1 курс

ПЛАН – КОНСПЕКТ
проведения занятия по дисциплине «Информатика»

Раздел 4. «Основы алгоритмизации и программирования.»

Тема 4.1: «Общие принципы построения базовых алгоритмических структур в среде программирования.»

часть 2

Подготовил: преподаватель
В.Н. Борисов

Рязань 2023

Вопросы занятия:

1. Алфавит языка Turbo Pascal.
2. Идентификаторы.
3. Служебные слова.
4. Типы данных.
5. Переменные и константы.
6. Структура программы.
7. Компиляция программы.
8. Целочисленный и вещественный типы данных.
9. Правила записи арифметических выражений.
10. Оператор присваивания.
11. Аналитический расчет результатов выполнения операции присваивания.
12. Основные выводы.

Время проведения занятия – 2 часа.

Введение.

В кратком изложении история языков программирования такова: изначально вычислительные машины программировались в машинном коде. То есть в их оперативную память напрямую вводили последовательность чисел, являющиеся кодами команд, которые процессор может выполнить. При этом программа составлялась с периодическим заглядыванием в таблицу кодов команд процессора и была отнюдь не наглядной. Затем появилась идея обозначить коды какими-то короткими, но осмысленными, и потому легко запоминаемыми словами - мнемониками, и создать программу, которая бы, руководствуясь таблицей команд, переводила последовательность мнемоник - мнемокод в последовательность машинных кодов. Такую программу называют ассемблером (assembler - сборочное устройство, транслятор, ассемблер). Программы стали гораздо нагляднее, но решение практических задач требовало написания очень длинных программ (например, файловый менеджер Volkov Commander имеет размер около 64000 байт). Тогда появились языки программирования высокого уровня. При их создании использовали то обстоятельство, что в программе часто встречаются участки одинакового кода, выполняющие какое либо одно действие: вывод строки, запись в файл, вычисление математической функции и т.д. В языках высокого уровня таким последовательностям кода присвоены имена, и программа составляется на условном языке, каждое, из слов которого заменяет десятки, ато и сотни команд процессора. Таким образом, программа становится еще нагляднее и короче. Существует множество условных языков высокого уровня, для каждого из них написано немало вариантов пограммы, переводящей условный код в последовательность машинных команд. Один из таких языков - Паскаль.

Язык программирования Pascal был разработан в 1968-1971 гг. Никлаусом Виртом в Цюрихском Институте информатики (Швейцария), и назван вчесть Блеза Паскаля - выдающегося математика, философа и физика 17-го века. Первоначальная цель разработки языка диктовалась необходимостью создания инструмента "для обучения программированию как систематической дисциплине". Однако очень скоро обнаружилась чрезвычайная эффективность языка Pascal в самых разнообразных приложениях, от решения небольших задач численного характера до разработки сложных программных систем - компиляторов, баз данных, операционных систем и т.п. К настоящему времени Pascal принадлежит к группе наиболее распространенных и популярных в мире языков программирования:

- существуют многочисленные реализации языка практически для всех машинных архитектур;
- разработаны десятки диалектов и проблемно-ориентированных расширений языка Pascal;
- обучение программированию и научно-технические публикации в значительной степени базируются на этом языке.

Характеристика и особенности языка

Существует ряд объективных причин, обусловивших выдающийся успех языка Pascal. Среди них в первую очередь необходимо указать следующие:

1. Язык в естественной и элегантной форме отразил важнейшие современные концепции технологии разработки программ:
   а) развитая система типов,
   б) ориентация на принципы структурного программирования,
   в) поддержка процесса пошаговой разработки.
2. Благодаря своей компактности, концептуальной целостности и ортогональности понятий, а также удачному первоначальному описанию, предложенному автором языка, Pascal оказался весьма легок для изучения и освоения. В противоположность громоздким многотомным описаниям таких языков, как PL/l, Cobol, FORTRAN, достаточно полное описание языка Pascal занимает около 30 страниц текста, а его синтаксические правила можно разместить на одной странице.
3. Несмотря на относительную простоту языка, он оказался пригоден для весьма широкого спектра приложений, в том числе для разработки очень больших и сложных программ, например, операционных систем.
4. Pascal весьма технологичен для реализации практически для всех, в том числе и нетрадиционных, машинных архитектур. Утверждается, что разработка Pascal-транслятора "почти не превышает по трудоемкости хорошую дипломную работу выпускника вуза". Благодаря этому для многих ЭВМ существует несколько различных реализаций языка, отражающих те или иные практические потребности программистов.

Язык Pascal стандартизован во многих странах. В 1983 году был принят международный стандарт (ISO 7185:1983)

Основные особенности языка Pascal.

1. Pascal является традиционным алгоритмическим языком программирования, продолжающим линию Algol-60. Это означает, что программа на языке Pascal представляет собой специально организованную последовательность шагов по преобразованию данных, приводящую к решению некоторой задачи. Это отличает Pascal от так называемых непроцедурных языков типа Prolog, по существу, представляющих собой формализмы для записи начальных условий некоторой задачи и синтезирующих решение посредством встроенных механизмов логического вывода.
2. Язык Pascal содержит удобные средства для представления данных. Развитая система типов позволяет адекватно описывать данные, подлежащие обработке, и конструировать структуры данных произвольной сложности. Pascal является типизированным языком, что означает фиксацию типов переменных при их описании, а также строгий контроль преобразований типов и контроль доступа к данным в соответствии с их типом (как на этапе компиляции, так и при исполнении программ).
3. Набор операторов языка Pascal отражает принципы структурного программирования и позволяет записывать достаточно сложные алгоритмы в компактной и элегантной форме. Pascal является процедурным языком с традиционной блочной структурой и статически определенными областями действия имен. Процедурный механизм сочетает в себе простоту реализации и использования и гибкие средства параметризации.
4. Синтаксис языка достаточно несложен. Программы записываются в свободном формате, что позволяет сделать их наглядными и удобными для изучения.
5. Паскаль - компилятор, тоесть, прежде чем начать испоолнение программы, Паскаль полностью прочитывает исходный текст, написанный программистом, и составляет последовательность машинных кодов, выполняющую те действия, которые описал программист в hqundmnl тексте. Эта последовательность сохраняется в файл с расширением ".EXE" и является самостоятельным исполняемым файлом, который может быть запущен сам по себе, уже без участия Паскаля и, даже, на другом компьютере, на котором Паскаль может быть не установлен.

