Основные термины языка ПАСКАЛЬ

Ввод означает считывание значений, поступающих с клавиатуры, с диска или из порта ввода-вывода.

Данные — это константы, переменные и структуры, содержащие числа (целые и вещественные), текст (символы и строки) или адреса (переменных и структур).

Операции осуществляют присваивание значений, их комбинирование (сложение, деление и т. д.) и сравнение значений (равные, неравные и т. д.).

Вывод означает запись информации на экран, на диск или в порт ввода-вывода.

Условное выполнение предполагает выполнение набора команд в случае, если удовлетворяется (является истинным) некоторое условие (если это условие не удовлетворяется, то эти команды пропускаются или же выполняется другой набор команд) или если некоторый элемент данных имеет некоторое специальное значение или значение из некоторого спектра.

Благодаря ***циклам*** некоторый набор команд выполняется повторно или фиксированное число раз, или пока является истинным некоторое условие, или пока некоторое условие не стало истинным.

Подпрограмма представляет собой набор команд, который имеет имя и может быть неоднократно вызван из любого места программы по его имени.

Компилятор (англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется. ***Компиляция*** означает перевод *исходной программы* с языка Pascal в *объектную программу* – на язык компьютера. Если при написании программы, программист допустил синтаксическую ошибку (опечатался, неправильно написал название команды и т.д.), то при запуске программы компилятор выдаст ошибку, т.к. не сможет перевести данный фрагмент программы на машинный язык.

Теперь рассмотрим, как эти элементы используются в Паскале.

Алфавит и словарь языка паскаль

Языком называется совокупность символов, соглашений и правил, используемых для общения. При записи алгоритма решения задачи на языке программирования необходимо четко знать правила написания и использования элементарных информационных и языковых единиц. Основой Паскаля, как и любого языка, является ***алфавит*** — конечный набор знаков, состоящий из букв, десятичных и шестнадцатеричных цифр, специальных символов.

В качестве букв в Паскале используются прописные и строчные буквы латинского алфавита:
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz и знак подчеркивания (_);
в качестве десятичных цифр: 0123456789.
Шестнадцатеричные цифры строятся из десятичных цифр и букв от A до F (или от a до f).
При написании программ применяются следующие специальные символы:

+	плюс	,	запятая
-	минус	.	точка
*	звездочка	:	двоеточие
/	дробная черта	[ ]	квадратные скобки
>	больше	{}	фигурные скобки
<	меньше	$	знак денежной единицы
=	равно	()	круглые скобки
;	точка с запятой	~	тильда
#	номер	@	коммерческое а
'	апостроф	нет обозначения	пробел

Комбинации специальных символов могут образовывать составные символы:

:=	присваивание	<=	меньше или равно
<>	неравно	>=	больше или равно
..	диапазон значений	(..)	альтернатива [ ]
(* *)	альтернатива {}		

Фигурные скобки { } в тексте программы можно использовать для написания ***комментариев*** к программе. Текст, записанный в таких скобках, компилятор не обрабатывает. Поэтому комментарии не влияют на решение задачи.

В программе эти пары символов нельзя разделять пробелами, если они используются как знаки операций отношения или ограничители комментария.

Примечание. Русские буквы в программе должны заключаться в апострофы, например 'Пример текста на русском языке'.

Примеры:

W— прописная латинская буква;

w— строчная латинская буква;

9 — цифра; $ — специальный символ; <> — составной символ.

Слова в Паскале

Неделимые последовательности знаков алфавита образуют ***слова,*** отделённые друг от друга разделителями и несущие определенный смысл в программе. Разделителем могут служить пробел, символ конца строки, комментарий. Набор слов, используемый в Паскале, можно разделить на три группы: зарезервированные слова, стандартные идентификаторы и идентификаторы пользователя.

Зарезервированные слова являются составной частью языка, имеют фиксированное начертание и раз и навсегда определенный смысл. Они не могут изменяться программистом. Зарезервированные слова версии языка Паскаль для персональных ЭВМ приведены ниже.

