которого являются действительными числами. Найти произведение компонент файла;

2. Дан файл F, компоненты которого являются действительными числами. Найти наименьшее из значений компонент с четными номерами.

3. Дан символьный файл F. Записать в файл H с сохранением порядка следования те символы файла F, которым в этом файле предшествует буква "а".

Контрольная работа

Вариант 3.

1. Дан файл F, компоненты которого являются действительными числами. Найти сумму квадратов компонент файла;
2. Дан файл F, компоненты которого являются действительными числами. Найти сумму наибольшего и наименьшего из значений компонент.
3. Дан символьный файл F. Записать в файл H с сохранением порядка следования те символы файла F вслед за которым в этом файле идет буква "а".

Вариант 4.

1. Дан файл F, компоненты которого являются действительными числами. Найти модуль суммы и квадрат произведения компонент файла;
2. Дан файл F, компоненты которого являются действительными числами. Найти наибольшее из значений компонент.
3. Определить количество слов в символьном файле.

Ответить на вопросы. Qbasic

1.1 Формат открытия файла
1.2 Какие операции возможны над файлами последовательного доступа. (открытие, чтение и\или запись, закрытие)
1.3 Приведите примеры имени файла.
1.4 Режимы работы с файлами (OUTPUT, APPEND, INPUT)
1.5 Для чего служит номер файла?
1.6 Чем отличаются операторы WRITE#, PRINT#?
1.7 Как редактируются последовательные файлы?
1.8 Роль оператора CLOSE?
1.9 Что показывает функция EOF?
1.10 Какие бывают методы доступа к информации в файлах?
1.11 Пример открытия файла.
1.12 Сколько файлов можно открыть?
1.13 Какова последовательность работы с файлами прямого доступа?
1.14 Формат оператора OPEN?
1.15 Каков результат действия операторов PUT, GET?
1.16 Назовите типы переменных.
1.17 Способ описания переменных.
1.18 Способ объявления переменных.
1.19 Занимаемый объем памяти переменными.
1.20 Какие переменные относятся к пользовательскому типу данных.

Ответить на вопросы. TPascal 7.0

1. *Что называется файлом?*
2. *В чем различие между структурой логического и структурой физического файла.*
3. *В чем сходство и различие между массивом и файлом?*
4. *По каким признакам классифицируют файла в Turbo Pascal 7.0?*
5. *Что необходимо выполнит для открытия файла?*
6. *Какие процедуры предназначены для открытия файла и как они работают?*
7. *Для чего предназначена процедура Close?*
8. *Каких типов допускаются описание типизированных файлов?*
9. *Как нумеруются элементы типизированных файлов?*
10. *По каким правилам выполняется чтение из типизированных файлов?*
11. *Каков формат имеет процедура **WRITE** для типизированных файлов?*
12. *Что возвращает функция Filepos в файле прямого доступа?*
13. *Что возвращает функция Filesize в файлах прямого доступа?*
14. *Что производит процедура Seek в файлах прямого доступа?*
15. *В чем состоят особенности текстовых файлов?*
16. *В чем отличие текстового файла от file of Char?*
17. *Как работает процедура Readln в текстовом файле?*
18. *Как работает процедура Writeln в текстовом файле?*
19. *Как объявляется нетипизированный файл?*
20. *В чем отличие типизированного файла от нетипизированного?*

