**Нетипизированные файлы (или файлы без типа)**

Нетипизированные файлы - это файлы, поддержка которых осуществляется с максимально возможной скоростью. Введение таких файлов в систему Турбо Паскаль было вызвано стремлением повысить эффективность программ, участвующих в интенсивном обмене с внешними наборами данных.

Эти файлы в отличие от уже рассмотренных не имеют строго определенного типа.

Для определения в программе нетипизированного файла служит зарезервированное слово file:

|  |
| --- |
| Var   MyFile : file; |

Внутренняя поддержка таких файлов выглядит наиболее близкой к аппаратной поддержке работы с внешними носителями. За счет этого достигается максимально возможная скорость доступа к наборам данных. Для нетипизированных файлов не нужно терять время на преобразование типов и поиск управляющих последовательностей, достаточно считать содержимое файла в определенную область памяти.

Нетипизированный файл является файлом прямого доступа, что говорит о возможности одновременного использования операций чтения и записи.

Для таких файлов самым важным параметром является длина записи в байтах. Открытие нетипизированного файла с длиной записи в 1 байт можно выполнить следующим образом:

rewrite(MyFile, 1) или reset(MyFile, 1)

Второй параметр, предназначенный только для использования с нетипизированными файлами, задает длину записи файла на сеанс работы.

Особенность аппаратной поддержки заключается в том, что при обращении к внешнему устройству минимально возможным объемом для считывания являются 128 байт. В стремлении добиться наибольшей эффективности файловых операций в Турбо Паскале принято соглашение, по которому длина записи нетипизированного файла по умолчанию составляет 128 байт. Поэтому после открытия файла с помощью вызовов:

rewrite(MyFile) или reset(MyFile)

все процедуры и функции, обслуживающие файлы прямого доступа, работают с записями длиной 128 байт. Каждый пользователь для своих программ может выбрать наиболее подходящий размер записи.

Турбо Паскаль не накладывает каких-либо ограничений на длину записи нетипизированного файла, за исключением требования положительности и ограничения максимальной длины 65535 байтами (емкость целого типа word). При этом следует учитывать два обстоятельства.

Во-первых, для обеспечения максимальной скорости обмена данными следует задавать длину, которая была бы кратна длине физического сектора дискового носителя информации (512 байт).

С другой стороны, нужно помнить, что общий размер файла может не быть кратен выбранному размеру записи (последняя запись может быть неполной). Для того, чтобы гарантированно обеспечить полное чтение всего файла, рекомендуется установить размер записи равным 1.

Более того, фактически пространство на диске выделяется любому файлу порциями - кластерами, которые в зависимости от типа диска могут занимать 2 и более смежных секторов. Как правило, кластер может быть прочитан или записан за один оборот диска, поэтому наивысшую скорость обмена данными можно получить, если указать длину записи, равную длине кластера.

При работе с нетипизированными файлами могут применяться все процедуры и функции, доступные типизированным файлам. Напомним эти процедуры и функции.

assign (МуFilе, 'с:\МуDirectory\result.dat') - процедура связывания логической файловой переменной МуFilе с конкретным физическим файлом на дисковом носителе информации;

closе (МуFilе) - процедура, закрывающая открытый файл;

rewrite (МуFilе) - процедура, создающая новый файл и открывающая его для записи или чтения; эта процедура имеет дополнительный параметр при работе с нетипизированными файлами, который будет рассмотрен позднее;

reset (МуFilе) - процедура, открывающая существующий файл данных; эта процедура имеет дополнительный параметр при работе с нетипизированными файлами, который будет рассмотрен позднее;

eof (МуFilе) - логическая функция, проверяющая, достигнут ли конец файла;

seek (МуFilе, n) - процедура, позволяющая явно изменить значение текущего указателя файла, установив его на элемент с номером n;

filesize(МуFilе) - функция, возвращающая позицию указателя по файлу; нумерация начинается с нуля;

filepos (МуFilе) - функция, возвращающая количество элементов файла;

rename(МуFilе, FileName) - процедура, позволяющая переименовать существующий файл;

truncate(МуFilе) - процедура, позволяющая удалить часть существующего файла, начиная с текущей позиции;

erase(МуFilе) - процедура, стирающая указанный файл,

Вы должны были заметить, что в списке нет процедур read и write. Для чтения информации из нетипизированного файла и записи информации в него только для данного типа файлов в Турбо Паскаль введены две новые процедуры, поддерживающие операции ввода-вывода с более высокой скоростью.

*Процедура BlockRead*

Формат обращения:

|  |
| --- |
| blockread(Var F : file; Var Buf; Kolblocks : word; result : word); |

Процедура считывает из файла F определенное число блоков в память, начиная с первого байта переменной Buf.

Параметр Buf представляет любую переменную, которая будет участвовать в обмене данными с дисками. Эту переменную нужно описать в программе так, чтобы ее размер не был меньше размера записи, установленного в параметрах rewrite или reset (как правило, для этих целей используется некоторый массив).

Параметр Kolblocks задает число считываемых блоков, которые должны быть прочитаны за одно обращение к диску.

Параметр result является необязательным и содержит после вызова процедуры число действительно считанных записей.

Использование параметра result подсказывает, что число считанных блоков может быть меньше, чем задано параметром Kolblocks. Если result указан при вызове, то ошибки ввода-вывода в такой ситуации не произойдет. Для отслеживания этой и других ошибок чтения можно использовать опции {$I-}, {$I+} и функцию IOresult.

Кроме того, что переменная F должна быть описана как нетипизированный файл, она должна быть связана с конкретным физическим диском процедурой assign. Файл должен быть открыт процедурой reset.

*Процедура BlockWrite*

Формат обращения:

|  |
| --- |
| blockwrite(Var F : file; Var Buf; Kolblocks : word; result : word); |

Процедура предназначена для быстрой передачи в файл F определенного числа записей из переменной Buf. Все параметры процедуры blockwrite аналогичны параметрам процедуры blockread. Разница лишь в том, что файл должне быть подготовлен для записи процедурой rewrite. Содержимое переменной Buf целиком помещается в файл, начиная с текущей записи.

Обе процедуры выполняют операции ввода-вывода блоками. Объем блока в байтах определяется по формуле:

Объем = Kolblocks \* recSize,

где recSize - размер записи файла, заданный при его открытии. Суммарный объем разового обмена не должен превышать 64 Кбайт. Помимо скорости передачи данных преимущество этих процедур заключается в возможности пользователя самостоятельно определять размер буфера для файловых операций. Эта возможность играет значительную роль в тех задачах, где необходимо жесткое планирование ресурсов. Программист должен позаботиться о том, чтобы длина внутреннего представления переменной Buf была достаточной для размещения всех байт при чтении информации с диска.

Дело в том, что чтение информации из файла в буфер, равно как и запись из буфера в файл, производится без типового контроля. Поэтому несоблюдение указанного условия может привести к порче соседних с буфером данных или к помещению на файл посторонней информации.

Если при чтении указана переменная Buf недостаточной длины или если в процессе записи на диск не окажется нужного свободного пространства, то произойдет следующее. Если последний параметр result в этих вызовах не задан, то возникает ошибка ввода-вывода; если параметр result задан, то ошибка не будет зафиксирована, а после выполнения процедуры его значение не будет совпадать с значением параметра Kolblocks. Последнее обстоятельство можно проверить, сравнив два указанных значения.

После завершения процедуры указатель смещается на result записей.

Список литературы:

<http://www.pascaler.ru/pascal/filetype/netipiz/1/>