Зарезервированные слова языка Паскаль

absolute	абсолютный	label	метка
and	логическое И	library	библиотека
array	массив	mod	остаток от деления
asm	ассемблер	nil	отсутствие
begin	начало блока	not	логическое НЕ
case	вариант	or	логическое ИЛИ
const	константа	of	из
constructor	конструктор	object	объект
div	деление нацело	packed	упакованный
goto	переход на	procedure	процедура
do	выполнять	program	программа
downto	уменьшить до	record	запись
destructor	деструктор (разрушитель)	repeat	повторять
else	иначе	set	множество
end	конец блока	shl	сдвиг битов влево
exports	экспорт	shr	сдвиг битов вправо
external	внешний	string	строка
file	файл	then	то
for	для	to	увеличивая
forward	опережающий	type	тип
function	функция	unit	модуль
if	если	until	до
implementation	реализация	uses	использовать
in	в (входит в...)	var	переменная
inline	основной	while	пока
interrupt	прерывание	with	с
interface	интерфейс	xor	исключающее ИЛИ
inherited	наследование		

Зарезервированные слова нельзя использовать в качестве имен, вводимых программистом для обозначения величин и т.д.

Примечание: компилятор языка Паскаль не делает различий между прописными и строчными буквами, поэтому текст программы можно писать в любом регистре и использовать это свойство для удобства чтения текста программы. Например, слова And, AND, and, aNd будут обработаны компилятором совершенно одинаково и их написание не повлияет на решение задачи.

Для того чтобы научиться правильно писать программы для компьютера, необходимо изучить ***синтаксис языка*** программирования (правила записи его конструкций) и его ***семантику*** (смысл и правила использования этих конструкций).

Структура программы

Программа реализует алгоритм решения задачи. В ней программист записывает последовательность действий, выполняемых над определенными данными с помощью определенных операций для реализации заданной цели. Основные характеристики программы: точность полученного результата, время выполнения и объем требуемой памяти. О соответствии этих показателей решаемой задаче и возможностям компьютера должен позаботиться сам программист. В большинстве случаев определяющим требованием является точность. Ограничения по объему памяти и времени выполнения носят менее жесткий характер.

Программа на языке Паскаль состоит из строк. Набор текста программы осуществляется с помощью встроенного редактора текстов системы программирования Турбо.

Программист, набирая текст программы, имеет право произвольно располагать строки на экране. Строка может начинаться с любой колонки, т. е. величина отступа от левой границы экрана для каждой строки устанавливается самим программистом с целью получить наиболее удобный для чтения, по его мнению, текст программы. Количество операторов в строке произвольно, но если в строке записывается один оператор, такая программа легче читается.

Существуют различные схемы написания программ на языке Паскаль, все они отличаются количеством отступов слева в каждой строке и различным использованием прописных букв.

