Лекция 5. СТРУКТУРИРОВАННЫЕ ТИПЫ ДАННЫХ

Тип данных (Data type)

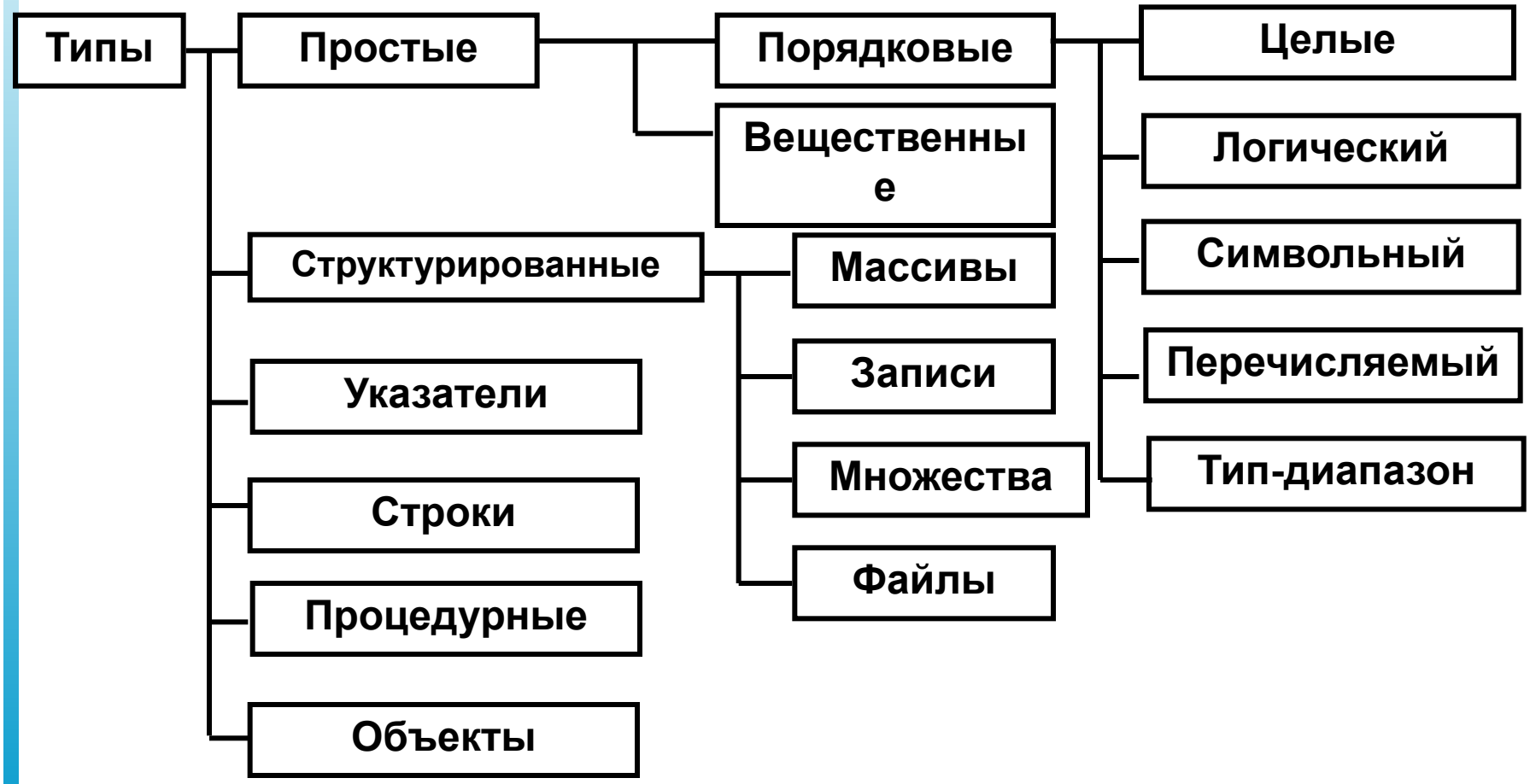
Тип данных - характеристика набора данных, которая определяет:

- диапазон возможных значений данных из набора;
- допустимые операции, которые можно выполнять над этими значениями;
- способ хранения этих значений в памяти.

Различают:

- простые типы данных: целые, действительные числа и др.;
- составные типы данных: массивы, файлы и др.

Типы


Типы
Простые
Порядковые
Целые
Логический
Символьный
Перечисляемый
Тип-диапазон
Вещественные
Структурированные
Массивы
Записи
Множества
Файлы
Указатели
Строки
Процедурные
Объекты

Структурированные типы данных. Массив.

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 его порядкового номера.

Описание типа массива задается следующим образом:

<имя типа> = ARRAY [<сп.инд.типов>] OF <тип>;

Здесь ***<имя типа>*** - правильный идентификатор; ***ARRAY, OF*** - зарезервированные слова (массив, из); ***<сп.инд.типов>*** - список из одного или нескольких индексных типов, разделенных 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***. Определить переменную как массив можно и непосредственно при описании этой переменной, без предварительного описания типа массива, например:

```
Var
  a,b : array [1..10, -2..2] of Real;
```

Обычно в качестве индексного типа используется тип-диапазон, в котором задаются границы изменения индексов. Так как тип ***<тип>***, идущий за словом ***OF***, - любой тип Турбо Паскаля, то он может быть, в частности, и другим массивом.