Задачи для самостоятельного решения

- 1. Дан файл F, компоненты которого являются действительными числами. Найти:
- а) Сумму компонент файла;
- б) Произведение компонент файла;
- в) Сумму квадратов компонент файла;
- г) Модуль суммы и квадрат произведения компонент файла;
- д) Последнюю компоненту файла;
- е) Два самых маленьких элемента;
- ж) Среднее арифметическое элементов;
- з) Среднее арифметическое для чисел, стоящих с М по К место в файле;
- и) Сумму чисел, стоящих на четных и нечетных местах.
- 2. Дан файл F, компоненты которого являются действительными числами. Найти:
- а) Наибольшее из значений компонент;
- б) Наименьшее из значений компонент с четными номерами;
- в) Наибольшее из значений компонент с нечетными номерами;
- г) Сумму наибольшего и наименьшего из значений компонент;
- д) Разность первой и последней компонент файла;
- е) минимальное из чисел, стоящих с М по К место в файле;
- ж) максимальное среди чисел, стоящих на 3, 6,9,…3к… местах;
- з) два самых больших элемента;
- и) сумму, для чисел находящихся в диапазоне от А до В.

Задачи для самостоятельного решения

3. Дан файл F, компоненты которого являются целыми числами. Найти:

а) Количество четных чисел среди компонент;

б) Количество удвоенных нечетных чисел среди компонент;

в) Количество квадратов нечетных чисел среди компонент;

г) Минимальное среди чисел кратных трем;

д) Количество простых чисел в файле;

е) Каких чисел больше: четных или нечетн;

ж) Количество чисел с двумя и тремя нулями в конце числа;

и) Произведение чисел кратных 3 и не превосходящих числа B.

4. Дано натуральное N. Записать в файл целые числа B1, B2, ..., Bn, определенные ниже:

а) I I=1,2,3,...,N

б) I!

в) I^2

г) 2^I

5. Дан символьный файл. Добавить в файл символы E,N,D.

Задачи для самостоятельного решения

6. Дан символьный файл F.

   а) Подсчитать число вхождений в файл сочетаний AB.

   б) Определить входит ли в файл сочетание abcdef?

   в) Подсчитать число вхождений в файл каждой из букв a, b, c, d?

7. Даны символьный файлы F,G. Определить совпадают ли компоненты файла F с компонентами файла G. Если файлы F и G отличаются друг от друга, то получить номер первой компоненты, в которой файлы F и G отличаются друг от друга. В случае, когда один из файлов имеет N компонент (n>=0) и повторяет начало другого (более длинного файла), ответом должно служить число N+1.

8. Даны символьный файлы F и G. Записать в файл H все начальные совпадающие компоненты файлов F и G.

9. Дан символьный файл F. Записать в файл H с сохранением порядка следования те символы файла F:

   а) которым в этом файле предшествует буква "a";

   б) вслед за которым в этом файле идет буква "a".

10. Дан символьный файл F. Группы символов, разделенные пробелами (одним или несколькими) и не содержащих пробелы внутри себя, будем называть словами. Удалить из файла однобуквенные слова и лишние пробелы. Результат занести в файл H.

Задачи для самостоятельного решения

11. Дан символьный файл F. Найти самое длинное слово среди слов, вторая буква которых есть "е". Если слов с наибольшей длиной несколько, то найти последнее. Если таких слов нет вообще, то сообщить об этом. Решить задачу:

    а) полагая, что слова состоят не более чем из 10 символов;

    б) без ограничения на число символов в слове.

12. Определить, сколько в файле имеется слов состоящих из 1-го, 2-х, 3-х и т.д. символов. Полагаем, что количество символов в слове не превосходит 20.

13. Определить количество слов в символьном файле.

14. Дан символьный файл. Предполагается, что длина одного слова не более 10. Подготовить файл H для печати слов в две колонки.

15. Прямая на плоскости задается уравнением Ax+By+C=0, где A и B одновременно не равны нулю. Пусть коэффициенты A,B,C целые числа. Пусть файл F содержит коэффициенты некоторых прямых (не менее трех). Переписать из файла F в файл H коэффициенты тех прямых, которые

    а) параллельные первой прямой, заданной в файле F;

    б) указаны в а), но дополнительно требуется, чтобы все прямые были различны;

    в) пересекали первую из прямых, заданных в файле F.

Задачи для самостоятельного решения.

