Элементы ЯПВУ. ФАЙЛЫ

Pascal | **C / C++**

Способы организации в ЯПВУ данных в виде файлов позволяют программ осуществлять обмен данными с внешними (по отношению к программам) хранилищами данных, иначе ввод/вывод. Универсальное формальное определения понятия ФАЙЛ – непростая задача, для целей первичного изучения ПЯВУ достаточным определением файла будет следующее:

Файл – это именованная структура данных (внешних по отношению к программе), представляющая собой последовательность элементов данных одного типа неопределенной длины. В ПЯВУ определены операции обмена данными с файлами.

В отношении файла применяются понятия *организация файла* и *дисциплина доступа*. Организация файла – это способ структурирования данных и размещения их во внешнем хранилище (носителе). Дисциплина доступа – это совокупность алгоритмов и соглашений, определяющих порядок взаимодействия программы с компонентами файла (перебора, обмена и пр.)

Возможны два разных способа организации обмена программы с файлом:

- **Записеориентированнный обмен, где Запись – это совокупность данных (фиксированной, переменной или неопределенной длины) обрабатываемых совместно при обмене данными между программой и файлом,**
- **Потокоориентированный обмен, где Поток – это совокупность строк произвольной длины, каждая из которых завершается последовательностью EOL - end of line (коды ASCII 13 (CR) + ASCII 10 (LF), EOL вводится нажатием клавиши Enter), а весь потокоориентированный файл может завершаться символом EOF - end of file (код ASCII 26, EOF вводится одновременным нажатием клавиш CTRL-Z).**

Элементы ЯПВУ. ФАЙЛЫ

Pascal

В Pascal для организации обмена с внешними хранилищами данных используется файловые типы. Элементы файла могут быть любого типа, но не "файл" или "объект". Любой файл может содержать неограниченное количество элементов.

ТИПЫ файлов:

- ♦ **типизированные файлы** – файлы с объявленным типом элементов,
- ♦ **нетипизированные (бестиповые) файлы** – файлы содержащие последовательности элементов произвольного типа (но с оговоренным размером элементов),
- ♦ **текстовые файлы** – файлы содержащие символьные строки переменной длины.

Pascal

Описания типов файлов

- Типизированные файлы

Type <имя_перемен_файлового_типа> = file of <тип_элементов_файла>;

Пример: Type NUM = file of integer;
Var F1, F11 : NUM; или

Var < имя_перемен_файлового_типа > : file of <тип_элементов_файла>;

Пример: Var F3 : file of integer;

- Нетипизированные файлы

Type <имя_перем_файлов_типа> = file;

Пример: Type NUM = file;
Var F1, F11 : NUM; или

Var <имя_перем_файлов_типа> : file;

Пример: Var F1 : file;

- Текстовые файлы

Type <имя_перем_файлов_типа> = text;

Пример: Type Filtxt = text;
Var ft : text; или

Var <имя_перем_файлов_типа> : text;

Пример: Var FT : text;

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Доступ к файлам может быть **последовательным** (очередной элемент можно прочитать/записать только после выполнения аналогичной операции над предыдущим элементом) или **прямым** (можно выполнить чтение/запись произвольного элемента файла по заданному адресу).

Последовательный доступ возможен к файловым переменным *всех* типов: текстовым (text), типизированным (file of) и нетипизированным (file).

Прямой доступ возможен только для переменных файлового типа file и file of, для этого с ними связано понятие текущей позиции. Она указывает на конкретный элемент файла, с которым в данный момент можно выполнять поэлементные действия. В результате выполнения операции текущая позиция может перемещаться настраиваясь на тот или иной элемент файла. Все элементы считаются пронумерованными (начиная с 0). Прямой доступ возможен потому, что данные в файлах типа file и file of условно разделены на блоки одинакового размера (элементы).

