Урок 1

Языки программирования

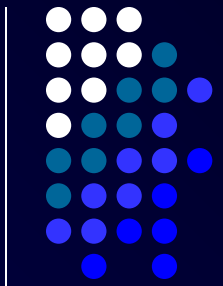
Qbasic и Turbo Pascal 7.0

Цель урока: Дать основные понятия о языках программирования.

План урока:

1. Основные сведения о языках программирования.
2. Основные средства языков. Алфавит языков. Служебные слова.
3. Структура программ.
4. Домашнее задание.

Основные сведения о языках программирования



Под языком понимают любую систему знаков (Знак – это объект, специально выделенный для передачи информации: буква, жест, положение переключателя и т.п.).

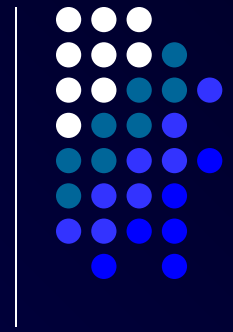
И здесь возникает следующая проблема - язык ЭВМ (машинный язык) весьма далек от понятий, которыми оперирует человек: регистр, переслать, перейти по адресу и т. д., и все это записывается в машинных кодах. Поэтому, чтобы компьютер мог понять написанную программу, она должна быть переведена на язык, понятный самому компьютеру. Этот процесс перевода называется трансляцией.

Существует два различных подхода к трансляции – интерпретация и компиляция:

Интерпретаторы

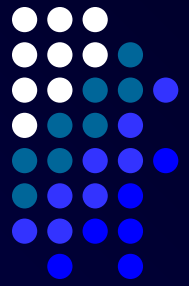
Языки программирования интерпретирующего типа при исполнении программы за один проход переводят в машинные коды одну строку программы. При большом размере программы процесс исполнения готовой программы занимает довольно много времени. В то же время при разработке программ режим интерпретации очень удобен, так как любое внесенное изменение сразу же переводится в машинные коды и исполняется.

Компиляторы



Языки компилирующего типа сначала переводят весь текст программы в машинные коды, а уже затем полученный файл может быть запущен на выполнение. Откомпилированная программа выполняется гораздо быстрее (в 5-10 раз), но наличие ошибок на этапе компиляции требует довольно много времени на их исправление.

ОСНОВНЫЕ СРЕДСТВА ЯЗЫКА

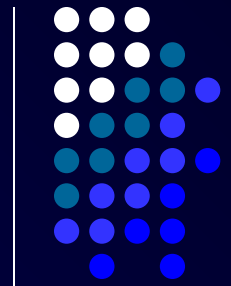


Символы языка - это элементарные знаки, используемые при составлении любых текстов. Набор таких символов называют алфавитом языка.

Алфавит (набор символов) языка QBasic и Turbo Pascal 7.0. включает:

- ✔ все латинские прописные и строчные буквы (A-Z,a-z);
- ✔ арабские цифры 0-9;
- ✔ знаки + - * \ / < > ^ , . ; : ' ( ) _ и др.;
- ✔ служебные слова

ОСНОВНЫЕ СРЕДСТВА ЯЗЫКА

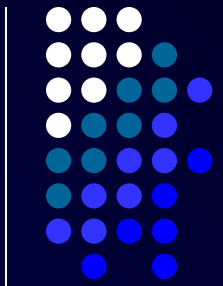


Для записи команд, имен функций, поясняющих терминов QBasic и Turbo Pascal 7.0 предусматривают набор строго определенных слов, которые называются служебными или зарезервированными (это английские мнемонические сокращения).

Служебные слова делятся на три категории:

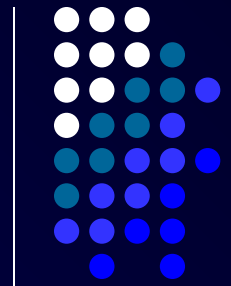
- операторы (коды операций) (PRINT, WRITELN и т. д.)
- функции (имена функций) (SIN,COS и т.д.)
- ключевые слова (AND, VAR, BEGIN, END и т.д.)

Полный список служебных слов приведен в приложениях любого справочника. Их используют только в том значении, которое заранее установлено в языке.