Структурированные типы данных. Множество.

Множества - это наборы однотипных логически связанных друг с другом объектов. Характер связей между объектами лишь подразумевается программистом и никак не контролируется Турбо Паскалем. Количество элементов, входящих в множество, может меняться в пределах от 0 до 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, причем порядок следования элементов в множестве безразличен. Если все элементы одного множества входят также и в другое, говорят о включении первого множества во второе. Пустое множество включается в любое другое.

Описание типа множества имеет вид:

<имя типа> = SET OF <баз.тип>;

Здесь ***<имя типа>*** - правильный идентификатор;

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

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

Задание множества:

```
type
  digitChar = set of '0'..'9';
  digit = set of 0. .9;
var
  sl,s2,s3 :digitChar;
  s4,s5,s6 :digit;
```

Структурные типы данных. Множество

Над множествами определены следующие операции:

* пересечение множеств; результат содержит элементы, общие для обоих множеств;

\+ объединение множеств; результат содержит элементы первого множества, дополненные недостающими элементами из второго множества:

\- разность множеств; результат содержит элементы из первого множества, которые не принадлежат второму:

= проверка эквивалентности; возвращает ***TRUE***, если оба множества эквивалентны;

<> проверка неэквивалентности; возвращает ***TRUE***, если оба множества неэквивалентны;

<= проверка вхождения; возвращает ***TRUE***, если первое множество включено во второе;

>= проверка вхождения; возвращает ***TRUE***, если второе множество включено в первое;

IN проверка принадлежности; в этой бинарной операции первый элемент - выражение, а второй - множество одного и того же типа; возвращает ***TRUE***, если выражение имеет значение, принадлежащее множеству:

Дополнительно к этим операциям можно использовать две процедуры.

INCLUDE - включает новый элемент во множество. Обращение к процедуре: ***INCLUDE (S,I)***

Здесь ***S*** - множество, состоящее из элементов базового типа ***TSetBase***;

I - элемент типа ***TSetBase***, который необходимо включить во множество.

EXCLUDE - исключает элемент из множества. Обращение: ***EXCLUDE(S,I)***

Параметры обращения - такие же, как у процедуры ***INCLUDE***.

Структурированные типы данных. Файл

Файловый тип или переменную файлового типа можно задать одним из трех способов:
<имя> = FILE OF <тип>; {типизированный файл}
<имя> = TEXT; {текстовый файл}
<имя> = FILE; {нетипизированный файл}
Здесь ***<имя>*** - имя файлового типа (правильный идентификатор);
FILE, OF - зарезервированные слова (файл, из);
TEXT - имя стандартного типа текстовых файлов;
<тип> - любой тип Турбо Паскаля, кроме файлов.

Файловая переменная связывается с именем файла в результате обращения к стандартной процедуре ***ASSIGN***:
ASSIGN (<ф.п.>, <имя файла или л.у.>);
Здесь ***<ф.п.>*** - файловая переменная (правильный идентификатор, объявленный в программе как переменная файлового типа);
<имя файла или л.у.> - текстовое выражение, содержащее имя файла или логическое устройство.

Структурированные типы данных. Файл

Инициировать файл означает указать для этого файла направление передачи данных. В Турбо Паскале можно открыть файл для чтения, для записи информации, а также для чтения и записи одновременно.

Для чтения файл инициируется с помощью стандартной процедуры ***RESET***:

RESET (<ф.п.>);

Здесь ***<ф.п.>*** - файловая переменная, связанная ранее процедурой ***ASSIGN*** с уже существующим файлом или логическим устройством-приемником информации.

Стандартная процедура ***REWRITE (<ф.п.>);*** инициирует запись информации в файл или в логическое устройство, связанное ранее с файловой переменной ***<ф.п.>.*** Процедурой ***REWRITE*** нельзя инициировать запись информации в ранее существовавший дисковый файл: при выполнении этой процедуры старый файл уничтожается и никаких сообщений об этом в программу не передается. Новый файл подготавливается к приему информации и его указатель принимает значение 0.

Стандартная процедура ***APPEND (<ф.п.>)*** инициирует запись в ранее существовавший текстовый файл для его расширения, при этом указатель файла устанавливается в его конец. Процедура ***APPEND*** применима только к текстовым файлам, т.е. их файловая переменная должна иметь тип ***TEXT***. Процедурой ***APPEND*** нельзя инициировать запись в типизированный или нетипизированный файл.

Если текстовый файл ранее уже был открыт с помощью ***RESET*** или ***REWRITE***, использование процедуры ***APPEND*** приведет к закрытию этого файла и открытию его вновь, но уже для добавления записей.

Структурные типы данных. Запись

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 <сп.полей> 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.

Чтобы упростить доступ к полям записи, используется оператор присоединения ***WITH***:

WITH <переменная> DO <оператор>;

Здесь ***WITH, DO*** - ключевые слова (с, делать);

<переменная> - имя переменной типа запись, за которым, возможно, следует список вложенных полей;

<оператор> - любой оператор Турбо Паскаля.

Задание записи и объявление переменных типа запись:

```
type
  BirthDay = record
  day,month : Byte;
  year : Word
  end;
var
  a,b : Birthday;
```