Практическое занятие

Тема: РАЗРАБОТКА ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ ТИПИЗИРОВАННЫХ ФАЙЛОВ

Цель занятия – формирование представления о файлах и организации диалога ввода-вывода данных в типизированные файлы

Общие сведения.

В типизированном файле может находиться последовательность данных какого либо одного типа. Отсюда и название. Типы, которые могут содержаться в типизированном файле могут быть любые, кроме строк переменной длины.

Для создания файловой переменной, которая будет указывать на типизированный файл, используется словосочетание file of, после которого указывается тип данных, которые будут содержаться в файле, например:

```
var f:file of integer;
```

Так же можно, в начале, создать свой тип, а затем создать переменную этого типа:

```
Type Tf = file of integer;
var f:Tf;
```

Далее необходимо связать файловую переменную с файлом, на который она будет указывать. Делается это с помощью процедуры assign:

```
assign(f,'c:\5.int');
```

В скобках первым параметром идёт файловая переменная, вторым – строка, которая содержит в себе имя файла. С помощью процедуры Assign мы связали файловую переменную с реальным файлом. Причём самого файла на момент вызова процедуры assign может ещё и не существовать. Просто процедура assign связывает файловую переменную и имя файла, а так же путь к нему.

Далее для того, что бы создать реальный файл необходимо вызвать процедуру rewrite(f). При этом будет создан новый пустой файл, причём если файл с таким именем уже существовал, то он будет удалён. Т.е. процедура rewrite создаёт новый файл, на который указывает файловая переменная:

```
rewrite(f);
```

После того, как был создан новый файл, в него можно записывать данные с помощью уже знакомой нам процедуры write. Причём первым параметром должна находиться файловая переменная, а затем через запятую перечисляются переменные или константы, которые необходимо записать в файл. Причём эти константы и переменные должны быть того типа, который указан при объявлении файловой переменной:

```
write(f,6,7,8,9);
```

Что бы считать данные из файла используется, знакомая нам процедура, read, причём первым параметром должна идти файловая переменная, а затем через запятую переменные, в которые должны быть записаны данные из файла:

```
read(f,a,b,c,d);
```

Когда работа с файлом закончена, его необходимо обязательно закрыть, иначе при завершении работы программы, если файл остался открытым, могут возникнуть проблемы в файловой системе жёсткого диска. Закрывается файл с помощью процедуры Close(f):

```
close(f);
```

Если необходимо открыть уже существующий файл, не удалив его, используется процедура Reset(f):

```
reset(f);
```

Пример программы:

```
Type Tf = file of integer;
var f:Tf;
    a,b,c,d:integer;
begin
  assign(f,'c:\5.int');
  rewrite(f);
  write(f,6,7,8,9);
  close(f);
  reset(f);
  read(f,a,b,c,d);
  writeln(a,b,c,d);
  close(f);
end.
```

Результат выполнения программы:

```
6789
```

В данном примере запись и чтение данных происходит по порядку. Иногда приходится считывать данные из какого-либо места в файле. Для этого существует следующее понятие: файловый указатель. При открытии или создании файла файловый указатель устанавливается в начало файла и указывает позицию 0. Когда мы записываем данные в файл, после записи каждого элемента файловый указатель перемещается на одну позицию. При чтении данных происходит то же самое, после чтения каждого элемента файловый указатель перемещается на одну позицию вперёд.

Для работы с файловым указателем существуют следующие процедуры и функции:

procedure **Seek**(f: file of T; n: LongInt);

Устанавливает текущую позицию файлового указателя в типизированном файле f на элемент с номером n.

function **FilePos**(f: file of T): LongInt;

Возвращает текущую позицию файлового указателя в типизированном файле f.

function **FileSize**(f: file of T): LongInt;

Возвращает количество элементов в типизированном файле f.

Пример:

```
Type Tf = file of integer;
var f:Tf;
    a,b,c,d:integer;
begin
  assign(f,'c:\5.int');
  reset(f);
  seek(f,2);                read(f,c);
  seek(f,filepos(f)-3); read(f,a);
  seek(f,filesize(f)-1);read(f,d);
  seek(f,1);                read(f,b);
  close(f);
  writeln(a,b,c,d);
end.
```

Результат выполнения программы:

```
3456
```

В этом примере чтение данных из файла мы произвели не по порядку, используя процедуры и функции для работы с файловым указателем. Так же стоит сделать следующие замечания: как уже было сказано, первый элемент имеет нулевую позицию, соответственно второй элемент первую позицию и т.д. отсюда, что бы считать или записать первый элемент файловый указатель необходимо поставить в нулевую позицию, второй элемент – в первую и т.д. Стоит ещё добавить, что номер позиции последнего элементы будет равен количеству элементов в файле за вычетом единицы. И ещё одно, прежде чем считывать данные из файла, рекомендую всегда устанавливать файловый указатель в нужную позицию.

Записывать элементы в файл можно так же не по порядку, используя процедуры и функции для работы с файловым указателем. Если взять предыдущий пример, то просто вместо процедуры read подставить процедуру write. Как уже было сказано выше, при записи элемента файловый указатель смещается на одну позицию, при этом происходит стирание старого элемента и запись нового в то место, где находился файловый указатель.