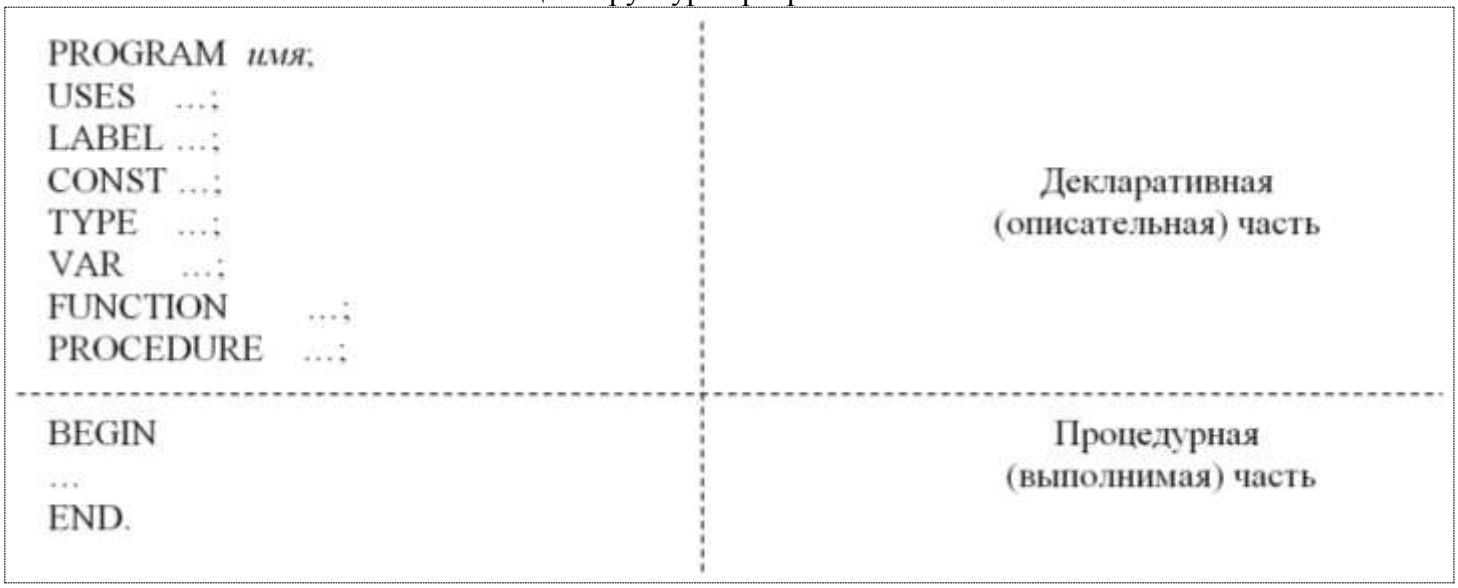
Структура программы на ЯП Паскаль состоит из двух основных частей:

- *Описательная (декларативная) часть*
- *Выполнимая (процедурная) часть*

Основной текст программы содержит список операторов, последовательное выполнение которых и приводит к решению поставленной задачи. Данный список операторов необходимо записывать в **выполнимую часть** программы. Начинается выполнимая часть со слова ***Begin*** и заканчивается словом ***End***. В конце программы после ключевого слова ***End*** ставится ‘.’ (точка), по которой компилятор и определяет конец программы.

В процессе решения поставленной задачи нередко требуются дополнительные объекты для ее решения (переменные, константы, функции, процедуры и пр.). Для описания всех необходимых объектов, которые будут использованы при решении данной задачи, предназначена **описательная часть** программы. Данная часть состоит из разделов, в каждом из которых описываются разные виды объектов. Разделы этой части являются *необязательными* и добавляются при необходимости. Например, если в программе будут использоваться константы, то надо добавить раздел **Const**. А если константы в программе не нужны, то и раздел **Const** добавлять в описательную часть незачем. В конце каждого раздела ставится ‘;’ (точка с запятой).

Общая структура программы ЯП Pascal



Рассмотрим подробнее назначение каждого из разделов описательной части программы.

Program – раздел для указания имени программы

В начале программы находится заголовок, состоящий в общем случае из зарезервированного слова *Program* и имени программы. Заголовок программы несет чисто смысловую нагрузку и может отсутствовать, однако рекомендуется всегда его записывать для быстрого распознавания нужной программы среди листингов других программ.

Пример:

```
Program Primer1;
```

Uses – раздел для подключения библиотечных модулей

Этот раздел состоит из зарезервированного слова *Uses* и списка имен подключаемых стандартных и пользовательских библиотечных модулей.

Формат:

```
Uses <имя1>;
```

Подробно о структуре и организации библиотечных модулей будет рассказано ниже.

Данный раздел необходим, если мы собираемся использовать в своей программе дополнительные процедуры и функции, которые находятся в специальных библиотечных модулях (точнее, там находится описание их работы). Для того, чтобы компилятор понял как работает такая процедура или функция, необходимо как раз в разделе uses подключить соответствующую библиотеку.