16. Багаж пассажира характеризуется количеством вещей и общим весом вещей. Дан файл, содержащий информацию о багаже несколько пассажиров. (т.е. соответствующую пару чисел)

а) Найти багаж, средней вес одной вещи в котором отличается не более чем на 0,3кг от общего среднего веса вещи.

б) Найти число пассажиров, имеющих более двух вещей, и число пассажиров, количество вещей которых превосходят среднее число вещей.

в) Определить имеются ли два пассажира, багажи которых совпадают по числу вещей и различаются по весу не более чем на 0,5кг.

г) Выяснить, имеется ли пассажир, багаж которого превышает багаж каждого из остальных пассажиров и по числу вещей, и по весу.

д) Выяснить, имеется ли пассажир, багаж которого состоит из одной вещи весом не менее 30кг.

е) Дать сведения о багаже, число вещей в котором не меньше, чем в любом другом багаже с этим же числом вещей.

17. Сведения об ученике состоят из его имени и фамилии и названия класса, в котором он учится. Дан файл Н, содержащий сведения об учениках школы.

а) Выяснить, имеются ли в школе однофамильцы.

б) Выяснить, имеются ли однофамильцы в каких-либо параллельных классах.

в) Выяснить, имеются ли однофамильцы в каком-нибудь классе.

г) Ответить на вопросы а)-в), но в отношении учеников, у которых совпадают и имя и фамилия.

Задачи для самостоятельного решения.

д) Выяснить, в каких классах насчитывается более 35 учеников.

е) Выяснить, на сколько человек в восьмых классах больше, чем в десятых.

ж) Собрать в файле G сведения об учениках 9-х и 10-х классов, поместив в начале сведения об учениках класса 9а, затем 9б т.д., затем 10а, 10б и т.д.

з) Получить список учеников данного класса по следующим образцам:

фамилия имя

фамилия и.

и. Фамилия

18. Дан файл F, содержащий те же самые сведения об учениках школы, что и в предыдущей задаче, и дополнительно отметки, полученные учениками в последней четверти.

а) Выяснить, сколько учеников школы не имеют отметок ниже четырех

б) Собрать в файле G сведения о лучших учениках школы, т.е. об учениках, не имеющих отметок ниже четырех и по сумме баллов не уступающих другим ученикам своего и параллельных классов.

19. Сведения об автомобиле состоят из его марки, номера и фамилии владельца. Дан файл F, содержащий сведения о нескольких автомобилях.

Найти:

а) фамилии владельцев и номера автомобилей данной марки;

б) количество автомобилей каждой марки.

Задачи для самостоятельного решения

20. Дан файл F, содержащий различные даты. Каждая дата - это число, месяц, год. Найти:

а) год с наименьшим номером;

б) все весенние даты;

в) самую позднюю дату.

21. Дан файл F, содержащий сведения о книгах. Сведения о каждой из книг - это фамилия автора, название и год издания.

а) Найти названия книг данного автора, изданных с 1960 г.

б) Определить, имеется ли книга с названием "Информатика". Если да, то сообщить фамилию автора и год издания. Если таких книг несколько, то сообщить имеющиеся сведения обо всех этих книгах.

22. Дан файл F, который содержит номера телефонов сотрудников учреждения: указывается фамилия сотрудника, его инициалы и номер телефона. Найти телефон сотрудника по его фамилии и инициалам.

23. Дан файл F, содержащий сведения о веществах; указывается название вещества, его удельный вес и проводимость (проводник, полупроводник, изолятор).

а) Найти удельные веса и названия всех полупроводников.

б) Выбрать данные о проводниках и упорядочить их по убыванию удельных весов.

Задачи для самостоятельного решения

24. Дан файл F, содержащий сведения об экспортируемых товарах: указывается наименование товара, страна, импортирующая товар, и объем поставляемой партии в штуках. Найти страны, в которые экспортируется данный товар, и общий объем его экспорта.