TURBO PASCAL.

Прошло много времени с момента появления Паскаля на рынке программных продуктов, прежде чем он получил всеобщее признание. Признание программистов и простых пользователей пришло вследствие появления языка программирования Турбо Паскаль (ТП) -диалекта языка, созданного американской фирмой Борланд. Эта фирма объединила очень быстрый компилятор с редактором текста и добавила к

стандартному Паскалю мощное расширение, что способствовало успеху первой версии этого языка.

В 1985 году на рынке ПЭВМ появился язык программирования Турбо Паскаль (версия 3.0) с компилятором стандартного Паскаля. С тех пор Паскаль стал применяться в общеобразовательных, профессионально-технических школах и в сфере высшего образования в качестве "первого" языка программирования. Благодаря простоте использования язык Турбо Паскаль получил широкое распространение и в любительских кругах. Повышению популярности Турбо Паскаля способствовал набор небольших сопутствующих программ (т.н. Tools), позволяющих получать чрезвычайно компактную, быструю и легко читаемую программу.

Эти качества Турбо Паскаля были высоко оценены и в среде профессиональных программистов. Встроенный редактор текста использует достаточно широко распространенную систему команд, берущую начало от пакета WordStar и хорошо знакомую каждому, кто интенсивно использует ПЭВМ.

В появившемся со временем пакете Турбо Паскаль 4.0 было устранено большинство подвергавшихся критике ограничений компилятора и была повышена производительность системы. Кроме того, новый компилятор версии 4.0 имел существенные отличия от предыдущей версии. Наиболее важным нововведением была UNIT- концепция, заимствованная из языка Модула-2. Это дало возможность реализовать в рамках ТП разработку крупных программных продуктов.

С выходом в свет версии 5.0 ТП получил еще большие шансы на благосклонную реакцию со стороны профессиональных пользователей благодаря встроенному в среду программирования интегрированному отладчику, который позволил повысить производительность труда.

Существенно улучшила технические характеристики ТП реализация аппарата перекрытий (overlays), позволяющего строить мощные программные комплексы, рассчитанные на эксплуатацию в малых по объему областях памяти. Суть механизма перекрытий сводится к делению программы на части, поочередно загружаемые по мере необходимости с дискеты или магнитного диска (винчестера) в одну и ту же область памяти, заменяя при этом находившуюся там часть программы.

Кроме того, в ТП 5.0 были расширены возможности отладки (debugging) программ и обеспечена возможность поддержки расширенной помята в стандарте Lotus-Intel-Microsoft (LIMS/EMS 4.0). Сокращение EMS обозначает Expanded Memory Specification (спецификация расширенной памяти). Нельзя путать этот вид дополнительной памяти с другим - Extended Memory (сокращенно - XMS). EMS имеется на обычных ПЭВМ класса XT, в то время как Extended Memory -только на машинах AT-класса (с процессором 286, 386 и выше) при объеме памяти свыше 1 Мбайта.

В этой версии были также исправлены и улучшены библиотеки графических процедур, поставляемые вместе с пакетом ТП. При этом обеспечивалась полная совместимость с графическими адаптерами класса VGA (Video Graphics Array).

В рамках версии ТП 5.5 были осуществлены дальнейшие преобразования в

направлении улучшения технических характеристик пакета. Наряду с внутренними улучшениями и новыми возможностями встроенной справочной системы Help и большим набором учебных примеров, важным нововведением явилась реализация в языке концепции объектно-ориентированного программирования (ООП).

Через некоторое время на рынке появилась версия 6.0 ТП, в которой чисто теоретическая концепция объектно-ориентированного программирования была реализована практически с полным набором объектов, которые могли использоваться для решения прикладных задач пользователя. Кроме того, реализация системы меню приведена b соответствие со стандартом SAA (Turbo Vision). В качестве практического примера использования новых возможностей был реализован текстовый редактор, встроенный в IDE - Integrated Development Environment - интегрированную инструментальную оболочку. При этом сторонники программирования на ТП 6.0 получили возможность не только работать со встроенным многооконным текстовым редактором, но и использовать мышь, которая значительно облегчает работу пользователя.

В 1992 году фирма Borland International представила пользователям очередную версию языка программирования Паскаль - Турбо Паскаль 7.0. Наряду со всеми преимуществами, которые ТП 7.0 унаследовал от предыдущей версии ТП (многооконный режим работы, возможность использования мыши, возможность использования при написании программ языка программирования низкого уровня Ассемблер или прямого ввода машинного кода, возможность создавать объектно-ориентированные программы), в нем были произведены изменения и улучшения:

1. Появилась возможность выделять определенным цветом различные элементы исходного текста (зарезервированные слова, идентификаторы, числа и т.д.), позволяющая даже неопытным пользователям устранять ошибки на этапе ввода исходного текста.
2. Язык программирования ТП 7.0 был расширен (появилась возможность использовать типизированный адресный оператор, открытые массивы и строки и т.д.), что предоставило пользователю дополнительные возможности при решении повседневных задач.
3. Был улучшен компилятор, вследствие чего "коды программ" стали более эффективными.
4. Был улучшен интерфейс пользователя. Кроме того, в ТП 7.0 расширены возможности объектно-ориентированного программирования (в частности, расширены и улучшены возможности Turbo Vision).

Интегрированная среда Turbo Pascal-7.0

Огромную роль в массовом распространении Паскаля сыграла компания *Borland International*. Она сумела создать знаменитую *Turbo*-среду разработки. Это был огромный шаг вперед в облегчении процесса программирования.

Почему *Turbo*? *Turbo* в переводе с английского сленга означает ускорение. Компилятор, входящий в состав *Turbo Pascal* очень быстро переводит программу с языка программирования в машинные коды.

Первый вопрос: <u>Алфавит языка Turbo Pascal.</u>

Алфавит

Алфавит - это совокупность допустимых в языке символов. Алфавит Турбо Паскаль включает следующий набор основных символов:

- строчные и прописные латинские буквы:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z

- пробел
- подчеркивание: _
- арабские цифры: 0 1 2 3 4 5 6 7 8 9
- знаки операций: + - * / = <> < > <= >= := @
- ограничители: . , ' ( ) [ ] (. .) { } (* *) .. : ;
- спецификаторы: ^ # $

Алфавит Turbo Pascal

1. Прописные и строчные буквы латинского алфавита.