Например:

```
Uses CRT;          {подключение библиотечного модуля CRT}
…
Begin
   ClrScr;         {процедура очистки экрана, от англ.Clear Screen}
   …
End.
```

Процедура ClrScr будет работать в программе, только если подключить модуль CRT в разделе Uses. Иначе компилятор не поймет «какую работу» должна выполнить эта процедура.

Label – раздел для описания меток

Перед любым оператором языка Паскаль можно поставить метку, что позволяет выполнить прямой переход на этот оператор с помощью оператора перехода ***go to*** из любого места программы.

Перед употреблением метка должна быть описана. Раздел описания меток начинается зарезервированным словом label (метка), за которым следуют имена меток, разделенные запятыми. За последним именем ставится точка с запятой.

Формат:

```
Label  <имя>;
```

Const – раздел для описания констант

В разделе описания констант производится присваивание идентификаторам констант постоянных значений. Раздел начинается зарезервированным словом const, за которым следует ряд выражений, присваивающих идентификаторам постоянные числовые или строковые значения. Выражения присваивания отделяются друг от друга точкой с запятой.

Формат:

```
Const <идентификатор> = <значение>;
```

Type – раздел для описания пользовательских типов данных

Тип данных может быть либо описан непосредственно в разделе описания переменных, либо определяться идентификатором типа. Стандартные типы не требуют описания в отличие от типов, образованных пользователем. Выбор описания типа зависит, таким образом, только, от программиста и специфики программы.

Раздел описания типов данных начинается зарезервированным словом **Type**, за которым следуют одно или несколько определений типов, разделенных точкой с запятой.

Формат:

```
Type  <имя типа> = <значения типа>;
```

Пример.

```
Type  LatLetter = ('A'..'z') ;
      Days = 1..31;
      Matr = array[1..10]   of integer;
```

Каждое описание задает множество значений и связывает с этим множеством некоторое имя типа. Например, в данном описании тип LatLetter определяет множество букв латинского алфавита, Days — множество целых чисел от 1 до 31, Matr — массив из 10 целых чисел.

Var – раздел для описания переменных

Каждая встречающаяся в программе переменная должна быть описана. Описание обязательно предшествует использованию переменной. Раздел описания переменных начинается зарезервированным словом var (variable — переменная), затем через запятую перечисляются имена переменных и через двоеточие следуют их тип и точка с запятой.

Формат:

```
Var   <идентификатор> : <тип>;
```

Пример:

```
Var   A, B, Proizved  :  integer;
      Z : real;
      S : string;
```

В данном примере программы три переменных A, B и Proizved могут принимать целочисленные значения, переменная Z – переменная вещественного типа, а переменная S – переменная строкового типа.

Раздел *Var* является основным при написании программ, так как именно в нем программист указывает сколько переменных и какого типа потребуется при решении задачи.

Function, Procedure - разделы для описания процедур и функций пользователя

В этом разделе размещаются тела подпрограмм. Подпрограммой называется программная единица, имеющая имя, по которому она может быть вызвана из других частей программы. В языке Паскаль роль подпрограмм выполняют процедуры и функции. В общем случае подпрограмма имеет ту же структуру, что и программа. Для описания подпрограмм используются зарезервированные слова procedure и function, которые записываются в начале подпрограммы. Формат процедуры:

Процедуры и функции подразделяются на стандартные и определенные пользователем. Стандартные процедуры и функции являются частью языка и могут вызываться без предварительного описания. Описание процедур и функций пользователя обязательно.