QBasic	Turbo Pascal 7.0
Структура программы:	
	Program имя (input, output);
	Label; - раздел меток;
Const- раздел констант;	*Const; - раздел констант;*
Type - раздел типов;	*Type; - раздел типов;*
DEF - раздел переменных;	*Var; - раздел переменных;*
Procedure - раздел процедур и функций;	*Procedure - раздел процедур и функций;*
Function	*Function*
	BEGIN
оператор 1	*оператор 1;*
оператор 2	*оператор 2;*
. .	……
оператор n-1	*оператор n-1;*
оператор n	*оператор n;*
END.	*END.*

Домашнее задание



Подготовить ответы на вопросы:

Для чего служит компьютер?

Что называется алгоритмом?

Что называется алфавитом языка?

Что такое интерпретация и компиляция?

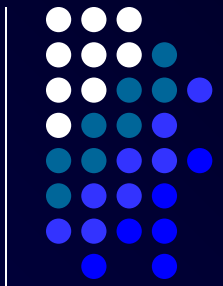
Что представляет собой программа?

Что включает в себя алфавит языков QBasic и Turbo Pascal 7.0?

В каком разделе происходит описание переменных?

Урок 2

Тема урока: **Языки программирования QBasic и Turbo Pascal 7.0.**

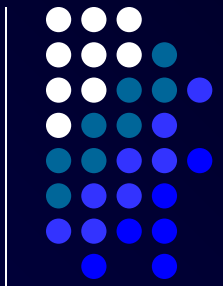


Цель урока: Дать основные понятия о языках программирования.

План урока:

1. Проверка домашнего задания.
2. Переменные, константы в QBasic.
3. Раздел описания типов в Turbo Pascal 7.0.
4. Домашнее задание.

Проверка домашнего задания



Вопросы.

Для чего служит компьютер?

Что называется алгоритмом?

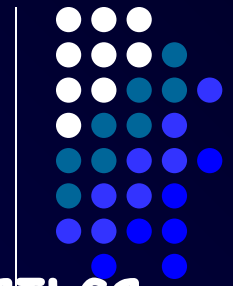
Что называется алфавитом языка?

Что такое интерпретация и компиляция?

Что представляет собой программа?

В каком разделе происходит описание переменных?

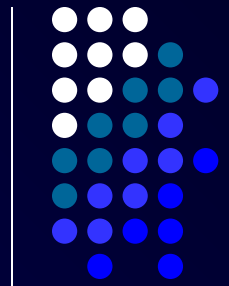
Переменные в QBasic



Переменная - это величина, которая может меняться при выполнении программы. Объявляя переменную или константу заданного типа, Вы отводите в памяти место, где будет храниться ее значение. Тип определяет размер и структуру памяти под переменную.

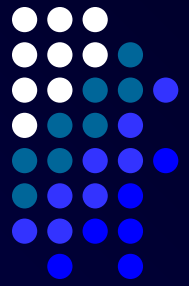
В QBasic существует две основные категории данных: числовые и символьные. Каждая категория включает в себя элементарные типы данных.

Переменные в QBasic



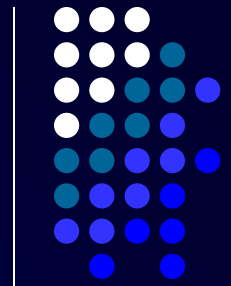
- Переменные числового типа;
- Переменные символьного типа.

ПЕРЕМЕННЫЕ ЧИСЛОВОГО ТИПА



- **ЦЕЛЫЕ (INTEGER);**
- **ДЛИННЫЕ ЦЕЛЫЕ (LONG);**
- **Переменные ОБЫЧНОЙ ТОЧНОСТИ (SINGLE);**
- **Переменные двойной точности (DOUBLE);**

ЦЕЛЫЕ (INTEGER)



ЦЕЛЫЕ (INTEGER) -занимают в памяти 2 байта. Диапазон от-32768 до 32767

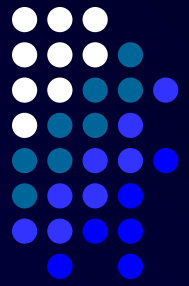
Присвоить переменной целый тип можно следующем образом:

А) поставить в начале программы DEFINT A-B

Б) с помощью суффикса A%

В) использовать оператор описания DIM A AS INTEGER

ДЛИННЫЕ ЦЕЛЫЕ (LONG)



занимают в памяти 4 байта. Диапазон - 2147483648 до 2147483647.