Pascal

Переменные файлового типа называются **логическими файлами**, а реальные устройства ввода/вывода и файлы на дисках – **физическими файлами.**

Имена физических файлов описываются строковыми переменными:

- 'history.pas' – имя дискового файла в текущей папке,
- 'd:\BP\source\massive.pas' – полное имя дискового файла,
- 'CON' – консоль (экран дисплея),
- 'LPT1', 'LPT2', 'LPT3', 'PRN' – принтер,
- 'COM1', COM2', AUX'- коммуникационный канал,
- 'NUL' – пустое устройство.

Если имя файла задается в виде пустой строки, то файловая переменная связывается, в зависимость от направления обмена информацией, со стандартными файлами Input (для чтения данных с клавиатуры) или Output (для вывода данных на экран), эти файлы в Borland Pascal считаются открытыми по умолчанию.

Для ввода/вывода в файл надо связать логический файл с физическим, т.к. функции и процедуры Pascal работают только с логическими файлами.

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Организация ввода/вывода в файл :

1. **Объявить файловую переменную,**
2. **Связать её с физическим файлом,**
3. **Открыть файл для чтения и/или записи,**
4. **Выполнить операции ввода/вывода,**
5. **Закрыть файл.**

Обмен между логическими и физическими файлами происходит через буфер в системной области оперативной памяти, который выделяется для каждого открытого файла.

При записи в файл все данные поступают в буфер. Передача данных на внешнее хранилище (дисковый файл или устройство) происходит после заполнения буфера или после специальной команды.

При чтении из файла данные считываются в буфер, причем считано будет не столько, сколько запрашивается, а сколько поместиться в буфер.

Подпрограммы можно разделить на две подгруппы: универсальные, пригодные для любого типа файлов и специализированные, применимые только для определенных типов.

Специализированные подпрограммы будут в дальнейшем отмечены перечнем типов, к которым они применимы, заключенным в круглые скобки: (text, file, file of).

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Процедура assign (f[, 'f_name']); – связывает логический файл f с физическим файлом f_name.
– f – имя файловой переменной (ифп),
– f_name – литеральное имя файла (иф)
Напр.: assign (F1, 'd:\BP\a.txt');
Если путь не задан, файл – в текущей папке. Связь файлов существует пока для переменной f не будет выполнена другая процедура assign (или close).

Процедура reset (f [, size]); – открывает логический файл f для чтения данных, начиная с первого элемента.
Здесь f – ифп, а size – размер записи в файле, используется только для нетипизированных файлов (file), по умолчанию 128 байт (это же размер буфера).
Напр.: reset (F1); Если файл не существует – выдается ошибка. Если уже открыт – открывается снова (текущая позиция в начале файла). Файл типа (text) открывается только на чтение.

Процедура append (f); – открывает текстовый файл для дополнения данных в конец файла.
Здесь f – ифп текстового типа (text).
Напр.: append (F1); Если файл не существует – выдается ошибка. Если уже открыт, то сначала закрывается и потом открывается. Текущая позиция устанавливается перед концом файла.

Процедура rewrite (f [, size]); – создает и открывает физический файл, имя которого присвоено логическому файлу f для записи данных, начиная с первого элемента.
Здесь f – ифп, а size – размер записи (используется только для (file), по умолчанию 128 байт).
Напр.: rewrite (F1); Если файл не существует он создается, если существует – он очищается и записывается с начала. Файл типа (text) открывается только на запись.

Процедура flush (f); – завершает обмен с файлом f без его закрытия (очищает буфер обмена). Здесь f – ифп.
Напр.: flush (F1); Для открытых файлов типа (text).

Процедура close (f); – закрывает открытый логический файл f. Здесь f – ифп.
Напр.: close (F1); Обязательно надо использовать close для завершения работы с открытым для записи (*выходным*) файлом, т.к. при её выполнении происходит выгрузка буфера. Если не выполнить close содержимое буфера может пропасть. Для *входных* файлов close можно не выполнять.