Рассмотрим структуру программы на примере программы решения задачи вычисления произведения двух целых чисел.

```
Program   Tutor3;                          {Заголовок программы}
Var                                        {Описание раздела переменных}
      A,B,Proizved : Integer;              {Переменные A,B,Proizved – целые}
Begin                                      {Начало программы}
  Writeln('Введите число A:');             {Вывод запроса на экран}
  Readln(A);                               {Ввод значения A с клавиатуры}
  Writeln('Введите число B:');
  Readln(B);
  Proizved:= A * B;                        {Вычисление переменной Proizved}
  Writeln('Произведение = ', Proizved);    {Вывод ответа}
end.                                       {Конец программы}
```

В данной программе в описательной части только два раздела *Program* и *Var*, а в выполнимой части программы (между *Begin* и *End*) записаны 6 операторов. Программа ожидает ввода двух целых чисел с клавиатуры и выводит на экран их произведение.

Идентификаторы

Для того чтобы программа решения задачи обладала свойством массовости, следует не употреблять конкретные значения величин, а использовать их обозначения для возможности изменения по ходу выполнения программы их значений. Для обозначения программ, а в программе переменных и постоянных величин, различных процедур, функций, объектов используются имена — ***идентификаторы*** (identification — установление соответствия объекта некоторому набору символов).

Для обозначения заранее определенных разработчиками языка типов данных, констант, процедур и функций служат ***стандартные идентификаторы,*** например: integer, Sin, Cos, Ln, Sqr, Sqrt, Read, Readln, Write, Writeln. В этом примере стандартный идентификатор Sin вызывает функцию, вычисляющую синус заданного угла, Read, Readln вызывают процедуру, организующую ввод данных, Write, Writeln вызывают процедуру, организующую вывод данных.

Для обозначения меток, констант, переменных, процедур и функций, определенных самим программистом, применяются ***идентификаторы пользователя.*** При этом идентификаторы в программе должны быть уникальными, т. е. в данном блоке программы не может использоваться один идентификатор для обозначения более чем одной переменной или постоянной величины и т.д.

Компилятор Турбо Паскаля строго следит за этим, и если это требование не соблюдается, компиляция прерывается, а на экран выводится сообщение об ошибке "Error 4: Duplicate identifier" и указывается дубликат идентификатора.

В идентификатор не могут входить пробелы, специальные символы алфавита. Обратите внимание, что буквы русского алфавита не могут входить в идентификатор Турбо Паскаля.

При записи программ следует соблюдать ***общие правила написания идентификаторов:***

1. Идентификатор начинается только с буквы или знака подчеркивания (исключение составляют метки, которые могут начинаться и цифрой, и буквой).
2. Идентификатор может состоять из букв, цифр и знака подчеркивания (пробелы, точки и другие специальные символы при написании идентификаторов недопустимы).
3. Между двумя идентификаторами должен быть по крайней мере один пробел.
4. Максимальная длина идентификатора 127 символов, но значимы только первые 63 символа.
5. При написании идентификаторов ***можно использовать как прописные, так и строчные буквы***.

Компилятор не делает различий между ними, хотя они и имеют различные номера в стандартном коде обмена информацией. На практике рекомендуется применять эту особенность для более простого чтения и понимания значений идентификаторов. Так, вместо идентификатора nomerotdela лучше написать NomerOtdela, выделив прописными буквами каждую из двух смысловых частей.

Правильно выбранные идентификаторы значительно облегчают чтение и понимание программы, а также уменьшают вероятность появления ошибок при модификации программ. Например, значение даты удобнее обозначить идентификатором Data, чем просто буквой D или любым другим символом.

Примеры:

`Metkal2`

`2graph` — ошибка, идентификатор начинается с цифры

`Block_56`

`Homer.Doma` — ошибка, идентификатор содержит точку

`Площадь` — ошибка, идентификатор содержит символы русского языка.

Константы и переменные

Решение задачи на компьютере — это процесс сбора, обработки и передачи информации. Поэтому любая программа имеет смысл, если она обрабатывает какие-либо данные. Как и в других языках программирования, в Паскале данные разделяются на константы и переменные. В программе константы и переменные определяются идентификаторами (именами), по которым к ним можно обращаться для получения текущих значений.