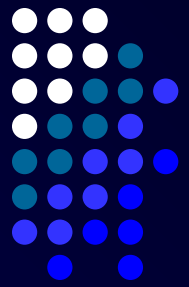
Присвоить переменной тип длинные целые можно следующем образом:

А) поставить в начале программы DEFLNG A-B

Б) с помощью суффикса A&

В) использовать оператор описания DIM A AS LONG.

Переменные обычной точности



Переменные ОБЫЧНОЙ ТОЧНОСТИ (SINGLE) - занимают в памяти 4 байта.

Диапазон от -3.402823E+38 до -2.802597E-45 и 2.8002597E-45 до +3.402823E+38

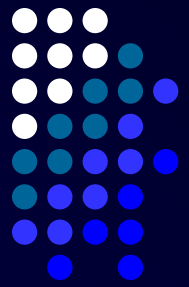
Присвоить переменной тип обычной точности можно следующем образом:

А) поставить в начале программы DEFSGN A-B

Б) с помощью суффикса A!

В) использовать оператор описания DIM A AS SINGLE

Переменные двойной точности (DOUBLE)



занимают в памяти 8 байта.

Диапазон от – 1.7976931348623 31D+308 до -4.940656458412465D -324 и 4.940656458412465D -324 до 1.7976931348623 1D +308

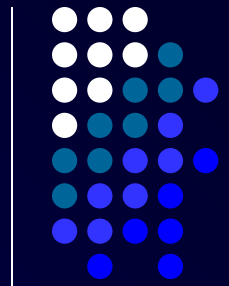
Присвоить переменной тип двойной точности можно следующем образом:

А) поставить в начале программы DEFDBL A-C

Б) с помощью суффикса A#

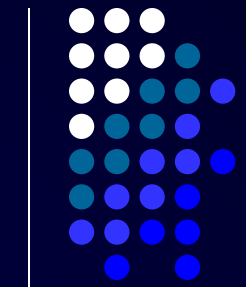
В) использовать оператор описания DIM A AS DOUBLE.

ПЕРЕМЕННЫЕ СИМВОЛЬНОГО ТИПА



- Строка переменной длины (STRING);
- Строка фиксированной длины (STRING*N);
- Переменные пользовательского типа.

СТРОКА ПЕРЕМЕННОЙ ДЛИНЫ (STRING)



СТРОКА ПЕРЕМЕННОЙ ДЛИНЫ (STRING) – это последовательность длиной до 32567 символов таблицы ASCII. В памяти занимает столько байт, какова ее длина +4 байта на описатель.

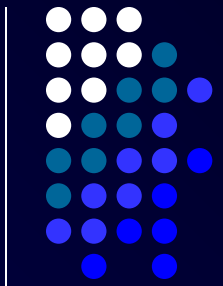
Присвоить переменной символьный тип можно следующем образом:

А) поставить в начале программы DEFSTRING A-C

Б) с помощью суффикса A$

В) использовать оператор описания DIM A AS STRING.

СТРОКА ФИКСИРОВАННОЙ ДЛИНЫ (STRING*N)

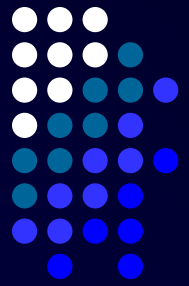


СТРОКА ФИКСИРОВАННОЙ ДЛИНЫ (STRING*N) – символьная строка длинною N байт. В памяти такая строка занимает N байт.

Присвоить переменной символьный тип можно следующем образом:

а) использовать оператор описания DIM A AS STRING*N.

ПЕРЕМЕННЫЕ ПОЛЬЗОВАТЕЛЬСКОГО ТИПА



Если данные, которые вы используете в программе, необходимо сгруппировать по какому-либо признаку, то для этого очень удобно использовать пользовательский тип данных (записи). Он составляется из простых типов данных (числовых и символьных), описанных выше.