25. Дан файл F, содержащий сведения об игрушках: указывается название игрушки (например, кукла, кубики, мяч, конструктор и т.д.), ее стоимость в копейках и возрастные границы детей, для которых игрушка предназначена (например, для детей от двух до пяти лет). Получить следующие сведения:

а) названия игрушек, цена которых не превышает 4 руб. и которые подходят детям 5 лет;

б) цену самого дорогого конструктора, оформленную по образцу ...руб. ...коп.;

в) названия наиболее дорогих игрушек (цена которых отличается от цены самой дорогой игрушки не более чем на 1 руб.);

г) названия игрушек, которые подходят как детям 4-х лет, так и детям 10 лет.

д) цены всех кубиков, оформленные по образцу, указанному в б);

е) можно ли подобрать игрушку, любую, кроме мяча, подходящую ребенку 3 лет, и дополнительно мяч так, чтобы суммарная стоимость игрушек не превосходила 5 руб.?

ж) имеется ли мяч ценой 2 руб. 50 коп., предназначенный детям от 3 до 8 лет?; если нет, занести сведения об этой игрушке в файл f.

Задачи для самостоятельного решения

26. Дан файл F, компоненты которого являются натуральными числами. Число компонент файла кратно четырем. Каждые две компоненты определяют координаты двух точек.

а) Считая, что заданы координаты концов отрезков, построить все такие отрезки.

б) Считая, что заданы координаты противоположных углов прямоугольника, построить все такие прямоугольники.

в) Считая, что заданы координаты центра окружности и одной из ее точек, построить все такие окружности.

27. Продумайте структуру файла, содержащую информацию о нападающих хоккейной команды, в которой каждому игроку соответствует одна запись. Напишите программу, извлекающего в буфер из файла данные о числе забитых голов восьмым игроком.

28. Напишите программу, которая облегчила бы зубному врачу учет пациентов. Информацию по каждому пациенту - имя, число поставленных ему пломб и сумму задолженности за лечение - храните отдельных логических записях в файлах прямого доступа. Запустите программу и введите условные данные по нескольким пациентам.

29. Напишите программу, которая позволила бы бухгалтеру задать идентификационный номер клиента, заставляет ЭВМ выдать нужную запись и выдать сумму задолженности клиента за телефон (квартиру и др.) Номер клиента совпадает с номером записи. Пусть программа учитывает уже уплаченную сумму и корректирует величину задолженности в записи.

Задачи для самостоятельного решения

- 30. Создайте файл прямого доступа, содержащий вещественные значения последовательно от 1.0 до 12.0.

  а) Покажите, что вам это удалось.

  б) Замените в файле записи под номерами 4, 7, 11 словами Оля, Коля, Ира.

  в) Покажите, что вам это удалось.
- 31. Запишите имена и коды всех служащих банка в файл прямого доступа.

  а) Покажите, что вам это удалось.

  б) Пусть ЭВМ помогает служащему охраны проверять прибывающих на работу. Когда служащий вводит имя вновь прибывшего, компьютер должен печатать соответствующий код.
- 32. Организуйте с помощью компьютера информацию об ассортименте дамских туфель в обувном магазине. В файле прямого доступа запишите артикул и число пар каждой модели обуви. Всего будет N моделей от 1001 до 1001+N. Проделайте следующее:

  а) сначала введите в файл исходное число пар обуви каждой модели;

  б) содержимое файла увяжите с работой кассового аппарата, чтобы учитывать оставшийся товар всякий раз после продажи очередной пары туфель;

  в) напечатать ассортиментный список с указанием артикула и имеющегося в наличии числа пар каждой модели.

Задачи для самостоятельного решения

33. Допустим вы - член оргкомитета Олимпийских игр. Зимняя олимпиада финишировала, и результаты соревнований записаны в файле прямого доступа. Ваша задача - прочитать данные

о призерах и напечатать таблицу вида:

ЛЫЖНЫЕ ГОНКИ СРЕДИ МУЖЧИН.