Константами называются элементы данных, значения которых установлены в описательной части программы и в процессе выполнения программы не изменяются. Константы задаются идентификаторами пользователя. Например, если вы используете в программе ваше имя, то его лучше всего задать константой, так как имя дается раз и навсегда, и не меняет своего значения.

Все константы должны быть описаны в специальном разделе, который начинается зарезервированным словом const (с англ. «constant» — константа).

Формат:

```
Const  <иденфификатор> = (значение константы>;
```

Например:

```
Const  MyName = 'Петя Иванов';
       Max = 1000;
       Num =-6.34;
```

Удачное именование констант пользователя делает программу более читаемой и позволяет быстро вносить корректировку в программу при изменении алгоритма.

В Турбо Паскале большое число констант определено стандартно, к ним можно обращаться без предварительного описания. Их называют зарезервированными константами. Наиболее употребительные из них приведены в таблице.

Зарезервированные константы

Идентификатор	Тип	Значение	Описание
True	Boolean	True	"Истина"
False	Boolean	False	"Ложь"
Maxint	Integer	32767	Максимальное целое
MaxLongint	Longint	2147483647	Максимальное длинное целое
pi	Real	3.14159265360	Число π

Переменными называют величины, которые могут менять свои значения в процессе выполнения программы. Каждые переменная принадлежит к определенному типу данных.

Тип переменных должен быть описан перед тем, как с переменными будут выполняться какие-либо действия. Этим мы как бы объявляем компьютеру, какие ячейки памяти мы собираемся использовать для хранения данных в своей программе. Тип данных указывает какие значения могут быть присвоены этой переменной в программе.

При описании переменной также указывается ее **имя** (идентификатор). При этом надо учитывать общие правила написания идентификаторов.

Значение переменной присваивается в выполнимой части программы с помощью оператора ‘:=’ (присваивания).

Само название "переменная" подразумевает, что содержимое объявленной области памяти будет изменяться в ходе выполнения программы. Переменные описываются в специальном разделе, который начинается зарезервированным словом **Var** (с англ. «variable» — переменная).

Формат:

```
Var   <идентификатор>   :   <тип>;
```

Пример:

```
Var   A, B : integer;            { A,B – переменные целого типа}
      Srednee : real;            { Summa – переменная вещественного типа}
Begin
   A:=2;
   B:=6;
   Srednee:=(A+B)/2;
   Writeln(Srednee);
End.
```

В данном примере для переменных A, B: их идентификаторами (именами) будут A,B, их типом данных будет тип Integer, а их значениями будут 2 и 6 соответственно.

Переменная всегда имеет:

1) **Имя** (идентификатор) - задает пользователь в разделе Var
2) **Тип данных** – указывает пользователь в разделе Var
3) **Значение** – присваивается в программе (в выполнимой части)

Подводя итог можно сформулировать следующие утверждения:

Имя переменной подобно ящичку, который можно заполнить различными значениями, что нельзя сделать с константой.

Имена констант и переменных должны соответствовать их назначению, в этом случае ваша программа будет абсолютно прозрачна для понимания.

Стандартные типы ЯП Pascal

Название типа	*Тип объекта*	*Множество значений*
Integer	целочисленный	$x \in [-32768; 32767]$
Byte	целочисленный	$x \in [0; 255]$
Real	действительный	$x=0, \ 10^{-38} \le \lvert x \rvert \le 10^{38}$
Char	символьный	символы ASCII кода
String	строковый	строка не более 255 символов
Boolean	логический	{False, True}

Логические операции и операции сравнения

операция	*название*	*пример*
<	меньше	*a<0*
>	больше	*b>3*
=	равно	*a=0*
<>	не равно	*b<>0*
>=	больше или равно	*a>=0*
<=	меньше или равно	*a<=0*
OR	дизъюнкция – логическое «или»	(*a>0*) OR (*b=0*)
AND	конъюнкция – логическое «и»	(*a>0*) AND (*b>0*)
NOT	отрицание – логическое «не»	NOT(*b>0*)