Например, нам необходимо ввести табельный номер работника, его фамилию и тарифную ставку.

Определяем пользовательский тип данных оператором TYPE.

TYPE RECORD

Tabnom AS INTEGER

Fam AS STRING*15

Staw AS DAUBLE

END TYPE

Присваиваем переменной RABOT пользовательский тип данных.

DIM RABOT AS RECORD

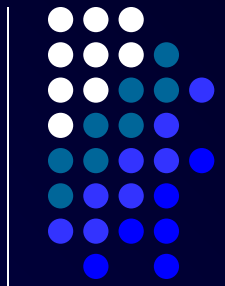
Пользовательский тип данных занимает в памяти столько байт, сколько занимают в сумме каждый из составляющих его элементов. (У нас длина равна 2+15+8=25).

В пользовательском типе данных используются только строки фиксированной длины.

СВОДНАЯ ТАБЛИЦА ОПИСАНИЯ ТИПОВ ДАННЫХ в QBasic

Суффикс	Тип переменной	Объявление DEF_тип	Описание AS_тип	Занимаемый объем
%	**Целые**	**DEFINT**	**INTEGER**	**2 байта**
&	**Длинные целые**	**DEFLNG**	**LONG**	**4 байта**
!	**Обычной точности**	**DEFSNG**	**SINGLE**	**4 байта**
#	**Двойной точности**	**DEFDBL**	**DOUBLE**	**8 байт**
$	**Строка переменной длины**	**DEFSTR**	**STRING**	1 байт на каждый символ + 4 байта на описатель
$	**Строка фиксированной длины**		**STRING*N**	**N байт**
			Пользовательский тип	Занимает столько байт, сколько занимают в сумме отдельные элементы

КОНСТАНТЫ



Числа, символы, строки, которые не изменяют своего значения в процессе выполнения программы - константы.

Неименованные константы символьные, числовые применяются тогда, когда их значение заранее известно и не подлежит изменению.

Например:

Print "Средняя заработная плата"

Именованные константы

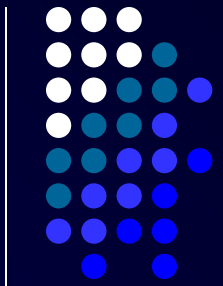
Они также бывают символьные и числовые, тех же типов, что и неименнованные. Чтобы использовать именованную константу, ее необходимо объявить при помощи ключевого слова CONST, например:

CONST M%=66

CONST T$=" число работников"

В дальнейшем к константе можно обращаться по имени.

Раздел описания типов в Turbo Pascal 7.0



В языке Turbo Pascal 7.0 все данные, используемые программой, должны принадлежать к какому-либо заранее известному типу данных.

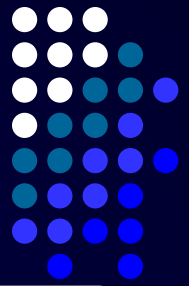
Тип данных определяет:

- формат представления данных в памяти ЭВМ;
- множество допустимых значений;
- множество допустимых операций.

Раздел описания типов в Turbo Pascal 7.0

Типы данных в языке п ния Turbo Pascal 7.0 делятся на пять ос ссов:

Простые типы данных



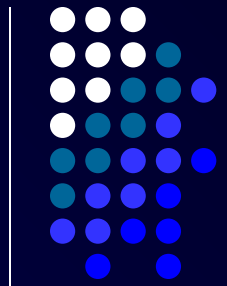
ПРОСТЫЕ ТИПЫ ДАННЫХ

ПОРЯДКОВЫЕ

ВЕЩЕСТВЕННЫЙ

— ЦЕЛЫЙ
— ЛОГИЧЕСКИЙ
— СИМВОЛЬНЫЙ
— ОГРАНИЧЕННЫЙ
— ПЕРЕЧИСЛЯЕМЫЙ

Порядковый тип



В математике порядковым числом называется номер элемента при перечислении.

