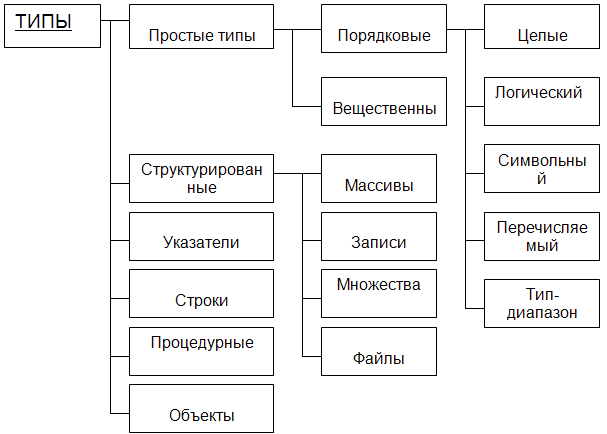
**билет 11 тема 2**

**Структурированные типы данных: массивы, записи, множества в языке Турбо Паскаль**

Структурированные типы данных определяют наборы однотипных или разнотипных компонент.



**СТРУКТУРИРОВАННЫЕ ТИПЫ**

Любой из структурированных типов характеризуется множественностью образующих этот тип элементов, т.е. переменная или константа структурированного типа всегда имеет несколько компонентов. Каждый компонент, в свою очередь, может принадлежать структурированному типу, что позволяет говорить о возможной вложенности типов. В Турбо Паскале допускается произвольная глубина вложенности типов, однако суммарная длина любого из них во внутреннем представлении не должна превышать 65520 байт.

Массивы в Турбо Паскале во многом схожи с аналогичными типами данных в других языках программирования. Отличительная особенность массивов заключается в том, что все их компоненты суть данные одного типа (возможно структурированного). Эти компоненты можно легко упорядочить и обеспечить доступ к любому из них простым указанием порядкового номера.

Описание массива задаётся следующим образом:<имя типа> = array [<сп.инд.типов>] of <тип>

Здесь <имя типа> - правильный индификатор;

Array, of – зарезервированные слова (массив, из);

<спюиндютипов> - список из одного или нескольких индексных типов, разделённых запятыми; квадратные скобки, обрамляющие список, - требование синтаксиса;

<тип> - любой тип Турбо Паскаля.

В качестве индексных типов в Турбо Паскале можно использовать любые порядковые типы, кроме LongInt и типов-диапазонов с базовым типом LongInt. Глубина вложенности структурированных типов вообще, а следовательно, и массивов – произвольная, поэтому количество элементов в списке индексов типов (размерность массива) не ограничено, однако суммарная длина внутреннего представления любого массива не может быть больше 65520 байт.

ЗАПИСИ

Запись – это структура данных, состоящая из фиксированного числа компонентов, называемых полями записи. В отличие от массива, компоненты (поля) записи могут быть различного типа. Чтобы можно было ссылаться на тот или иной компонент записи, поля именуются.

Структура объявления типа записи такова:<имя типа> = RECORD <спюполей> END

Здесь <имя типа> - правильный идентификатор;

RECORD, END – зарезервированные слова (запись, конец);

<сп.полей> - список полей; представляет собой последовательность разделов записи, между которыми ставится точка с запятой.

МНОЖЕСТВА Множества – это набор однотипных логических связанных друг с другом объектов. Характер связей между объектами лишь подразумевается программистом и никак не контролируется Турбо Паскалем. Количество элементов, входящих в множество, может меняются в пределах от 0до 256 (множество, не содержащее элементов, называется пустым).именно непостоянством количества своих элементов множества отличаются от массивов и записей.

Два множества считаются эквивалентными тогда и только тогда, когда все их элементы одинаковы, причём порядок следования элементов множества безразличен. Если все элементы одного множества входят также и в другое, говорят о включении первого множества во второе.

Описание типа множества имеет вид:<имя типа> = SET OF <баз.тип>

Здесь <имя типа> - правильный индификатор;

SET, OF – зарезервированные слова (множество, из);

<баз.тип> - базовый тип элементов множества, в качестве которого может использоваться любой порядковый тип, кроме WORD, INTEGER и LONGINT.