Процедура erase (f); – стирает физический файл связанный с файловой переменной (логическим файлом) f. Здесь f – ифп.
Напр.: erase (F1); Файл к моменту вызова erase должен быть закрыт.

Элементы ПЯВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Процедура rename (f, nm); – переименовывает на диске физический файл связанный с логическим файлом f.
f – ифп, nm – новое имя файла (литерал).
Напр.: rename (F1, new_name); Файл к моменту переименования должен быть закрыт.

Процедура chdir (path); – изменение (смена) текущей папки.
Здесь path – путь к папке на диске (литерал).
Напр.: chdir (d:\bp\bin);.

Процедура getdir (drv, path); – для заданного диска drv помещает имя текущей папки в строковую переменную path.
Здесь drv – номер дискового накопителя (0 – текущий диск, 1 – диск A:, 2 - B:, 3 - C: и т. д.), а path – путь к папке на диске (литерал).
Напр.: getdir (2, directory); GetDir не выполняет проверку наличия диска, а выдает строку '<очередная буква>:\'.

Процедура mkdir (path); – создает новую папку с путём заданным переменной path.
Здесь path – путь к папке на диске (литерал).
Напр.: mkdir (F1); В path не может последней папкой быть имя уже существующей папки.

Процедура rmdir (path); – удаляет пустую папку с путём заданным переменной path.
Здесь path – путь к папке на диске (литерал).
Напр.: rename (F1, new_name); Если путь доступа не существует, является пустым или задает текущую папку, происходит ошибка ввода/вывода.

Процедуры и функции: FindFirst, FindNext, GetFTime, SetFTime, GetFAttr, SetFAttr, FSplit, FSearch –
изучить самостоятельно.

Функция filesize (f); – возвращает общее число элементов файла f.
Здесь f – ифп. (file, file of)
Напр.: filesize (F1); Когда файл пуст функция возвращает значение 0.

Функция diskfree (drv); – возвращает число свободных байтов на диске drv.
Здесь drv – номер дискового накопителя
(0 – текущий диск, 1 – диск A:, 2 - B:, 3 - C: и т.д.)
Напр.: diskfree (3); При недопустимом значении drv функция возвращает значение -1.

Функция disksize (drv); – возвращает общее число байтов на диске drv.
Здесь drv – номер дискового накопителя
(0 – текущий диск, 1 – диск A:, 2 - B:, 3 - C: и т.д.)
Напр.: disksize (3); При недопустимом значении drv функция возвращает значение -1.

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Процедура read ([f,] v1 [, v2, …, vN]);

для текстовых файлов (text)

– считывает N значений из текстового файла в переменные.

Здесь f – ифп, а v1, …, vN – список переменных, в которые считываются значения из f.

Если параметр f опущен, то используется стандартная файловая переменная Input.

Каждый параметр v является переменной символьного, строкового, целого и вещественного типа.

Напр.: read (x, y); Для пустого файла переменная равна 0. Для символьного типа в переменную читается один символ. Для целого типа читается число со знаком до пробела, табуляции, конца строки (лидирующие пробелы и т.д. – игнорируются). Для вещественного типа читается десятичное число, аналогично целому. Для строкового типа в переменную читаются все символы до конца строки или файла. После считывания строки не делается пропуск до следующей строки (надо использовать ReadLn).

для типизированных файлов (file of)

– считывает в переменную элемент файла f. Типы переменной и элемента файла – одинаковы.

Напр.: read (F1, x, y); При каждом считывании указатель позиции сдвигается к следующему элементу, и так до конца файла. Файл должен быть открыт.

Процедура readln ([f,] v1 [, v2, …, vN]);

– выполняет процедуру read и переходит к следующей строке. ReadLn (f); - перемещает текущую позицию к следующей строке. Тип f -(text).

Процедура write ([f,] v1 [, v2, …, vN]);

для текстовых файлов (text)

– записывает N значений параметров v в файл f.

Здесь f – ифп, а v1, …, vN – список параметров, из которые записываются значения в f.