Под **порядковым типом** понимают тип данных, областью значений которых является упорядоченное счетное множество. Каждому элементу такого множества соответствует некоторое порядковое число, являющееся как раз его номером при перечислении.

Порядковый тип

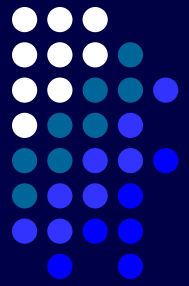
В любом порядковом типе для каждого значения, кроме первого, существует предшествующее значение, и для каждого значения, кроме последнего, существует последующее значение. В языке Turbo Pascal 7.0 существуют стандартные функции, позволяющие определять соответствующие значения для заданного значения:

- функция Pred(x) определяет предыдущее значение величины x,
- функция Succ(x) определяет последующее значение величины x;
- функция Ord(x) возвращает порядковый номер величины x.

Целочисленные типы

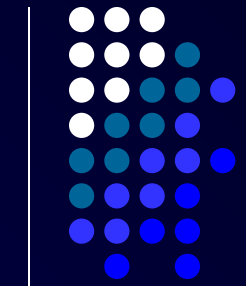
Тип	Диапазон возможных значений	Формат
Shorting	-128.. 127	1 байт со знаком
Integer	-32768..32767	2 байта со знаком
Longint	-2147483648..2147483647	4 байта со знаком
Byte	0..255	1 байт без знака
Word	0..65535	2 байта без знака

Вещественные типы



Тип	Диапазон возможных значений	Точность	Формат
Real	2.9E-39..1.7E38	11—12 знаков	6 байт
Single	1.5E-45..3.4E38	7—8 знаков	4 байта
Double	5.0E-324..1.7E308	15—16 знаков	8 байт
Extended	3.4E-4932..1.1E4932	19-20 знаков	10 байт
Comp	-9.2E18..9.2E18	19—20 знаков	8 байт

Логический тип и логические выражения (BOOLEAN)

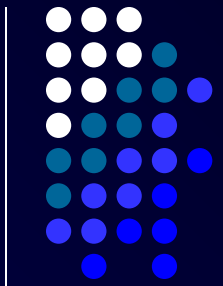


Переменные, константы и выражения логического типа имеют только два значения: **TRUE** (истина) или **FALSE** (ложь)

Например, при сравнении чисел A и B результат будет TRUE (истина) или FALSE (ложь в зависимости от значений A и B).

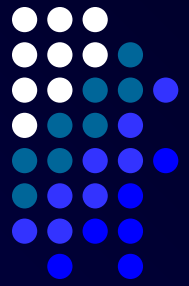
Кроме операций сравнения, логический результат дает функция целой величины **Odd** (нечетный), которая имеет значение TRUE , если ее аргумент нечетный, и FALSE , если ее аргумент четный.

Логический тип и логические выражения (BOOLEAN)



- Для логических переменных определены операции: NOT, AND, OR, XOR. В QBasic определены еще две логических операции: логическое следование IMP и эквивалентность EQW.
- Логический тип, как и целые типы, относятся к порядковым типам. Порядковый тип – это тип данных, для которого определены отношения порядка, то есть для любого элемента можно определить последующий и предыдущий элемент.

Логический тип и логические выражения (BOOLEAN)



Логические операции, операции отношения и арифметические операции часто встречаются в одном выражении. Причем отношения, стоящие слева и справа от знака логической операции, должны быть заключены в скобки, поскольку логические операции имеют более высокий приоритет. Вообще, в логическом выражении принят следующий приоритет операций:

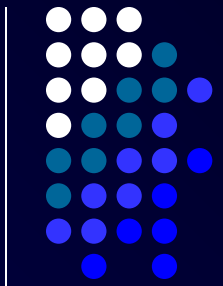
NOT

AND, *, DIV, MOD

OR, XOR, +, -

операции сравнения.

Логический тип и логические выражения (BOOLEAN)



Порядок выполнения операций определяется скобками. Например, в логическом выражении A OR B AND NOT (A OR B) сначала выполняется заключенная в скобки операция OR, а затем операции NOT, AND, OR.

