Именованные и типизированные константы

**Именованная константа**

Именованная константа — это имя (идентификатор), которое в программе используется вместо самой константы.

Именованная константа, как и переменная, перед использованием должна быть объявлена. В общем виде инструкция объявления именованной константы выглядит следующим образом:

константа = значение;

где:

константа — имя константы;

значение — значение константы.

Именованные константы объявляются в программе в разделе объявления констант, который начинается словом const. Ниже приведен пример объявления именованных констант (целой, строковой и дробной).

const

Bound = 10;

Title = 'Скорость бега';

pi = 3.1415926;

После объявления именованной константы в программе вместо самой константы можно использовать ее имя.

В отличие от переменной, при объявлении константы тип явно не указывают. Тип константы определяется ее видом, например:

125 — константа целого типа;

0.0 — константа вещественного типа;

' выполнить ' — строковая константа;

' \' — символьная константа.

**Типизированная константа**

***Типизированные константы фактически являются переменными с статическим классом памяти***. То есть такими, которые получают описанное для них значение только один раз в начале выполнения программы а при каждом новом входе в процедуру (функцию), где они объявлены заново не инициализируются и сохраняют свое значение, полученное к время предыдущего вызова процедуры (функции). Типизированные константы можно использовать точно так же, как и переменные того же самого типа, и они могут появляться в левой части оператора присваивания.

    Кроме обычных константных выражений для задания значения типизированной константы используют также константные адресные выражения. ***Константное адресное выражение*** - это выражение, значением которого является адрес глобальной переменной, типизированной константы, процедуры или функции. Константное адресное выражение не может ссылаться к локальным переменным процедур или к динамическим переменным, поскольку их адреса нельзя вычислить во время компиляции.

    Так как типизированная константа фактически представляет собой инициализируемую переменную, то она не может использоваться в объявлениях других констант или типов.

    Описание типизированной константы имеет следующий вид:

**const**

<ИдентификаторКонстанты> : <Тип> = <ЗначениеКонстанты>;

    Типизированные константы стандартных типов:

**const**

ArrLength : Integer = 100;

Step : Real = 0.001;

Flag : Boolean = False;

LineFeed : Char = #10;

**var**

Buffer : array[0..1023] **of** Byte;

**const**

BufferOfs : Word = Ofs(Buffer);

BufferSeg : Word = Seg(Buffer);

Ptr : Pointer = @Buffer;

    Типизированные константы строкового типа:

**const**

Newline : String [2] = #13#10;

Name : String [14] = 'Object Pascal' ;

    Типизированные константы типа "указатель":

**type**

Ptr = ^Integer;

**const**

IntPtr : Ptr = nil;

Int1 : Integer = 0;

IntlPtr : Ptr = @Int1;

**Object Pascal** поддерживает работу с типизированными константами следующих структурированных типов:

* типа ***"массив"*** (**array**);
* типа ***"множество"*** (**set**);
* типа ***"запись"*** (**record**).

    В описании константы структурированного типа указываются значения каждого ее компонента соответственно определенным синтаксическим правилам.

**Типизированные константы типа "массив"**

    При описании константы типа "массив" компоненты каждой размерности массива заключаются в отдельные скобки и разделяются запятыми. Компоненты, расположенные в самых внутренних скобках соответствуют последней (самой правой) размерности массива.

    Примеры констант типа "массив":

* одномерный числовой массив
* **const**
* DigVector : **array** [1..7] **of** Real = (0.1, 3.25, 21.32, 55, 11.99, 78.1, 4.5);
* двумерный числовой массив
* **const**
* DigMatrix : **array** [1. .3,1..2] **of** Integer = ( (1,2), (3,4), (5,6) );

    В результате элементы матрицы DigMatrix получат такие значение

DigMatrix[l,l] = 1;

DigMatrix[1,2] = 2;

DigMatrix[2,1] = 3;

DigMatrix[2,2] = 4;

DigMatrix[3,1] = 5;

DigMatrix[3,2] = 6;

* трехмерный числовой массив
* **const**
* Dig3D : **array** [1..4,1..3,1..2] **of** Byte = ( ((1,2),(1,2),(1,2)),
* ((1,2), (1,2), (1,2)),
* ((1,2), (1,2), (1,2)),
* ((1,2), (1,2), (1,2)) );
* одномерный массив символов
* **const**
* CharVect1 : **array** [1..6] **of** Char = ('P' ,'A' , 'S' , 'C, 'A', 'L') ;
* или более кратко
* CharVect2 : **array** [1..6] **of** Char = 'PASCAL';

**Типизированные константы типа "множество"**

    Синтаксис описания констант типа "множество" определяется следующими правилами.

    Каждый элемент константы типа "множество" может представлять собой либо отдельную константу соответствующего типа, либо интервал значений, состоящий из двух констант, разделенных символом . . (две точки).

    Примеры констант типа "множество":

**type**

Digits = **set** **of** 0..9;

CharDig = **set** **of** '0'..'9';

**const**

DigSetl : Digits = [0, 2, 4, 6, 8];

DigSet2 : Digits = [1..3, 5..7];

CharDigSet1 : CharDig = [ '0' , '2' , '4' , '6' , '8' ] ;

CharDigSet2 : CharDig = ['0'..'3', '5'..'7'];

CharSet : **set** **of** Char = ['a'..'z','A'..'Z'];

**Типизированные константы типа "запись"**

    В описании константы типа "запись" указываются как значения всех полей записи, так и их идентификаторы. В типизированных константах типа "запись" не допускаются поля файлового типа. В вариантных константах-записях допускается указание только того варианта полей, который соответствует установленной предварительно константе поля-признака. Поля указываются в том же порядке, в котором они следуют в описании типа.

    Примеры типизированных констант типа "запись".

**type**

Reс = **record**

R : Real;

В : Boolean;

C : Char;

end;

ArrayOfRec = **array** [1..3] **of** Rec;

RecOfArray = **record**

ArrInt : **array** [1..3] **of** Integer;

ArrChar : **array** [1..2] **of** Char;

end;

RecOfRec = **record**

I : Integer;

S : String;

Z : Rec;

end;

**const**

RecElem : Rec = ( R: 3.1415; B: True; С : '\*' ) ;

ArrRec : ArrayOfRec =

( ( R: 3.1415; B: True; C : '\*' ),

( R: 0.0; B: False; C : '$' ),

( R: 6.2832; B: True; C : '&' ) );

RecArr : RecOfArray =

( Arrlnt: (1,2 ,3); ArrChar: (' 1' , ' 2') ) ;

RecRec : RecOfRec =

( I: 32767; S: 'PASCAL';

Z: ( R: 3.1415; В: True; С: '\*' ) ) ;

**Типизированные константы процедурного типа**

    Константа процедурного типа должна быть идентификатором процедуры или функции, совместимым по присваиванию с типом константы.

**type**

FuncType = Function (X : Real);

**function** SH(X: Real): Real; far;

**begin**

SH : = (Exp(X)-Exp(-X)) / 2;

**end;**

**function** CH(X: Real): Real; far;

**begin**

CH : = (Exp(X)+Exp(-X)) / 2;

**end;**

**const**

CurrentFunc: FuncType = SH;

Литература

<http://www.realcoding.net/articles/imenovannaya-konstanta.html>

<http://it.kgsu.ru/Ob_Pas/obpas014.html>