**Множественный тип данных**

**Множество** — это структурированный тип данных, представляющий собой набор взаимосвязанных по какому-либо признаку или группе признаков объектов, которые можно рассматривать как единое целое. Каждый объект в множестве называется элементом множества.

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], ['а',‘b','с'], ['a'..'z'].

Если множество не имеет элементов, оно называется пустым и обозначается как []. Количество элементов множества называется его мощностью.

Множество может принимать все значения базового типа. Базовый тип не должен превышать 256 возможных значений. Поэтому базовым типом множества могут быть byte, char, boolean и производные от них типы.

Множество в памяти хранится как массив битов, в котором каждый бит указывает является ли элемент принадлежащим объявленному множеству или нет. Максимальное число элементов множества 256, а данные типа множество могут занимать не более 32 байт.

Число байтов, выделяемых для данных типа множество, вычисляется по формуле:

**ByteSize = (max div 8) - (min div 8) + 1,**

где max и min — верхняя и нижняя границы базового типа данного множества.

Номер байта для конкретного элемента Е вычисляется по формуле:

**ByteNumber = (E div 8) - (min div 8),**

номер бита внутри этого байта по формуле:

**BitNumber = E mod 8**

Не имеет значения порядок записи элементов множества внутри конструктора. Например, [1, 2, 3] и [3, 2, 1] — это эквивалентные множества.

Каждый элемент в множестве учитывается только один раз. Поэтому множество [1, 2, 3, 4, 2, 3, 4, 5] эквивалентно [1..5].

Переменные множественного типа описываются так:  
**Var <идентификатор> : set of <базовый тип>;**

Например:

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];

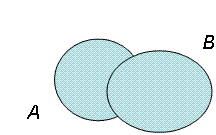
B : = ['m', 'n', 'k']; C : = [True, False];

D : = A;

Кроме того, выражения могут включать в себя операции над множествами.

**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:

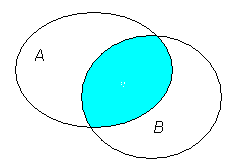
1) [1, 2, 3, 4] + [3, 4, 5, 6] => [1, 2, 3, 4, 5, 6]

2) []+[‘a’..’z’]+[‘A’..’E’, ‘k’]  => [‘A’..’E’, ‘a’..’z’]

3) [5<4, true and false] + [true]  => [false, 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 «\*»



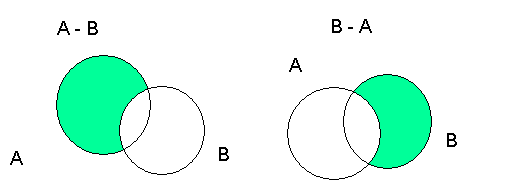
Примеры:

1) [1, 2, 3, 4] \* [3, 4, 5, 6] => [3, 4]

2) [‘a’..’z’]\*[‘A’..’E’, ‘k’]  => [‘k’]

3) [5<4, true and false] \* [true]  => []

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



Примеры:

1a) [1, 2, 3, 4] - [3, 4, 5, 6] => [1, 2]

1b)  [3, 4, 5, 6] - [1, 2, 3, 4]  => [5, 6]

2a) [‘a’..’z’]-[‘A’..’E’, ‘k’]  => [‘a’..’j’, ‘i’..’z’]

2b) [‘A’..’E’, ‘k’] - [‘a’..’z’]  => [‘A’..’E’]

3a) [5<4, true and false] - [true] => [false]

3b) [true] - [5<4, true and false] => [true]

**Операция вхождения**. Это операция, устанавливающая связь между множеством и скалярной величиной, тип которой совпадает с базовым типом множества. Если x — такая скалярная величина, а M — множество, то операция вхождения записывается так: **x in M.**

Результат — логическая величина true, если значение x входит в множество M, и false — в противном случае.

Например, 4 in [3, 4, 7, 9] –– true, 5 in [3, 4, 7, 9] –– false.

Используя данную операцию, можно не только работать с элементами множества, но и, даже если в решении задачи явно не используются множества, некоторые логические выражения можно записать более лаконично.

1) Натуральное число n является двухзначным. Вместо выражения **(n >= 10) and (n <=99)** можно записать **n in [10..99]**.

2) Символ c является русской буквой. Вместо выражения **(c >= ‘А’) and (c <= ‘Я’) or (c>=‘а’) and (c<=‘п’) or (c>=‘р’) and (c<=‘я’)** пишем **c in [‘А’.. ‘Я’, ‘а’.. ‘п’, ‘р’.. ‘я’]** и т.д.

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)

Рассмотрим несколько примеров использования множеств при решении задач.