A, B, C, D, E... a, b, c, d, e...

2. Десятичные цифры. 0 1 2 3 4 5 6 7 8 9

3. Знаки арифметических операций.

+ сложение; - вычитание; * умножение; / деление

4. Знаки операций отношений.

> больше

< меньше

< > не равно

< = меньше или равно

> = больше или равно

5. Специальные символы.

{} - . , : ; ' # [ ] $ ( ) ^ @ _ пробел

: = присвоить

(* *) можно использовать вместо фигурных скобок ({ })

*Лексика языка Pascal

Неделимые последовательности знаков алфавита образуют *лексемы*. В языке Pascal различают такие виды лексем:

- константы;
- зарезервированные слова;
- идентификаторы;
- знаки операций;
- разделители;
- комментарии.

Второй вопрос: <u>Идентификаторы.</u>

Идентификатор является именем, которое использует программист при обращении к какому-то значению. В качестве имен не могут быть использованы зарезервированные слова (слова-символы, например слово "**PROGRAM**"). В стандартном Паскале идентификаторы используются для обозначения переменных, констант, типов, процедур и функций. Имена могут быть длинными, но при трансляции рассматривается ограниченное число символов (по стандарту Паскаля - первые восемь символов, то есть идентификаторы "**dlinniy_identifikator1**" и "**dlinniy_identifikator2**" будут восприняты компилятором как одно и тоже слово). В Турбо-Паскале идентификатор кроме символов букв и цифр может содержать символ "_" (подчеркивание). Подчеркивание полезно, когда имя состоит из нескольких осмысленных слов. В общем случае следом за зарезервированным словом PROGRAM в программе стоит пробел, разделяющий в Паскале слова, и далее - имя, данное программе. Это имя представляет собой пример идентификатора. Идентификатор - имя, свободно избираемое программистом для элементов программы (процедур, функций, констант, переменных и типов данных). При обозначении какого-либо элемента программы с помощью идентификатора Вы должны руководствоваться следующими постулатами:

- Идентификатор должен начинаться буквой или символом подчеркивания "_".
- ТП 7.0 не различает прописные и строчные буквы. Поэтому можно записать **WriteLN**, **Writeln** или даже **wRITeLn**, не опасаясь быть непонятым компилятором. Для него все три записи эквивалентны.
- Начиная со второй позиции, в идентификаторе можно применять наряду с буквами цифры
- Пробел в ТП 7.0 является разделителем и не может стоять внутри идентификатора. Для создания идентификаторов, состоящих из двух слов, можно воспользоваться большими буквами (например,

  ReadText) или символом подчеркивания (**Read_Text**), но не пробелом (**bRead Text** - два отдельных слова).

- Применение других символов (букв неанглийского алфавита, знаков препинания, псевдографических символов и т.п.) в идентификаторах не допускается.
- Зарезервированные слова (такие как **BEGIN**, **END** или **PROGRAM**) в качестве идентификаторов не используются.
- Идентификаторы могут быть любой длины, но сравнение их между собой производится по первым 63 символам.

Кроме того, при написании программы рекомендуется следовать ряду правил хорошего тона, которые упрощают редактирование программы и повышают ее наглядность:

- Изобретая идентификаторы, старайтесь делать их "осмысленными", не экономьте на именах - имя **ReadTest** всегда лучше, чем **RT**. Не бойтесь потратить время на написание длинных идентификаторов. Встроенный в ИПО (Интегрированная Пользовательская Оболочка) редактор предоставляет возможность копировать и переносить фрагменты текста.
- Все структуры языка имеют англоязычные идентификаторы. Для своих элементов Вы можете изобрести русскоязычные идентификаторы (записанные латинскими литерами), например **PROGRAM Privetstvie**. Но для удобства, создавая идентификаторы, выполняйте не транслитерацию русских слов в английские, а перевод на английский язык (приветствие - welcome).

Слово **PROGRAM** и последующий идентификатор представляют собой незаконченный оператор программы. Для правильного оформления его необходимо завершить символом точка с запятой (**;**). Таким образов завершается каждый оператор программы на языке Паскаль, за исключением последнего оператора **END**, после которого всегда ставится точка, тем самым информируя компилятор об окончании текста программы.

Третий вопрос: <u>Служебные слова.</u>

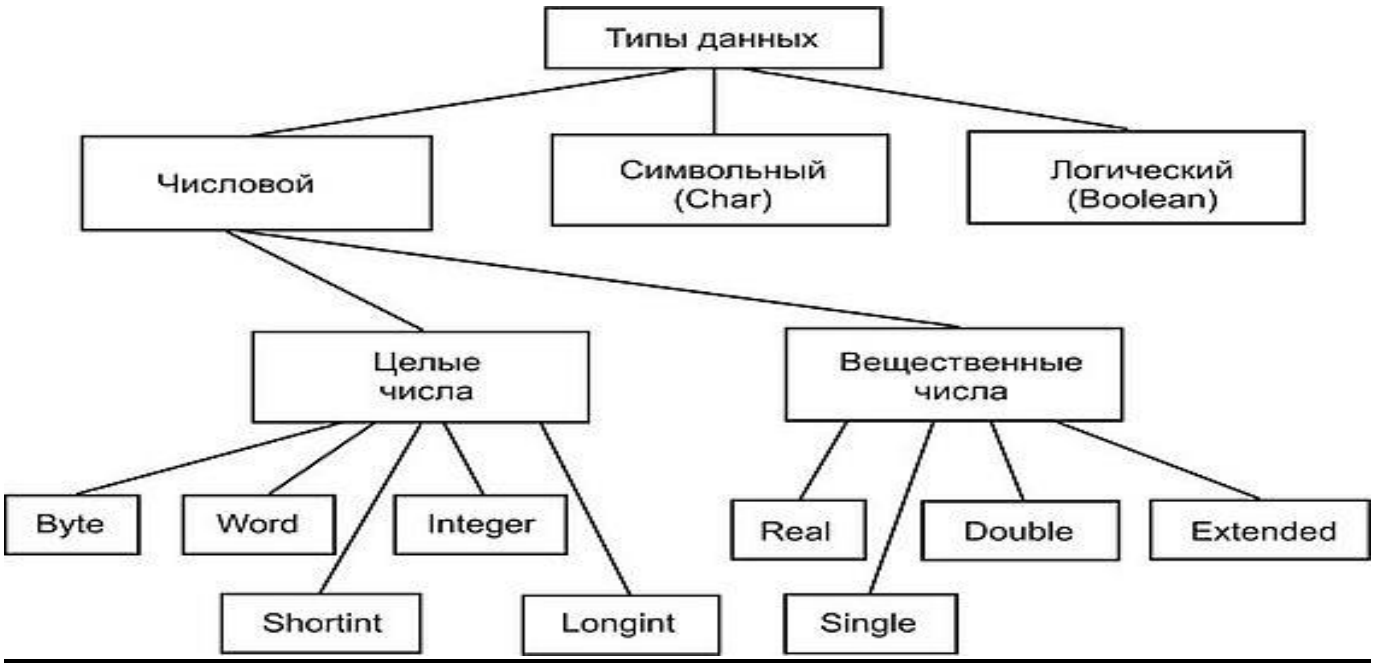
Таблица 3. Некоторые зарезервированные слова (всего 55 слов)

