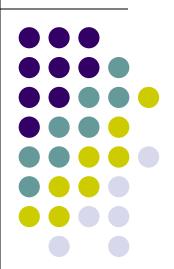
Система типов данных в языке Паскаль

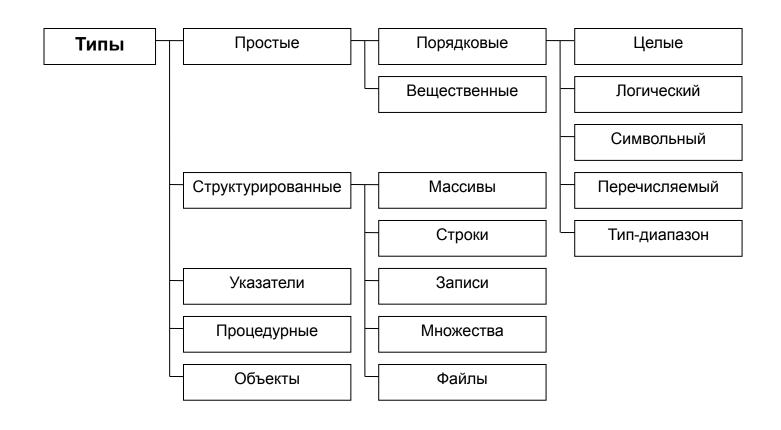
Перечисляемый тип, типдиапазон, множество



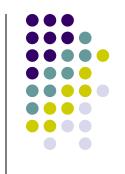
Система типов языка Паскаль



Данные, обрабатываемые программой, записанной на языке TurboPascal, принадлежат к одному из следующих *типов*, классификация которых представлена на схеме:



Перечисляемый тип



Перечисляемый тип задается перечислением тех значений, которые он может получать. Определяется как упорядоченный набор идентификаторов, заданных путем их перечисления.

Например:

Type Colors = (red, green, blue);

Var Col : Colors;

Переменная Col может принять одно из трех значений: red, green, blue.

Таким образом, каждое значение именуется некоторым идентификатором и располагается в списке, ограниченном круглыми скобками. Идентификаторы перечисляются через запятую.





Значения перечисляемого типа *упорядочены*: первое имеет порядковый номер 0, второе — 1 и т.д. Можно использовать следующие *стандартные функции*:

Ord (x) – возвращает порядковый номер элемента x;

Succ (x) – возвращает значение, следующее за x;

Pred (x) – возвращает значение, предшествующее x.





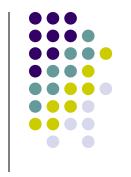
```
В приведенном выше примере:
Type Colors = ( red, green, blue );
Var Col : Colors;
Begin
 a := ord (red); { Значение переменной a = 0 }
 col := succ (green); { col = blue }
 col := pred (col); { col = green }
```





- Значения перечисляемого типа можно сравнивать: сравниваются их порядковые номера.
- К данным этого типа нельзя применять стандартные команды ввода (Readln) и вывода (Write).
- **Назначение** перечисляемого типа сделать текст программы более **наглядным** (читабельным).





Тип-диапазон называют также *ограниченным* и *интервальным* типом.

Тип-диапазон есть подмножество своего базового типа, в качестве которого может выступать любой порядковый тип, кроме самого типа-диапазона (т.е. типы Integer, Boolean, Char, перечисляемый тип).





Диапазон задается *границами* своих значений внутри базового типа:

<минимальное значение>.. <максимальное значение>

Причем минимальное значение должно быть больше либо равно максимальному.





Например:

```
Туре Digit = '0'..'9'; { тип-диапазон, ограничение наложено на Char } Year = 1900..2007; { тип-диапазон, ограничение на Integer } Week = {mon, tues, wed, thur, fri, sat, sun); { перечисляемый тип (дни недели)}

Var d : Digit; y : Year; m : 1..12; { переменная m относится к ограниченному типу}
```

work : mon .. fri; { тип-диапазон, ограничение наложено на Week }

Тип-диапазон



Тип-диапазон наследует все свойства своего базового типа.

Назначение типа-диапазона:

- наглядность программы;
- дополнительная проверка корректности данных.

Множество (множественный тип)



Множество – это *неупорядоченный* набор *однотипных* элементов.

Количество элементов в множестве — от 0 до 256.

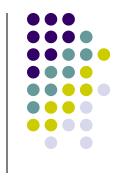
Пустое множество — это множество, которое не содержит ни одного элемента.

Два множества эквивалентны, если все их элементы одинаковы.

Первое множество *включено* во второе, если все элементы первого множества являются также элементами второго.

Пустое множество включено в любое другое.

Множество



Описание множественного типа:

Туре <имя типа> = Set Of <базовый тип>;

В качестве базового типа может использоваться любой порядковый тип, мощность которого не больше 256. Из стандартных — это Char, Boolean. Integer напрямую в качестве базового типа для множества использовать нельзя. Сначала нужно описать тип-диапазон (не более 256 чисел).

Множество

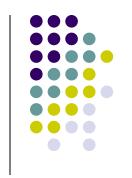


Например:

```
Type digit = Set Of 0..9;
setchar = Set Of Char;
```

Var d1, d2 : digit; c : setchar;





Для задания множества (т.е. присваивания ему некоторых значений) используется конструктор множества — это список элементов множества, разделенных запятыми. Список ограничен квадратными скобками.

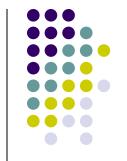
В качестве элементов могут быть:

константы — <u>базового типа</u> диапазоны

Множество

Например:

```
d1 := [ 0..3 , 6 ];
d2 := [ ];
c := ['a' .. 'z' , 'A' .. 'Z' ];
```



Операции над множествами:

*	<i>пересечение</i> множеств: результат содержит элементы, общие для двух множеств
+	объединение множеств: результат содержит элементы 1-го множества, дополненные недостающими элементами из 2-го множества
-	разность множеств: результат содержит элементы из 1-го множества, которых нет во 2-м множестве
=	проверка <i>эквивалентности</i>
<>	проверка <i>неэквивалентности</i>
<= >=	проверка <i>вхождения</i>
in	проверка <i>принадлежности</i> элемента множеству (элемент задается как выражение соответствующего типа)

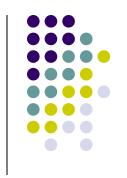
Стандартные процедуры:



- Include (S, i); включает элемент і в множество S;
- Exclude (S, i); исключает элемент i из множества S.

Процедуры исполняются быстрее, чем операции + и -.

Решение задач



Пример 1. Составить программу, которая выведет на экран числа от 1 до 9 в случайном порядке.

Алгоритм: Будем использовать **множество** для хранения тех чисел, которые уже выведены на экран. Сначала это множество пустое. Затем девять раз должны повторяться следующие действия: выбор случайного числа (функция Random) осуществляется до тех пор, пока не будет выбрано число, которого нет в множестве; после чего это число выводится на экран и включается в множество.