Золотая медаль: СЕРГЕЙ ТОЛСТОЙ СССР

Серебряная медаль ИГГИ ПИГГИ США

Бронзовая медаль: Альфредо Минчини ИТАЛ

и т.д. для шести олимпийских дисциплин. Отведите 26 байт на название дисциплины, 30 байт на имя каждого призера и 4 байта на название страны. Напишите программу, которая запрашивала бы по каждой олимпийской дисциплине сведения о медалистах. Разделите эту задачу на три части:

а) напишите программу для ввода данных имен призеров с указанием

ими страны - каждой дисциплине:

б) Напишите программу для выдачи на печать списка призеров с указанием имени и представляемой страны для каждого из них;

в) напишите программу для обработки запросов по любой дисциплине.

Задачи для самостоятельного решения.

34. Воспользуйтесь файлом прямого доступа для хранения библиографической информации. В начальном блоке (размером 512 байт) поддерживайте индекс указателей на остальную часть файла. В последующих блоках разместите 64-байтовые записи, по восемь в блоке, хранящие полные библиографические данные по каждому материалу (статье). Индекс каждой статьи образуется из ее сокращенного (в 12 символов) названия и двух целых чисел, указывающих соответственно номер блока и номер библиографической записи внутри блока. Напишите четыре программы:

а) начальная установка файла. Вероятно, будет удобно в каждом индексе заранее указать номера блока и записи;

б) занесение очередной статьи. Запросите у библиотекаря сокращенное название статьи и запищите ее в индекс. Затем запросите полное библиографическое описание и заносите его в нужный блок;

в) печать индексов и полных библиографических справок по всем статьям;

г) поиск и выдачу информации о любой статье по ее индексу.

Задачи для самостоятельного решения

Текстовые файлы — это файлы, содержащие символы, разделенные на строки. Причем в конце каждой строки стоит признак конца строки.

35. Дан текстовый файл, содержащий целые числа. Найти:
    a) максимальный элемент в каждой строке;
    b) номер данного числа, если такого нет в данной строке, то сообщить об этом.
36. Дан текстовый файл, содержащий строки. Найти:
    a) количество строк;
    b) количество строк, начинающихся и заканчивающихся одинаковыми символами;
    c) самые короткие строки;
    a) симметричные строки.
37. Дан текстовый файл. Вставить в начало каждой строки ее номер и записать преобразованные строки в новый файл.
38. Даны два текстовых файла. Записать в третий только те строки, которые есть и в первом и во втором файлах.
39. Дан текстовый файл. Дописать в его конце следующие данные: количество строк, количество символов в каждой строке, количество элементов в каждой строке.

Используемая литература

1. Абрамов С.А., Гнездилов Г.Г. и др. Задачи по программированию. - М.: НАУКА, 1988г.
2. Гейн А.Г. и др. Основы информатики и вычислительной техники. - М., ПРОСВЕЩЕНИЕ, 1993г.
3. Лепехин Ю.В Сорок пять минут с компьютером. Волгоград: ПЕРЕМЕНА,1996г
4. Филиппов С.В. Занимательный BASIC. - М.:изд-во ЭКОМ, 1997г.

   Мельникова О.И., Бенюшкина А.Ю. Начала программирования на языке QBASIC. – М.:Издательство ЭКОМ,1998г
5. Вострикова З.П., Вострикова Щ.Ю., Туева С.С. Программирование на языке «Бейсик» для персональных ЭВМ.
6. Зельдер Г.А. Программируем на языке QUICK BASIC 4. - М. ABF 1997г.

   Марченко А.И., Марченко Л.А Turbo PASCAL. - Киев:ВЕК, 1999г.
7. Окулов С.М. Турбо ПАСКАЛЬ 7.0.- Киев: издательская группа BHV, 2000г.