Слово	Смысл слова	Слово	Смысл слова
array	Массив	if	Если
begin	Начало блока	label	Метка
const	Константа	not	Логическое НЕ
div	Деление нацело	of	Из
else	Иначе	program	Программа
end	Конец блока	then	То
file	Файл	type	Тип
for	Для	uses	Использовать
function	Функция	var	Переменная

Словарь языка

Служебное слово языка Паскаль	Значение служебного слова
and	и
array	массив
begin	начало
do	выполнить
else	иначе
for	для
if	если
of	из
or	или
procedure	процедура
program	программа
repeat	повторять
then	то
to	до (увеличивая до)
until	до (до тех пор, пока)
var	переменная
while	пока

Четвертый вопрос: <u>Типы данных.</u>



Порядковые типы языка Паскаль

Идентификатор	Допустимые значения	Формат
ShortInt	-128...127	1 байт со знаком
Integer	-32768...32767	2 байта со знаком
LongInt	-2147483648...2147483647	4 байта со знаком
Byte	0...255	1 байт без знака
Word	0...65535	2 байта без знака
Boolean	false, true	Логический тип (1 байт)
Char	Символы из расширенного набора символов кода ASCII	Символьный (1 байт)

Идентификатор	Длина, байт	Диапазон (множество) значений
Целые типы		
integer	2	–32768…32767
byte	1	0…255
word	2	0…65535
shortint	1	–128…127
longint	4	–2147483648…2147483647
Вещественные типы		
real	6	$2{,}9 \cdot 10^{-39}$…$1{,}7 \cdot 10^{38}$ (11–12)
singl	4	$1{,}5 \cdot 10^{-45}$…$3{,}4 \cdot 10^{38}$ (7–8)
double	8	$5 \cdot 10^{-324}$…$1{,}7 \cdot 10^{308}$ (15–16)
extended	10	$3{,}4 \cdot 10^{-4932}$…$1{,}1 \cdot 10^{4932}$ (19–20)
Логический тип		
boolean	1	true (истина), false (ложь)
Символьный тип		
char	1	все символы кода ASCII

Пятый вопрос: <u>Переменные и константы.</u>

Переменные

Переменная – **это величина, имеющая имя, тип и значение. Значение переменной можно изменять во время работы программы.**

Типы переменных:

- **integer** { целая }
- **real** { вещественная }
- **char** { один символ }
- **string** { символьная строка }
- **boolean** { логическая }

Объявление переменных (выделение памяти):

```
var  a, b: integer;
     Q: real;
     s1, s2: string;
```

Константы

- **В качестве констант в Турбо Паскале могут использоваться:**
- *целые, вещественные и шестнадцатеричные числа,*
- *логические константы, символы, строки символов,*
- *конструкторы множеств*
- *и признак неопределенного указателя NIL.*
- **Ключевое слово const показывает начало раздела описаний именованных констант.**
- Пример
- const e=2.7182818285; lang='Turbo Pascal 7.1';

Элементы языка Pascal: Константы

- **именованные нетипизированные константы** (имеют имя, описываются в специальном разделе **const**, тип определяется автоматически)

```
const n = -10;
      x = 2.5;
      c = 'z';
      s = 'string';
```

- **именованные типизированные константы** - переменные(!) с начальным значением, которое к моменту старта программы уже известно. Следовательно, типизированные константы нельзя использовать для определения других констант, типов данных и переменных. Их значения можно изменять в процессе работы программы.

```
const <имя_константы> : <тип_константы> =
                        <нач_значение>;
```

```
const n: integer = -10;
      c: char = 'z';
```

2.7. Введение в Turbo Pascal

2.7.2. Элементы языка Turbo Pascal

Типизированные константы

Const <идентификатор 1>: <тип 1> = <значение 1>[; <идентификатор 2>: <тип 2> = <значение 2>[; ...]];

Типизированными называют константы, значения которых устанавливаются при описании их типа в разделе описания.

Типизированным константам можно присваивать другие значения в ходе выполнения программы, поэтому фактически они представляют собой переменные с начальными значениями.

Объявление типизированных констант

```
Const
 X : Real = 0.5;
 Str : String = 'Привет!';
 B : Byte = 255;
 I : Integer = 1;
 F : Boolean = True;
 C : Char = 'M';
```

```
Type
 Colors = (black, red, white);
 Digit = '0'.. '9';
Const
 Col : Colors = black;
 Dig : Digit = '5';
```

Переменные и константы. Идентификаторы.

Программы ВР обрабатывают 2 класса данных:

константы ***и*** **переменные.**

сохраняют своё значение в ходе выполнения программы.

могут менять своё значение в процессе выполнения программы.

И переменные и константы имеют своё собственное уникальное имя – идентификатор.

В имени (идентификаторе)

Можно использовать:
буквы латинского алфавита, цифры, знак подчеркивания _ . Имя должно начинаться с буквы.

Нельзя использовать:
пробел, русские буквы, название операторов и стандартных функций языка Паскаль.