Если параметр f опущен, то используется стандартная файловая переменная Output.

Каждый параметр v является выражением символьного, целого, вещественного, строкового, упакованного строкового или булевого типа, значение которого записывается в файл f. Параметр v имеет вид:

expr [: size [:dec]] где

expr – вводимое в файл выражение, size – минимальная ширина поля dec – число десятичных знаков в вещественном числе с плавающей точкой.

Напр.: write (F1, y+z^3, x); Size – целое число больше 0, по умолчанию равна 17, dec по умолчанию – десятичная строка с плавающей точкой. Для строкового типа: если size опущена в файл записывается expr без лидирующих пробелов; если size больше чем expr, то перед десятичной строкой добавляются лидирующие пробелы.

для типизированных файлов (file of)

– записывает N значений переменных v в файл f. Типы переменной и элемента файла – одинаковы.

Напр.: write (F1, x, y); При каждой записи текущая позиция сдвигается к следующему элементу, когда достигнут EOF – файл расширяется.

Процедура writeln ([f,] v1 [, v2, …, vN]);

– выполняет процедуру write и записывает в файл EOL. Writeln (f); -записывает в файл EOL.(text)

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Процедура

blockread (f, buf, count[, result]);

– считывает count элементов из файла f в переменную buf. **(file, file of)**

Здесь f – ифп; buf – переменная любого типа, в которую происходит считывание; count – выражение, определяющее количество считываемых элементов; result – число фактически считанных элементов.

Напр.: blockread (F1, k, x, c); Размер записи нетипизированного файла равен размеру буфера. Если result опущен, то при несовпадении количества прочитанных элементов с count – ошибка ввода/вывода. За один вызов процедуры можно считать не более 64 Кбайт. После завершения чтения текущая позиция продвигается на число записей равное result.

Процедура

blockwrite (f, buf, count[, result]);

– записывает count элементов в файл f из переменной buf. **(file, file of)**

Здесь f – ифп; buf – переменная любого типа, из которой происходит запись; count – выражение, определяющее количество записываемых элементов; result–число фактически записанных элементов.

Напр.: blockwrite (F1, k, x, c); Размер записи нетипизированного файла = буферу. Если result опущен, то при несовпадении количества прочитанных элементов с count – ошибка. За один вызов процедуры можно считать не более 64 Кбайт. После выполнения текущая позиция смещается на result.

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Функция filepos (f); – возвращает текущую позицию в файле f.
Здесь f – ифп. (file, file of)
Напр.: filepos (F1); Если текущей позицией является начало файла, функция возвращает значение 0, а если – конец файла, то размер файла.

Процедура seek (f, n); – перемещает текущую позицию в файле f к элементу n.
Здесь f – ифп, n – порядковый номер элемента, целое число. (file, file of)
Напр.: seek (F1, 10); Номер первого элемента файла – 0. Чтобы расширить файл, можно переместить текущую позицию в конец файла: seek (F1, filesize(f));,- а затем добавить элементы.

Процедура truncate (f); – усекает размер файла f до текущей позиции.
Здесь f – ифп. (file, file of)
Напр.: trancate (F1); Все элементы после текущей позиции в файле F1 удаляются и текущая позиция становится концом файла.

Функция eof [(f)]; – возвращает True, если при чтении текущая позиция находится за последним элементом файла f или файл пуст, иначе False.
Здесь f – ифп.
Напр.: eof (F1); Если имя файла f опущено, используется файл Input.

Функция seekeof [(f)]; – возвращает для файла f True, если при чтении текущая позиция находится за последним элементом файла f или файл пуст, иначе False.
Здесь f – ифп. (text)
Аналогична функции eof(f); – но пропускает все пробелы, знаки табуляции и EOL, как лидирующие, так и от последнего значащего символа до конца файла.
Напр.: seekeof (F1); Можно исп. при считывании числовых значений из текстового файла.