Для задания множества используется так называемый конструктор множества: список спецификаций элементов множества, отделяемых друг от друга запятыми; список обрамляется квадратными скобками. Спецификациями элементов могут быть константы или выражения базового типа, а также – тип-диапазон того же базового типа.

СТРОКИ

Тип STRING (строка) в Турбо Паскале широко используется для обработки текстов. Он во многом похож на одномерный массив символов ARRAY [0..N] OF CHAR, однако, в отличие от последнего, количество символов в строке – переменной может меняться от 0 до N, где N – максимальное количество символов в строке. Значение N определяется объявлением типа STRING[N] N и может быть любой константой порядкового типа, но не больше 255. Турбо Паскаль разрешает не указывать N, в том случае длина строки принимается максимально возможной, а именно N=255.

Строка в Турбо Паскале трактуется как цепочка символов. К любому символу в строке можно обратиться точно так же, как к элементу одномерного массива ARRAY [0..N] OF CHAR.

ФАЙЛЫ

Под файлом понимается либо именованная область внешней памяти ПК, либо логическое устройство – потенциальный источник или приёмник информации. Любой файл имеет три характерные особенности. Во-первых, у него есть имя, что даёт возможность программе работать одновременно с несколькими файлами. Во-вторых, он содержит компоненты одного типа. Типом компонентов может быть любой тип Турбо Паскаля, кроме файлов. Иными словами, нельзя создать "файл файлов". В-третьих, длина вновь создаваемого файла никак не оговаривается при его объявлении и ограничивается только ёмкостью устройств внешней памяти. Файловый тип или переменную файлового типа можно задать одним из трёх способов:

<имя>= FILE OF <тип>;

<имя>=TEXT;

<имя> = FILE;

Здесь <имя> - имя файлового типа (правильный индификатор);

FILE, OF – зарезервированные слова (файл, из);

TEXT – имя стандартного типа текстовых файлов;

<тип> - любой тип Турбо Паскаля, кроме файлов.

В зависимости от способа объявления можно выделить три вида файлов:

• типизированные файлы (задаются предложением FILE OF…);

• текстовые файлы (определяются типом TEXT);

• нетипизированные файлы (определяются типом FILE).

ПРОЦЕДУРНЫЕ ТИПЫ

Процедурные типы — это нововведение фирмы Borland (в стандартном Паскале таких типов нет). Основное назначение этих типов — дать программисту гибкие средства передачи функций и процедур в качестве фактических параметров обращения к другим процедурам и функциям.

Для объявления процедурного типа используется заголовок процедуры (функции), в котором опускается её имя, например:

type

Proc = procedure;

Proc1 = procedure(var X, Y: Integer);

StrProc = procedure(S: String);

MathFunc = function(X: Real): Real:

DeviceFunc = function(var F: Text): Integer;

MaxFunc = function(A, B: Real; F: MathFunc): Real;

Как видно из приведённых примеров существует два вида процедурных типов: тип-процедура и тип-функция.

В программе могут быть объявлены переменные процедурных типов, например, так:

Var

P1: Proc1;

F1, f2: MathFunc;

Ap: array [1..N] of Proc1

Переменным процедурных типов допускается присваивать в качестве значений имена соответствующих подпрограмм. После такого присваивания имя переменной становится синонимом имени подпрограммы.

ТИП ОБЪЕКТ

Тип объект — это структура, состоящая из фиксированного числа компонент. Каждая компонента — это или поле, которое содержит данные определенного типа, или метод, который производит операции над объектом. Аналогично объявлению переменных, объявление поля, указывает тип данных поля и идентификатор имени этого поля; и аналогично объявлению процедуры или функции объявление метода указывает заголовок процедуры, функции, конструктора или деструктора.

Тип объект может наследовать компоненты от другого типа объекта.

Сфера действия типа объекта состоит из него самого и всех его порожденных типов.

В отличие от других типов, тип объект может быть объявлен в части объявления типов в самой внешней части программы или модуля.