В Паскале является обязательным описание всех имен переменных и констант в специальных разделах программы.

Переменные

Переменная – это величина, имеющая имя, тип и значение. Значение переменной можно изменять во время работы программы.

Типы переменных:

- integer { целая }
- real { вещественная }
- char { один символ }
- string { символьная строка }
- boolean { логическая }

Объявление переменных (выделение памяти):

```
var  a, b: integer;
     Q: real;
     s1, s2: string;
```

Строковые константы и переменные

Строка – это последовательность символов. Каждый символ занимает 1 байт памяти (код ASCII). Количество символов в строке называется её длиной. Длина строки может находиться в диапазоне от 0 до 255.

Строковая константа – это последовательность символов, заключённая в апострофы.

Пример: ‘Язык программирования Паскаль’

Описание строковой переменной

Var <имя>: string[N];

Длина строки

Пример: Var name: string[20];
adr: string; {параметр длины может не указываться}

Шестой вопрос: <u>Структура программы.</u>




Структура программы на языке Паскаль

Программа - упорядоченная последовательность действий для ЭВМ, реализующая алгоритм решения какой-либо задачи на языке программирования или в машинных кодах

Заголовок

Program <имя>;

Раздел описания

```
CONST {РАЗДЕЛ ОПИСАНИЯ КОНСТАНТ};
TYPE {РАЗДЕЛ ОПИСАНИЯ ТИПОВ};
VAR {РАЗДЕЛ ОПИСАНИЯ ПЕРЕМЕННЫХ};
LABEL {РАЗДЕЛ ОПИСАНИЯ МЕТОК}
FUNCTION ...      {ОПИСАНИЕ ПРОЦЕДУР И
PROCEDURE ...     ФУНКЦИЙ ПРОГРАММИСТОМ}
```

Раздел операторов

Begin

{ операторы}

End.

Структура программы

```
Program   Имя программы;
Uses      Имя модуля;
Label     Имя метки;
Const     Имя константы = Значение Константы;
Type      Имя типа= Значение Типа;
Var       Имя Переменной: Тип;
Begin
          {инструкция основной программы}
End
```

Синтаксис языка Паскаль

Все программы пишутся английскими буквами, при этом не имеет значение будут они большие или маленькие, но пробелы соблюдать надо обязательно.

Седьмой вопрос: <u>Компиляция программы.</u>

Компиляция - это **процесс перевода программы на язык машинных команд**. Прежде чем программа на Паскале будет выполнена, ее необходимо откомпилировать. В результате компиляции Turbo Pascal создает исполняемый файл с тем же именем, что и файл, содержащий программу, но с расширением .exe. Впоследствии этот файл может быть выполнен, как любой другой исполняемый файл.
Turbo Pascal выполняет компиляцию программы по нажатию клавиши [F9].

Компиляция программы

Восьмой вопрос: <u>Целочисленный и вещественный типы данных.</u>

Сведения изложены в вопросе № 4 «Тип данных».

Целые типы

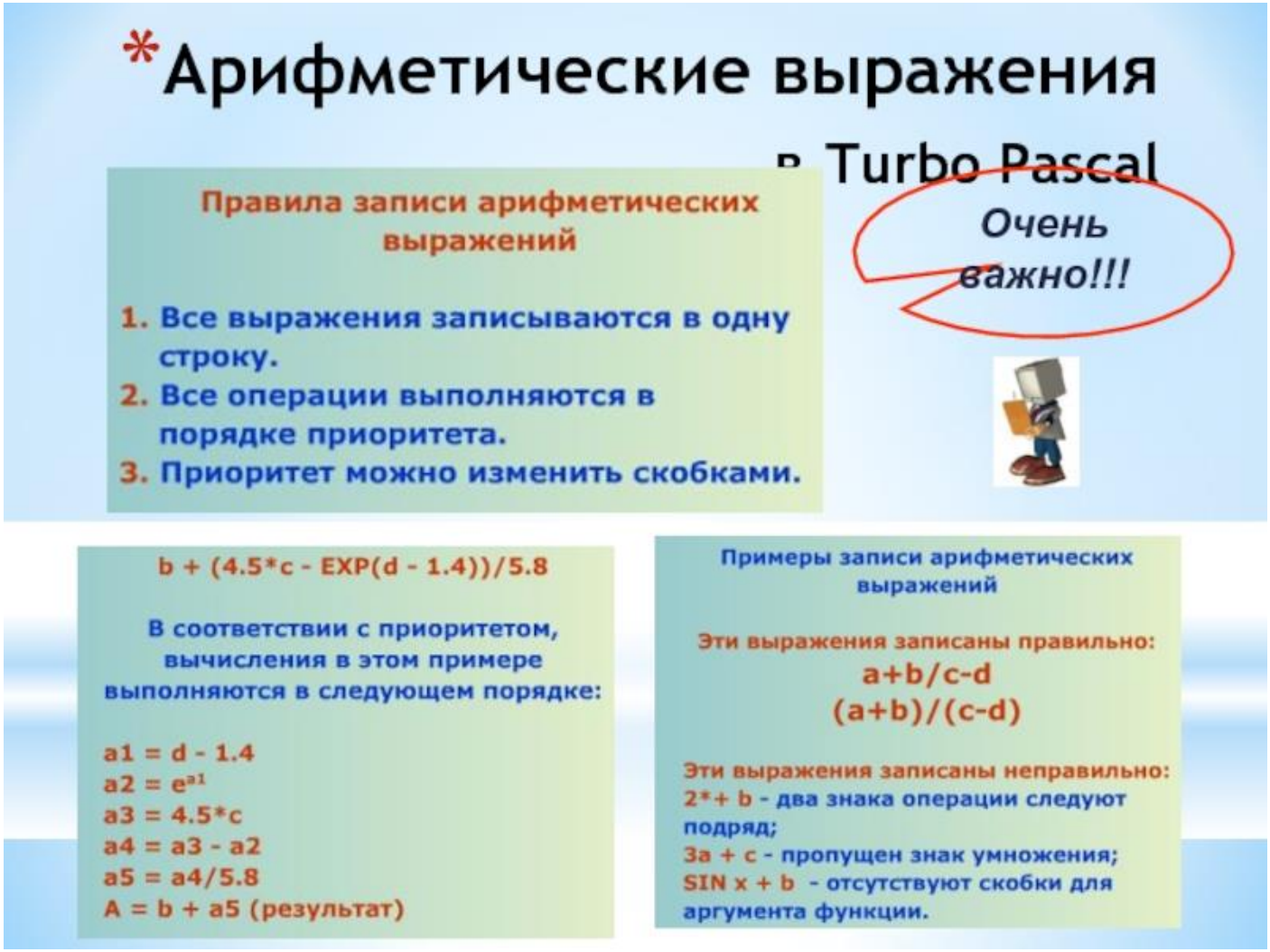
Название	Длина, байт	Диапазон значений
Byte	1	0…255
ShortInt	1	-128…+127
Word	2	0…65535
Integer	2	-32768…+32767
LongInt	4	-2147483648…+2147483647