**Задача 1.** В городе имеется n высших учебных заведений, которые производят закупку компьютерной техники. Есть шесть компьютерных фирм: «Диалог», «Avicom», «Нэта», «Сервер», «Декада», «Dega.ru». Ответить на следующие вопросы:  
1) в каких фирмах закупка производилась каждым из вузов?  
2) в каких фирмах закупка производилась хотя бы одним из вузов?  
3) в каких фирмах ни один из вузов не закупал компьютеры?

Решим задачу с использованием множеств. Для удобства дальнейших манипуляций в порядке следования занумеруем компьютерные фирмы, начиная с единицы. Занесём информации о месте закупок компьютеров каждым из вузов в отдельное множество.

Ответ на первый вопрос можно получить, выполнив пересечение всех таких множеств.

Ответ на второй вопрос – результат объединения множеств.

И, наконец, на последний – разность множества всех фирм и множества фирм, где хотя бы один вуз делал покупки.

program ex\_set\_1;

type firma = set of 1..6;

v = array[0..20] of firma;

const f: array [1..6] of string[10] = ('Диалог', 'Avicom', 'Нэта', 'Сервер', 'Декада', 'Dega.ru');

{процедура ввода информации о закупке компьютеров в очередной фирме}

procedure vvod(var a: firma);

var i: byte; ans: 0..1;

begin

a:= [];

for i := 1 to 6 do

begin

Write('Вуз покупал компьютеры в фирме ', f[i], ' (1 - да, 0 - нет)? '); ReadLn(ans);

if ans = 1 then a:=a+[i]

end;

end;

{процедура вывода элементов массива, номера которых содержатся в множестве}

procedure Print(a : firma);

var i: byte;

begin

for i := 1 to 6 do if i in a then write(f[i]:10);

writeln

end;

{Процедура, дающая ответ на первый вопрос}

procedure Rez1(a: v; n : byte; var b : firma);

var i : byte;

begin

b := [1..6];

for i := 0 to n-1 do b := b \* a[i];

end;

{Процедура, дающая ответ на второй вопрос}

procedure Rez2(a: v; n : byte; var b : firma);

var i : byte;

begin

b := [];

for i := 0 to n-1 do b := b + a[i];

end;

var a: v; n, i : byte; c : firma;

begin

write('Сколько вузов делали закупку? '); readln(n);

for i := 0 to n-1 do vvod(a[i]);

Rez1(a, n, c);

writeln('Каждый из вузов закупил компьютеры в фирмах: '); Print(c);

Rez2(a, n, c);

writeln('Хотя бы один из вузов закупил компьютеры в фирмах: '); Print(c);

writeln('Ни один из вузов не закупил компьютеры в фирмах: '); Print([1..6]-c);

end.

**Задача 2.** Сгенерировать *n* множеств (нумерацию начать с 1). Вывести элементы, которые входят во все множества с номерами, кратными трём, но не входят в первое множество.

program ex\_set\_2;

type mn = set of byte;

v = array[1..30] of mn;

{процедура ввода информации в очередное множество}

procedure vvod(var a: mn);

var i, n, vsp: byte;

begin

a:= []; n := 1 +random(200);

for i := 1 to n do

begin

vsp:= random(256);

a:=a+[vsp]

end;

end;

{процедура вывода элементов множества}

procedure Print(a : mn);

var i: byte;

begin

for i := 0 to 255 do if i in a then write(i:4);

writeln

end;

{Процедура, дающая ответ на вопрос}

procedure Rez(a: v; n : byte; var b : mn);

var i : byte;

begin

b := [0..255];

i:= 3;

while i <= n do

begin

b := b \* a[i];

i := i + 3

end;

b := b - a[1]

end;

var a: v; n, i : byte; c : mn;

begin

randomize;

write('Сколько множеств? '); readln(n);

for i := 1 to n do

begin vvod(a[i]);

print (a[i])

end;

Rez(a, n, c);

Print(c);

end.

**Задача 3.** Дана строка. Сохранить в ней только первые вхождения символов, удалив все остальные.

program ex\_set\_3;

var m : set of char;

s : string; i : byte;

begin

write('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)

end.

**Контрольные вопросы и задания**

1. Что такое множество?
2. Почему множество является структурированным типом данных?
3. Как хранится множество в памяти ЭВМ? Какой максимальный объем оперативной памяти может быть отведен под хранение одного множества?
4. Какие операции можно выполнять над множествами?
5. Как добавить элемент в множество?
6. Как исключить элемент из множества?
7. Как вывести элементы множества? Как подсчистать количество элементов в множестве?