Так, тип объекта не может быть объявлен в части объявления переменных или внутри процедуры, функции или метода.

ТИП УКАЗАТЕЛЬ

Тип указатель (ссылочный тип) определяет множество значений, которые указывают на динамические переменные определенного типа, называемого базовым типом. Переменная с типом указатель содержит адрес динамической переменной в памяти.

Если базовый тип является еще не описанным идентификатором, то он должен быть описан в той же самой части описания типов, что и тип указатель.

Переменной-указателю можно присвоить значение с помощью процедуры New, операции @ или функции Ptr. Процедура New отводит новую область памяти в динамически распределяемой области для динамических переменных и сохраняет адрес этой области в переменной указателя. Операция @ ориентирует переменную указателя на область памяти, содержащую существующую переменную, включая и те переменные, которые имеют идентификаторы. Функция Ptr ориентирует переменную указателя на определенный адрес в памяти.

Зарезервированное слово nil обозначает константу со значением указателя, которая ни на что не указывает.

Встроенный указатель типа обозначает нетипизованный указатель, то есть указатель, который не указывает ни на какой определенный тип. Переменные типа Pointer могут быть разыменованы; указание символа ^ после такой переменной вызывает появление ошибки. Как и значение, обозначаемое словом nil, значения типа Pointer совместимы со всеми другими типами указателей. В разделе "Указатели и динамические переменные" в Главе 4 вы можете найти синтаксис ссылки на динамические переменные, которые указываются с помощью указателя-переменной.

ТИПИЗИРОВАННЫЕ КОНСТАНТЫ

В Турбо Паскале допускается использование типизированных констант. Они задаются в разделе объявления констант следующим образом:

<индификатор> : <тип> = <значение>

паскаль массив константа программа

Здесь <индификатор> - индификатор константы;

<тип> - тип константы;

<значение> - значение константы.

Типизированным константам можно присваивать другие значения в ходе выполнения программы, поэтому фактически они представляют собой переменные с начальными значениями. Типизированная константа приобретает указанное в её объявлении значение, т.е. инициируется, лишь один раз: к моменту начала работы программы. При повторном входе в блок (процедуру или функцию), в котором она объявлена, инициация типизированной константы не производится и она сохраняет то значение, которое имела к моменту выхода из блока.

Типизированные константы могут быть любого типа, кроме файлов. Нельзя также объявить типизированную константу – запись, если хотя бы одно из её полей является полем файлового типа.

Поскольку типизированная константа фактически не отличается от переменной, её нельзя использовать в качестве значения при объявлении других констант или границ типа-диапазона.

КОНСТАНТЫ ПРОСТЫХ ТИПОВ И ТИПА STRING

Объявление таких констант обычно не вызывает трудностей, так как в качестве их значения используется нетипизированные константы или их идентификаторы.

КОНСТАНТЫ-МАССИВЫ

В качестве начального значения типизированной константы-массива используются список констант, отделённых друг от друга запятыми; список заключается в круглые скобки.

При объявлении многомерных констант-массивов множество констант, соответствующих каждому измерению, заключается в дополнительные круглые скобки и отделяется от соседнего множества запятыми. В результате образуются вложенные структуры множеств, причём глубина вложения должна соответствовать количеству измерений (размерности) массива. Самые внутренние множества констант связываются с измерением самого правого индекса массива.

КОНСТАНТЫ-ЗАПИСИ

Определение константы-записи имеет следующий вид:

<индификатор> : <тип> (<сп.знач.полей>)

Здесь <индификатор> – индификатор константы;

<тип> – тип записи;

<сп.инд.полей> – список значений полей.

Список значений полей представляет собой список из последовательностей вида: имя поля, двоеточие и константа. Элементы списка отделяются друг от друга двоеточиями.

КОНСТАНТЫ-МНОЖЕСТВА

Значение типизированной константы-множества задаётся в виде правильного конструктора множеств.

КОНСТАНТЫ-УКАЗАТЕЛИ

Единственным значением типизированной константы-указателя может быть только NIL.