Вещественные типы

Название	Длина, байт	Количество значащих цифр	Диапазон десятичного порядка
Real	6	11...22	-39…38

Double	8	15…16	-324…+308
Extended	10	19...20	-4951…+4932
Comp	8	19...20	$-2*10^{63}+1…+2*10^{63}-1$

Девятый вопрос: <u>Правила записи арифметических выражений.</u>



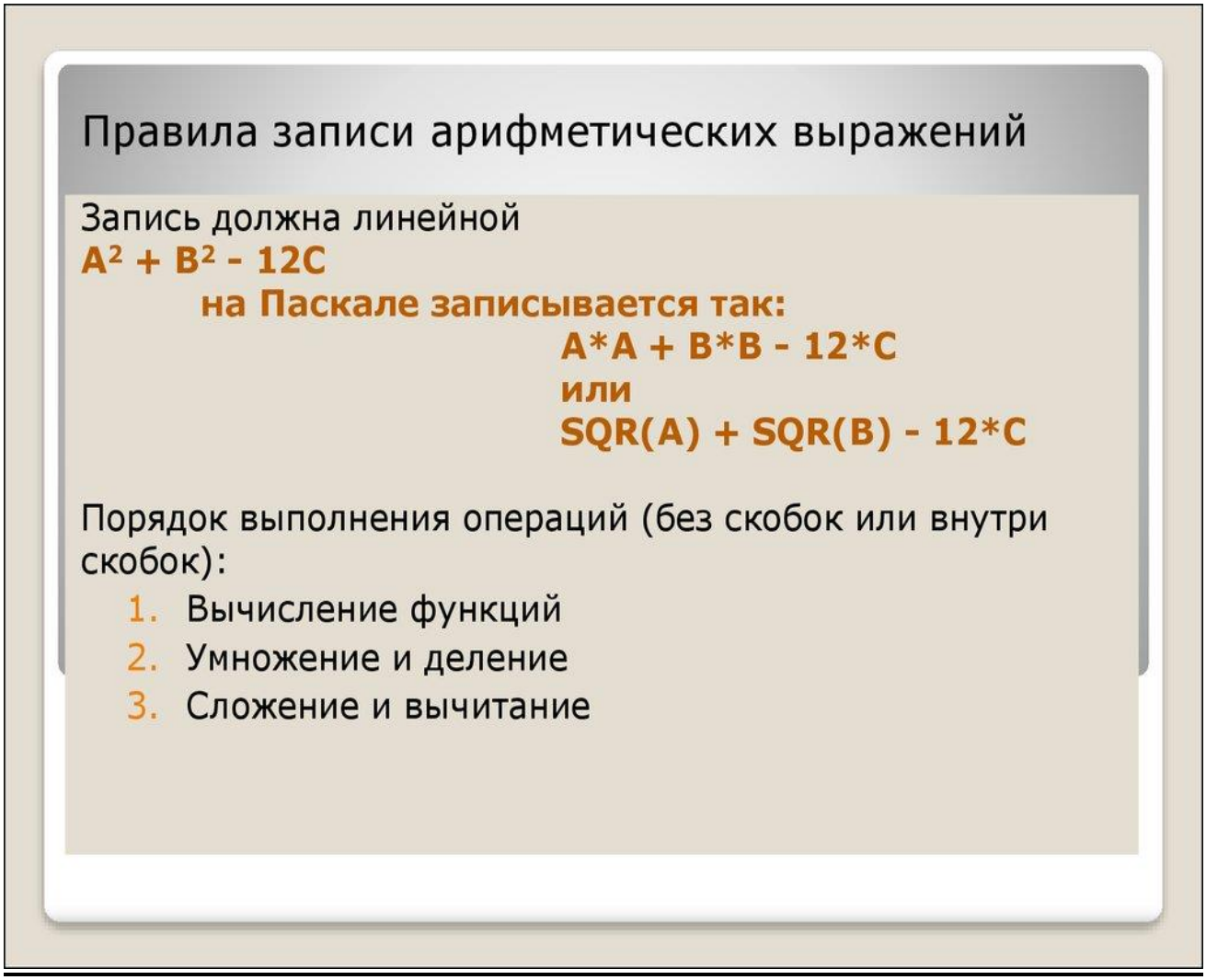
Арифметические функции языка Pascal

Функция	Описание	Пример
`abs(x:R):R` `abs(y:I):I`	вычисление модуля числа |x|, |y|	`abs(-5.5)` ⇒ `5.5` `abs(-12)` ⇒ `12`
`frac(x:R):R`	вычисление дробной части числа	`frac(5.18)` ⇒ `0.18` `frac(4.58)` ⇒ `0.58`
`int(x:R):R`	вычисление целой части числа	`int(5.18)` ⇒ `5.0` `int(4.58)` ⇒ `4.0`
`trunc(x:R):I`	вычисление целой части числа	`trunc (5.18)` ⇒ `5` `trunc (4.58)` ⇒ `4`
`round(x:R):I`	округление числа	`round(5.18)` ⇒ `5` `round(4.58)` ⇒ `5`
`sqr(x:R):R` `sqr(y:I):I`	возведение числа в квадрат	`sqr(2.5)` ⇒ `6.25` `sqr(9)` ⇒ `81`
`sqrt(x:R):R`	вычисление квадратного корня из числа	`sqrt(4.84)` ⇒ `2.2` `sqrt(36)` ⇒ `6.0`

I – целые числа (integer, byte),
R – вещественные числа (real)

