

# Множества

***Множество*** - это ограниченная совокупность различных элементов одного типа

Множественный тип описывается с помощью служебных слов **Set of**, например:

```
type M = Set of B;
```

Здесь M - множественный тип, B - базовый тип.

Пример описания переменной множественного типа:

```
type
```

```
    M = Set of 'A'..'D';
```

```
var
```

```
    MS: M;
```

Возможно прямое описание множества: `var C: Set of`

`0..7;`

***Все элементы множества должны принадлежать одному из порядковых типов, содержащему не более 256 значений. Этот тип называется базовым типом множества. Базовый тип задается диапазоном или перечислением.***

Область значений типа множество — набор всевозможных подмножеств, составленных из элементов базового типа.

В выражениях на языке Паскаль значения элементов множества указываются в квадратных скобках: [1,2,3,4],

['a','b','c'], ['a'..'z'].

Если множество не имеет элементов, оно называется пустым и обозначается как [].

Примеры описания множеств:

Var A, D : Set Of Byte;

B : Set Of 'a'..'z';

C : Set Of Boolean;

Нельзя вводить значения во множественную переменную процедурой ввода и выводить процедурой вывода.

Множественная переменная может получить конкретное значение только в результате выполнения оператора присваивания:

<множественная переменная> := <множественное выражение>;

Например:

A := [50, 100, 150, 200];

B := ['m', 'n', 'k']; C := [True, False];

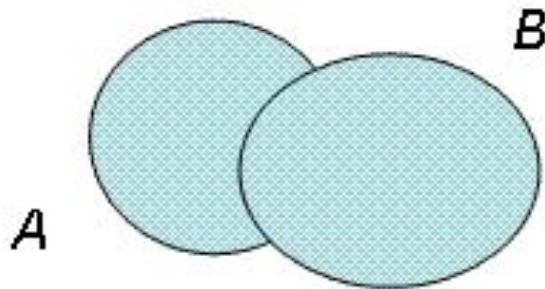
D := A;

# Операции над множествами

**Объединением** двух множеств  $A$  и  $B$  называется множество, состоящее из элементов, входящих хотя бы в одно из множеств  $A$  или  $B$ . Знак операции объединения в Паскале «**+**».

Примеры:

- 1)  $[1, 2, 3, 4] + [3, 4, 5, 6] \Rightarrow [1, 2, 3, 4, 5, 6]$
- 2)  $[] + ['a'..'z'] + ['A'..'E', 'k'] \Rightarrow ['A'..'E', 'a'..'z']$
- 3)  $[5 < 4, \text{true and false}] + [\text{true}] \Rightarrow [\text{false}, \text{true}]$

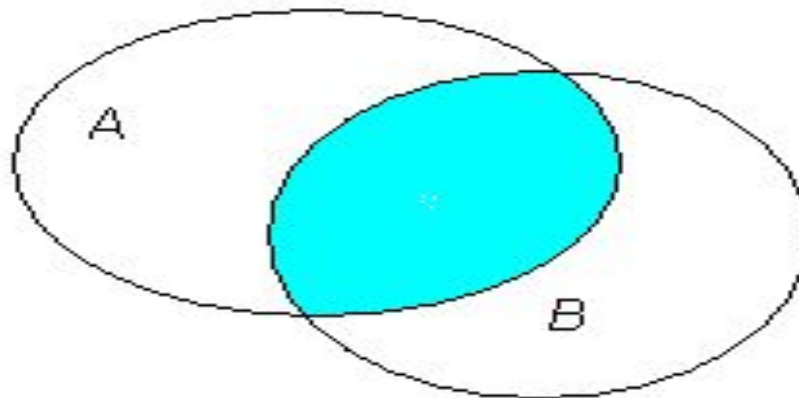


**Пересечением** двух множеств A и B называется множество, состоящее из элементов, одновременно входящих во множество A и во множество B.

Знак операции пересечения в Паскале «\*»

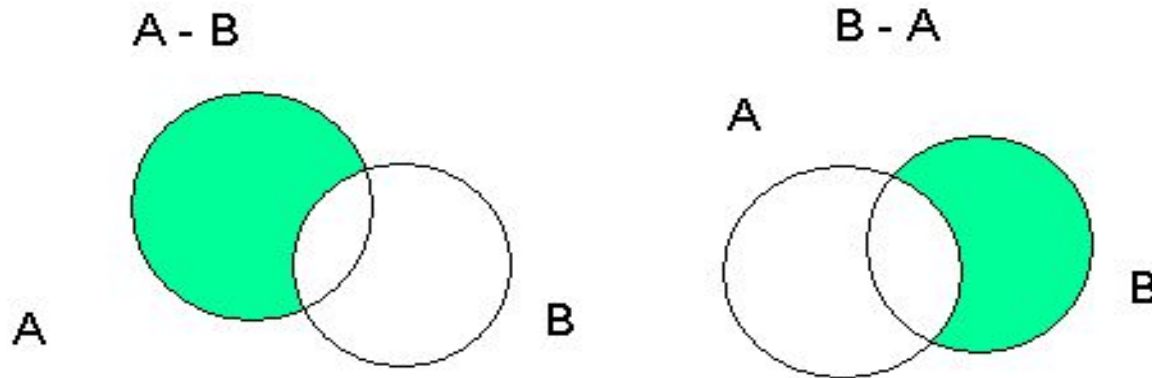
Примеры:

- 1)  $[1, 2, 3, 4] * [3, 4, 5, 6] \Rightarrow [3, 4]$
- 2)  $['a'..'z'] * ['A'..'E', 'k'] \Rightarrow ['k']$
- 3)  $[5 < 4, \text{true and false}] * [\text{true}] \Rightarrow []$



**Разностью** двух множеств A и B называется множество, состоящее из элементов множества A, не входящих во

В



Примеры:

1a)  $[1, 2, 3, 4] - [3, 4, 5, 6] \Rightarrow [1, 2]$

1b)  $[3, 4, 5, 6] - [1, 2, 3, 4] \Rightarrow [5, 6]$

2a)  $['a'..'z'] - ['A'..'E', 'k'] \Rightarrow ['a'..'j', 'i'..'z']$

2b)  $['A'..'E', 'k'] - ['a'..'z'] \Rightarrow ['A'..'E']$

3a)  $[5 < 4, \text{true and false}] - [\text{true}] \Rightarrow [\text{false}]$

3b)  $[\text{true}] - [5 < 4, \text{true and false}] \Rightarrow [\text{true}]$

**Добавить** новый элемент в множество можно с использованием операции объединения. Например,  $a := a + [5]$  Для этих же целей в Turbo Pascal 7.0 предназначена процедура **Include: include (M, A)** M – множество, A – переменная того же типа, что и элементы множества M. Тот же пример можно записать так: Include (a, 5)

**Исключить** элемент из множества можно с помощью операции «разность множеств». Например,  $a := a - [5]$  Для этих же целей в Turbo Pascal 7.0 предназначена процедура **Exclude: exclude (M, A)** M – множество, A – переменная того же типа, что и элементы множества M. Тот же пример можно записать так: Exclude (a, 5)



**Задача.** Дана строка. Сохранить в ней только первые вхождения символов, удалив все остальные.

```
program ex_set_;  
var m : set of char;  
    s : string; i : byte;  
begin  
    writeln('Введите строку: ');  
    readln(s);  
    m := [];  
    i := 1;  
    while i <= length(s) do  
        if s[i] in m then delete(s, i, 1)  
            else  
begin  
m:=m+[s[i]]; i := i + 1;  
end;  
        writeln(s);  
    Readln;  
end.
```

# Записи

**Запись** представляет собой совокупность ограниченного числа логически связанных компонент, принадлежащих к ***разным типам***. Компоненты записи называются полями, каждое из которых определяется именем. Поле записи содержит имя поля, вслед за которым через двоеточие указывается тип этого поля. Поля записи могут относиться к любому типу, допустимому в языке Паскаль.

Описание записи в языке Паскаль осуществляется с помощью служебного слова **record**, вслед за которым описываются компоненты записи. Завершается описание записи служебным словом **end**.

Например, телефонный справочник содержит фамилии и номера телефонов, поэтому отдельную строку в таком справочнике удобно представить в виде следующей записи:

```
type TRec = Record  
    FIO: String[20];  
    TEL: String[7];  
end;  
var rec: TRec;
```

Описание записей возможно и без использования имени типа, например:

```
var rec: Record  
    FIO: String[20];  
    TEL: String[7]  
end;
```

Обращение к записи в целом допускается только в операторах присваивания, где слева и справа от знака присваивания используются имена записей одинакового типа. Во всех остальных случаях оперируют отдельными полями записей. Чтобы обратиться к отдельной компоненте записи, необходимо задать имя записи и через точку указать имя нужного поля, например:  
rec.FIO, rec.TEL

Обращение к компонентам записей можно упростить, если воспользоваться оператором присоединения **with**.

Он позволяет заменить составные имена, характеризующие каждое поле, просто на имена полей, а имя записи определить в операторе присоединения:

**with rec do оператор;**

Здесь rec - имя записи, оператор - оператор, простой или составной. Оператор представляет собой область действия оператора присоединения, в пределах которой можно не использовать составные имена. Например для нашего случая:

```
with rec do begin  
    FIO:='Иванов А.А.';  
    TEL:='2223322';  
end;
```

Такая алгоритмическая конструкция полностью идентична следующей:

```
rec.FIO:='Иванов А.А.';  
rec.TEL:='2223322';
```