В языке Turbo Pascal 7.0 нет возможности ввода логических данных с помощью оператора read. Однако предусмотрен вывод значений переменных логического типа с помощью оператора write. В этом случае для идентификаторов FALSE и TRUE автоматически отводится по 6 позиций: две — перед словом TRUE и одна - перед FALSE.

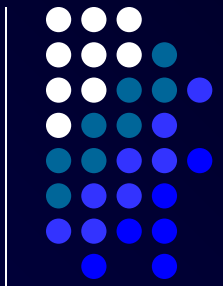
Перечисляемый тип данных

Этот тип данных получил название перечисляемого, потому что он задается в виде перечисления некоторых значений. Эти значения образуют упорядоченное множество и являются константами этого типа. Для объявления переменной список возможных значений, разделенных запятой, указывается в круглых скобках. Например,

Var month: (january, february, marth, april, may, june, july, august, september, october, november, december);

Упорядоченность элементов перечисляемого типа определяется порядком их следования. Самый левый имеет минимальное значение (значение функции ord для него равно 0), а наиболее правый — максимальное.

Символьный тип данных



Описание: идентификатор Char,

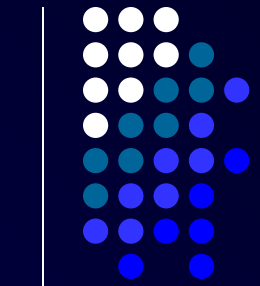
Диапазон значений: значением переменной этого типа может быть любой символ — это буквы, цифры, знаки препинания и специальные символы. Каждому символу алфавита соответствует индивидуальный числовой код от 0 до 255.

Так как символы языка упорядочены, то к символьным данным применимы операции сравнения. Операция сравнения осуществляется следующим образом: из двух символов меньше тот, который встречается в таблице ASCII раньше.

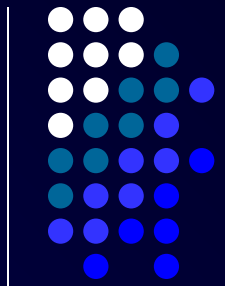
Обычно значения для переменных типа char задаются в апострофах: ch:= '*'; a:= '3'; letter:='G'.

Кроме того, имеется возможность задавать значения указанием непосредственного числового значения ASCII-кода: kd:=#65 {символ 'A'}; s:=#10 {клавиша <Enter>}.

Так как символьный тип является порядковым типом данных, то для него справедливо все, что было сказано о порядковых типах.



Интервальный (ограниченный) тип данных



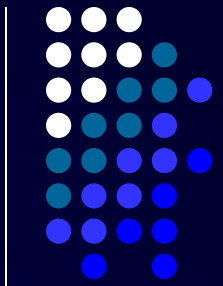
- интервал значений порядкового типа, называемого базовым типом. Описание типа задает наименьшее и наибольшее значения, входящие в этот интервал. Например,

```
Var a:1..25;   ch: 'a'..'z';
```

Здесь переменные a и ch могут принимать значения только из указанного интервала; базовым типом для переменной a является целый тип, а для переменной ch — символьный.

Переменная ограниченного типа сохраняет все свойства переменных базового типа.

Интервальный (ограниченный) тип данных

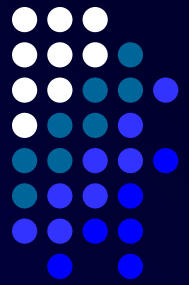


Использование ограниченного типа делает программу более наглядной и понятной. Например, если в программе переменная ***b*** может принимать только значения 3, 4, 5, 6, 7, 8, то лучше описать ее следующим образом:

Var b:3..8; чем Var b: Integer;

так как в случае выхода значения b за диапазон 3..8 в первом случае будет выдано диагностическое сообщение, которое поможет найти ошибку. Во втором случае будет получен неправильный результат, что затруднит поиск ошибки. Таким образом, второй вариант описания переменной следует использовать в тех случаях, когда диапазон значений заранее неизвестен, либо занимает весь допустимый интервал значений для рассматриваемого типа.

Константы и типизированные константы

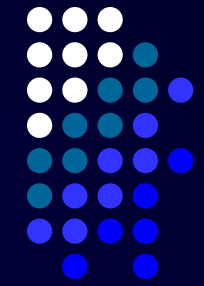


Числа, символы, строки множества, которые не изменяют своего значения в процессе выполнения программы, должны объявляться как постоянные, т.е. константы.