Вещественный тип

\+ - * /
= <> < > <= >=
Abs(x) – модуль x
Sqr(x) – квадрат x
Sqrt(x) – корень квадратный из x
Exp(x) – экспонента, e в степени x
Sin(x) – синус x
Cos(x) – косинус x
Ln(x) – натуральный логарифм x
Arctan(x) – арктангенс x
Trunc(x) – отбрасывает вещественную часть (результат – целый)
Round(x) – округляет до целого (результат – целый)



Десятый вопрос: <u>Оператор присваивания.</u>

Оператор присваивания

Простейшие операторы Паскаля

Структура оператора:

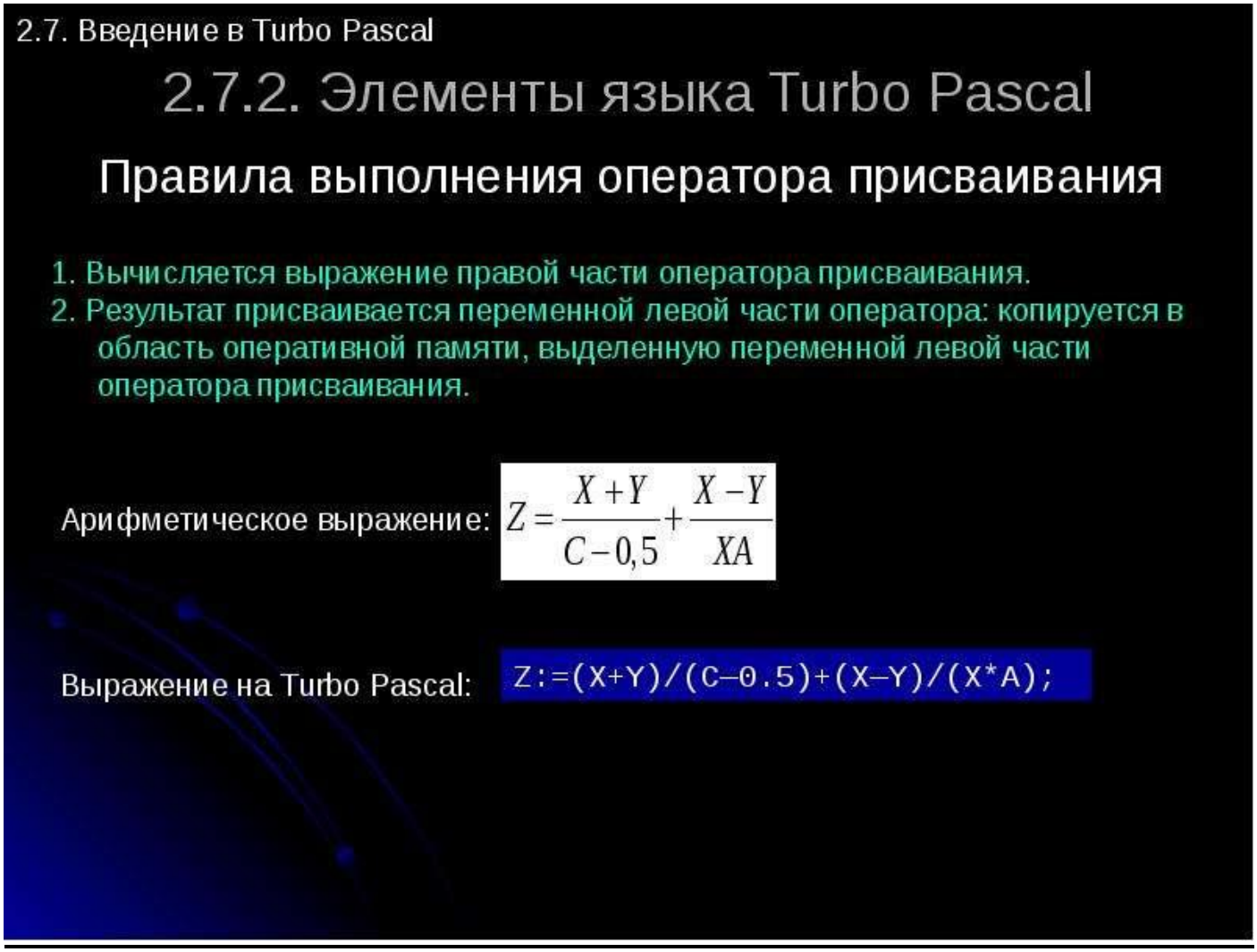
<переменная>:=<выражение>;

Замечания:

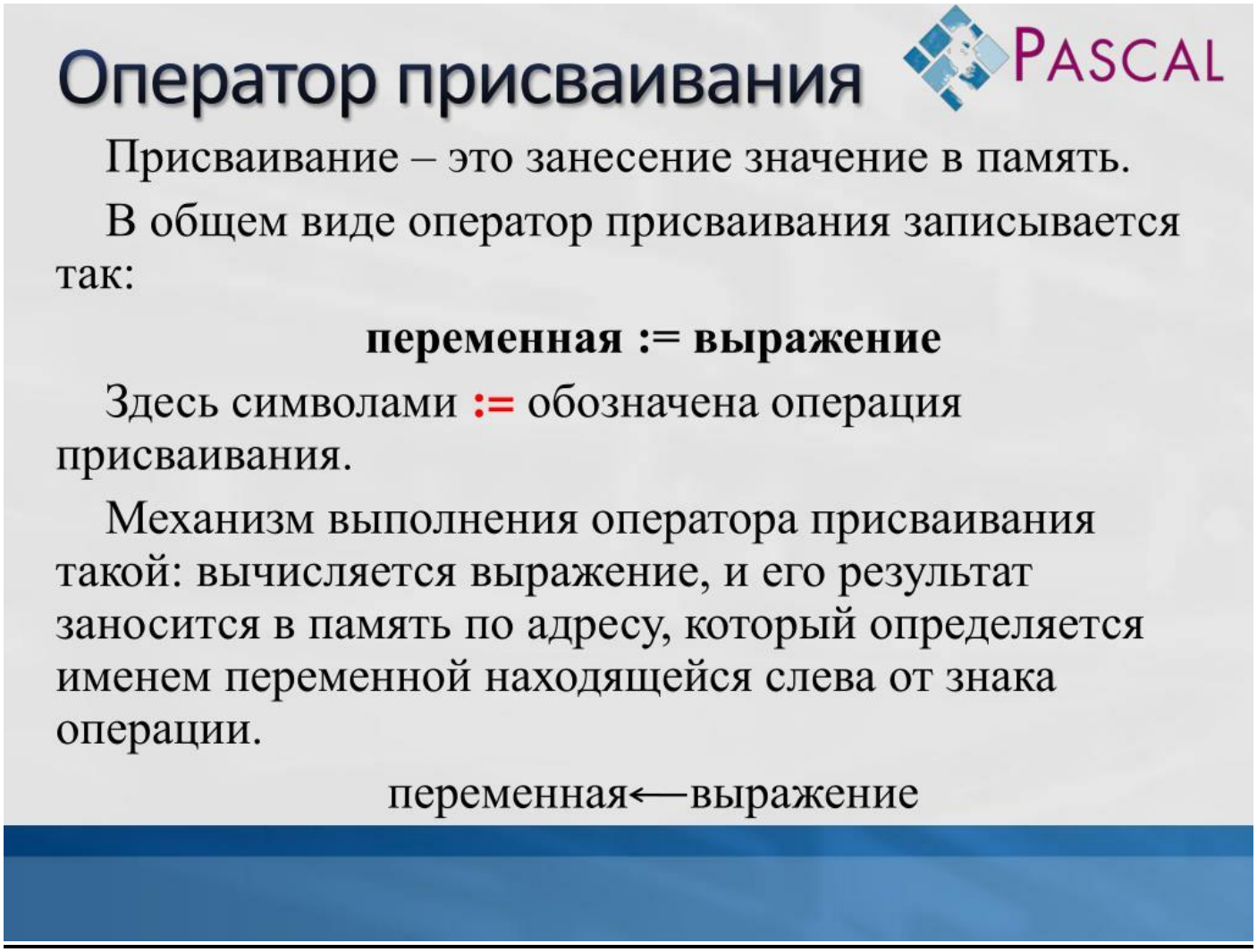
Слева от знака «:=» в операторе присваивания записывается имя той переменной, которой нужно присвоить новое значение, а справа – выражение, значение которого надо сначала вычислить, а затем присвоить указанной переменной

Примеры операторов

Оператор	Комментарий
X:=10;	Переменной X присваивается 10
X:=2*a+6;	Переменной X присваивается значение выражения 2*a+6
A:='Информатика';	Текстовой переменной A присваивается текстовая константа 'Информатика'
S:=S+1;	Переменной S присваивается предыдущее значение S плюс 1



Одиннадцатый вопрос: <u>Аналитический расчет результатов выполнения операции присваивания.</u>



Двенадцатый вопрос: <u>Основные выводы.</u>

Структура программы на языке Паскаль.

Программа на языке Паскаль имеет следующую структуру:
program <имя>;
<описательная часть>;
<раздел функций и процедур>;
begin
<исполнительная часть>;
end.
Зарезервированные слова:
• program <имя> – необязательная строка;
• begin – начало;
• end – конец.
<имя> – присваивается составителем программы (строится по правилам составления имен переменных).

Описательная часть программы.

Все переменные, используемые в программе, должны быть описаны.
Описание начинается со служебного слова var.
Например:
program ff;
var
i,n: integer;
x,y,z: real;
begin;
...
Список переменных от типа отделяется – “:”, одно описание от другого – “;”, список переменных перечисляется через – “,”. Если в программе используются метки, то они описываются с помощью служебного слова label. Метки могут быть числовые и символьные.

В программах на Паскале можно использовать константы, которые описываются с помощью служебного слова **const.**
Const n=100;
Переменная-константа (n) более в программе не описывается, ее тип определяется присвоенным ей числовым значением.

С помощью служебного слова **uses** можно подключать к программе стандартные библиотечные модули. Стандартные модули объединяют

функции определенного назначения и в случае необходимости подключаются к программе.

Например, для использования функции очистки экрана (clrscr) к программе подключают стандартный модуль crt.
Uses crt;

Исполнительная часть программы.

Выполнение программы начинается именно с исполнительной части.

Отдельные инструкции, входящие в программу, называются операторами. Операторы отделяются один от другого – **“;”**.
Операторы бывают трех типов:
1) пустой оператор;
2) простой оператор;
3) составной оператор.
Структура составного оператора:
begin

<оператор 1>; <оператор 2>; …<оператор N>;
end;

Основные операторы.

Оператор присваивания:
<переменная> : = <выражения>;
где «: =» – знак присваивания.
Следующие выражения читаются одинаково
x:=2; x:= 2;
y:=d+beta; y:=d+Beta;
Заглавные и прописные буквы в программе интерпретируются одинаково.

Оператор ввода:

readln (<список - ввода>);

где readln – имя оператора ввода; <список - ввода> – список имен переменных, разделенных запятыми.
Например:
readln (a,b,c);
По данному оператору с клавиатуры необходимо ввести значения
переменных a, b и c.

По оператору
readln ;
компьютер ожидает нажатия любой клавиши. Используется как последний в программе, чтобы успеть записать результаты вычислений.

Оператор вывода:

writeln (<список - вывода>);

где writeln – имя оператора вывода; <список - вывода> – список переменных вывода, разделенных запятыми.
Оператор
writeln ;
Оператор writeln без списка вывода можно использовать для пропуска пустых строк при оформлении вывода результатов.
В операторе writeln можно использовать формат вывода значений переменных.
Например: writeln (’ a = ’, a:8:3, ’ b = ’ , b:3);
Первая цифра (8) после имени переменной вещественного типа определяет количество позиций, выделенных под число, включая знак и
десятичную точку, а вторая цифра (3) определяет количество позиций
выделенных под дробную часть числа. Цифра, стоящая после имени

переменной целого типа, определяет количество позиций, отводимых под число, включая знак.

Комментарии в программе.

В любом месте программы можно записать пояснительный текст – комментарий. Он не обрабатывается во время выполнения программы. Текст комментария ограничен символами { }.

```
…
{ Пояснения к программе – комментарии}
…
```

Комментарии удобно использовать в программе при отладке для временного исключения группы операторов, заключив их в фигурные скобки.