Для чтения всех данных из файла, удобно пользоваться следующей функцией:

function **Eof**(f: FileType): boolean;

Возвращает True, если файловый указатель находится в конце файла f. Данная функция применима ко всем трём типам файлов.

Пример:

```
Type Tf = file of integer;
var f:Tf;
    temp:integer;
begin
  assign(f,'c:\111111.int');
  reset(f);
  While not eof(f) do
   begin
     read(f,temp);
     write(temp);
   end;
  close(f);
end.
```

Результат выполнения программы:

```
3456
```

Процедуры и функции для работы с типизированными файлами:

function **FilePos**(f: file of T): LongInt;

Возвращает текущую позицию файлового указателя в типизированном файле f. Иногда возникает необходимость узнать текущую позицию файлового указателя. Например, если вы пишете не всю программу целиком, а какую-то подпрограмму, и при выходе из вашей подпрограммы файловый указатель необходимо вернуть в прежнее место:

```
var f:file of integer;
    a,c:integer;

Procedure VivodFila;
var pos:longint;
    temp:integer;
begin
```

```
  pos:=filepos(f);
  seek(f,0);
  While not eof(f) do
   begin
     read(f,temp);
     write(temp,' ');
   end;
  seek(f,pos);
end;

begin
  assign(f,'c:\111111.int');
  reset(f);
  seek(f,2);
  VivodFila;
  writeln;
  read(f,a,c);
  writeln(a,'+',c,'=',a+c);
  close(f);
end.
```

Результат выполнения программы:

```
20 4 30 40
30+40=70
```

procedure **Truncate**(f: file of T);

Усекает типизированный файл f, отбрасывая все элементы с позиции файлового указателя.

```
var f:file of integer;
Procedure VivodFila;
.......................
begin
  assign(f,'c:\1234');
  rewrite(f);
  write(f,1,2,3,4,5,6);
  vivodFila;
  writeln;
  seek(f,3);
  Truncate(f);
```

```
    vivodFila;
    close(f);
end.
```

Результат выполнения программы:

```
1  2  3  4  5  6
1  2  3
```

Пример 1.

1. Необходимо создать массив случайных целых чисел, записать его в файл, затем вывести его на экран из файла.

Решение.

1.

```
Program Ispolzovanie_Massiva;
//Программа демонстрирует работу с типизированными файлами

const imya_file = 'c:\massiv.int';//Константа содержит название файла
      max = 30;// размер массива
var massiv: array [1..max] of integer;
    f:file of integer;
    i:integer;

begin
  //Заполняем массив
  for i:=1 to max do
    massiv[i]:=random(1,1000);

  //Записываем в файл
  assign(f,imya_file);
  rewrite(f);
  for i:=1 to max do
    write(f,massiv[i]);
  close(f);

  //Читаем массив из файла
  assign(f,imya_file);
  reset(f);
  for i:=1 to max do
    read(f,massiv[i]);
  close(f);

  //Выводим массив на экран
  for i:=1 to max do
    begin
      write(massiv[i]:5);
      if (i mod 10)=0 then writeln;//Создаем новую строчку
    end;
end.
```

```
  920  288  950  972  812  413  244  579  112  357
  513  855  943  299  292  694   51  976  309  973
  457  561   45  855  308  284  511  119  146  809
```

Пример 2: поменять строки в файле местами.

```
program type_fail;
uses crt;
var
f: file of string;
i: integer;
s, s1, s2, s3: string;
begin
assign (f, 'file.dat');
s1:='Pascal';
s2:=' на ';
s3:='Программирование';
rewrite(f);
write(f, s1, s2, s3); {запись строк в файл}
reset(f);
write('Выводим как есть: ');
while not eof(f) do {вывод содержимого файла}
begin
read(f, s);
write(s);
end;
writeln;
write('Вывод после обработки: ');
for i:=2 downto 0 do
begin
seek(f, i); {смена позиции указателя}
read(f, s);
write(s);
end;
close(f);
end.
```

Задачи для самостоятельного решения:

1. Создать типизированный файл и записать туда n вещественных чисел. Алгоритм решения: открыть файл для записи, ввести значение n, в цикле ввести очередное вещественное число и записать его в файл, закрыть файл.

2. Создать типизированный файл, куда записать n целых чисел. Сформировать массив положительных чисел, делящихся на семь без остатка, используя элементы исходного файла, и упорядочить его по возрастанию элементов.

3. Создать типизированный файл, куда записать n целых чисел. Из файла целых чисел сформировать массив, записав в него только чётные компоненты, находящиеся до минимального элемента.

Задания для выполнения

1. Подготовить конспект.
2. Решить задачи.
3. Написать выводы по работе и ответить на контрольные вопросы.

Контрольные вопросы

1. Что такое типизированный файл?
2. Какие типы данных могут быть элементами типизированного файла?
3. Как описать типизированный файл?
4. Какие операции доступны при работе с типизированными файлами?
5. Как узнать количество элементов типизированного файла?
6. Как управлять положением указателя?