Например:

```
CONST   E=2.718281828;   (Число Эйлера)
C=2.99792458E+8;         (скорость света в м\с)
CUBE =12;                (число ребер куба)
HELLO = 'Привет' ;       (строка – приветствие)
On = true;               (логические константы)
Off = false;
```

Константы и типизированные константы

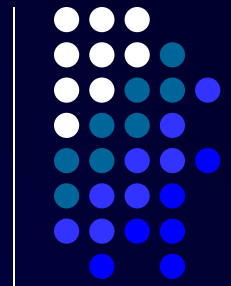


Типизированные константы могут изменять свое значение в процессе выполнения программы. Типизированные константы, в сущности, правильнее считать переменными, но правила их объявления ближе к правилу объявления констант, поэтому они получили такое название.

Например:

```
CONST FLAG: BOOLENT=TRUE;
   A:INTEGER=90;
```

Домашнее задание

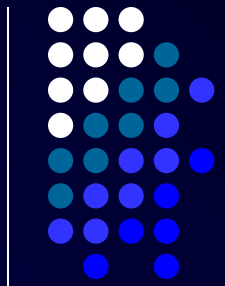


Подготовить ответы на вопросы:

1. Чем характеризуется переменная?
2. Перечислите типы данных?
3. Как описываются переменные?
4. Какова структура программы?
5. С чего начинается основная программа?
6. Чем заканчивается программа?

Урок 3

Тема урока: **Встроенные функции.**



Цель урока: Дать основные понятия о языках программирования.

План урока:

1. Проверка домашнего задания.
2. Встроенные функции в языках QBasic и Turbo Pascal 7.0.
3. Операторы ввода/вывода в языках QBasic и Turbo Pascal 7.0.
4. Первые программы на языках QBasic и Turbo Pascal 7.0.
5. Решение задач.
6. Домашнее задание.

Ход урока

1. Проверка домашнего задания.

Вопросы.

Чем характеризуется переменная?

Перечислите типы данных?

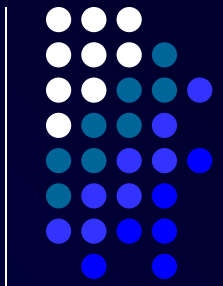
Как описываются переменные?

Какова структура программы?

С чего начинается основная программа?

Чем заканчивается программа?

Встроенные функции



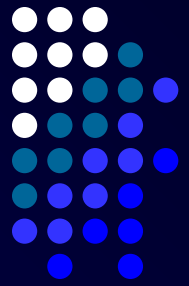
Каждая функция имеет одну форму записи:

Имя-функции (аргумент)

Имя функции выбирается из таблицы, а аргумент записывается в виде арифметического выражения.

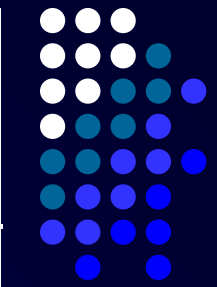
Например: ABS(x).

Некоторые встроенные функции



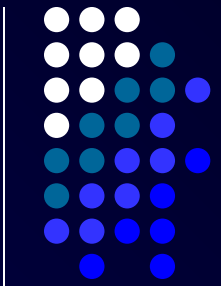
Запись на QBasic	Математическая запись	Запись Turbo Pascal 7.0
ABS(x)	|x|	ABS(x)
SIN(x)	sin x	SIN(X)
COS(x)	cos x	COS(X)
TAN(x)	tg x	—
ATN(x)	arctg x	ArcTAN(X)
INT(x)	целая часть x	INT(X)
SQR (x)	квадратный корень из x	SQRT(X)
RND[(x)]	выдает случайное число из интервала (0,1)	RANDOM[(X)]