Функция eoln [(f)]; – возвращает для файла f True, если при чтении текущая позиция находится за последним элементом стоки или строка пуста, иначе False.
Здесь f – ифп. . (text)
Текущая позиция – на EOL – True, иначе False.
Напр.: eoln (F1); Если имя файла f опущено, используется файл Input.

Функция seekeoln [(f)]; – возвращает для файла f True, если при чтении текущая позиция находится за последним элементом стоки или строка пуста, иначе False.
Здесь f – ифп. . (text)
Аналогична функции eoln(f); - но пропускает все пробелы и знаки табуляции, как лидирующие, так и от последнего значащего символа до конца строки.
Напр.: seekeoln (F1); Можно исп. при считывании числовых значений из текстового файла. Если имя файла f опущено, используется файл Input.

Элементы ЯПВУ. ФАЙЛЫ

Pascal

Pascal

Подпрограммы для работы с файлами

Процедура settexbuf (f, buf[, size]); – переназначает буфер для обмена с текстовым файлом.

Здесь f – ифп., buf – переменная для размещения буфера (любого типа – нетипизированная), size – выражение – размер буфера в байтах. Settexbuf действует до нового assign(f). (text)

Напр.: SetTextBuf(F1, bf, 512); По умолчанию size равна sizeoff(buf). Рекомендуется задавать settextbuf до открытия файла (или сразу после открытия), чтобы не потерять данные буфера. Оптимальный размер буфера равен размеру сектора диска.

Функция ioresult; – возвращает – код ошибки последней операции ввода/вывода.

При нормальном завершении код – 0.

Напр.: IOResult; Работает только при включенном режиме проверки ошибок ввода/вывода, ключ компиляции {$I-} (по умолчанию ключ {$I+} – выключен).

Стандартные текстовые файлы

input – стандартный файл ввода (по умолчанию – клавиатура).

output – стандартный файл вывода (по умолчанию – экран дисплея).

Разрешено переназначение стандартных файлов ввода/вывода:

```
assign (output, 'd:\BP\outfile.dat');
```

ФАЙЛЫ

Pascal | Практические занятия | Pascal

Задание 1. Создать на диске Z:\ файл numbers.txt, записать в него 5 введенных с клавиатуры целых чисел.

```
Program file_5num;
(* Создает на диске Z:\ файл numbers.txt и записывает в
него 5 целых чисел введенных с пользователем с
клавиатуры *)
var
 f : text;        (* текстовый файл *)
 n : integer;     (* вводимое число *)
 i : integer;     (* счетчик чисел *)
begin
  writeln ('Создание и заполнение файла numbers.txt');
  assign (f, 'z:\numbers.txt');   (* Связь с файлом *)
  rewrite (f); (*Созд. и откр. файл в режиме перезаписи*)
   writeln ('Введите 5 целых чисел, нажимая Enter после
каждого числа');
    for i:=1   to  5  do
     begin
       write ('->');
       readln (n); (* чтение числа из буфера клавиатуры*)
       writeln (f, n); (* запись считанного числа в файл *)
     end;
     close (f);     (* закрыть файл *)
       writeln  ('Введенные числа записаны в файл ',
'z:\numbers.txt');
   readln;
end.
```

Задание 2. Вывести на экран содержимое файла numbers.txt.

```
Program file5displ;
(* Выводит на экран файл z:\numbers.txt *)
var
 f : text;        (* текстовый файл *)
 n : integer;  (* вводимое число *)
begin
  writeln ('Содержимое файла numbers.txt');
  writeln ('---------------------------------------');
  assign (f, 'z:\numbers.txt');  (*Связь с файлом*)
  reset (f);                  (* Открыть файл для чтения *)
   while not EOF(f) do  (*Выполнять до конца файла*)
    begin
      readln (f, n);  (* Читать число из файла *)
     writeln (n); (*Вывести  прочитанное числа на экран *)
    end;
   close (f);     (* закрыть файл *)
   writeln ('---------------------------------------');
   readln;
end.
```