Некоторые встроенные функции



Запись на QBasic	Математическая запись	Запись Turbo Pascal 7.0
X2	Квадрат аргумента	SQR(X)
EXP(X)	Показательная функция	EXP(X)
LOG(x)	Натуральный логарифм	LN(N)
---------	Число ПИ	PI без аргумента
SGN(x)	Знак числа	—
CINT(x)	возвращает число равное целой части (округляет по правилам арифметики тип LONGINT	ROUND(X)
X – INT(x)	дробная часть числа	FRAC(X)
FIX(x)	округляет числи, отбрасывая дробную часть числа тип LONGINT	TRUNC(X)
X=X+Y	увеличивает X на величину Y	INC(X,Y)
X=X-Y	уменьшает X на величину Y	DEC(X,Y)

ОПЕРАТОРЫ ВВОДА, ВЫВОДА



- *ОПЕРАТОРЫ ВЫВОДА*

QBasic *Turbo Pascal 7.0*

1. PRINT a1;a2; …;an; *WRITE(a1,a2,…,an) - выводит последовательно значения переменных* *a1,a2, … ,an*

Переход на следующую строку не происходит

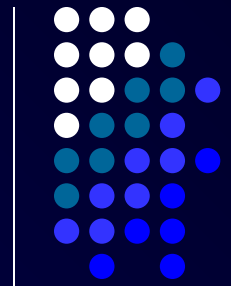
2. PRINT a1;a2; …;an *WRITELN(a1,a2,..,an) выводит последовательно значения a1,a2,.. ,an.*

Переход на следующую строку происходит

3. PRINT *WRITELN - осуществляет переход на новую строку.*

Последовательное расположение операторов 1) и 3) равносильно одному оператору 2).

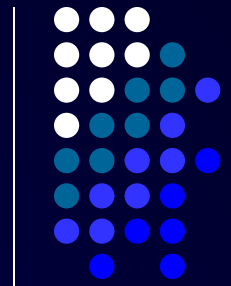
ОПЕРАТОРЫ ВВОДА



QBasic

1. INPUT a1,a2,…,an;
2. INPUT a1,a2,…an
3. INKEY$ - функция ввод/вывода, читающая символы с клавиатуры. Например, ждет нажатия заданного количества символов, пароля или управляющих клавиш. Код клавиши ENTER =13, а код клавиши ESC =27
4. DO
5. PRINT “Для выхода нажмите клавишу ENTER”
6. LOOP WHILE INKEY$ <>CHR(13)

Turbo Pascal 7.01.



1. READ(a1,a2,...,an) - каждое вводимое значение получают последовательно переменные a1,a2,... , an;
2. READLN(a1,a2,..,an) - каждое вводимое значение получают последовательно переменные a1,a2,..., an.
3. READLN - переход на новую строку при вводе данных. Такой оператор применяется, когда исполнение программы желательно задержать до нажатия клавиши ENTER.

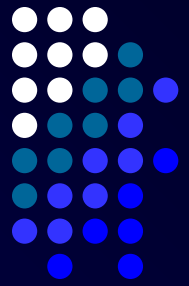
Первые программы на языках QBasic и Turbo Pascal 7.0

Вычислить периметр прямоугольника со сторонами A и B.

Решение.

Обозначим периметр буквой P, тогда P =(A+B)*2

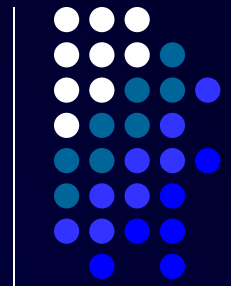
Составим программу при конкретных значениях A и B. Пусть A = 8, B = 21.



‘ PRIM2	Program prim2;
	Uses Crt;
Описываем переменные.	
DEFINT A-B, P	Var a,b,p: real;
Производим очистку экрана	
CLS	Begin
	clrscr;
Присваиваем переменным значения	
A=8	a:=8;
B=21	b:=21;
Вычисляем периметр прямоугольника	
P=(A+B)*2	p:=(a+b)*2;
Выводим значения периметра экран.	
PRINT “ P=”;P	Write('p=' ,p);
END (необязательный оператор)	END.(обязательный оператор)

QB TP

Задания для самостоятельного решения:



А) Занести в переменную P по очереди значения некоторых вышеприведенных функций, аргументом которых является сумма A+B.

P = SIN(A+B) *p:= sin (a + b);*

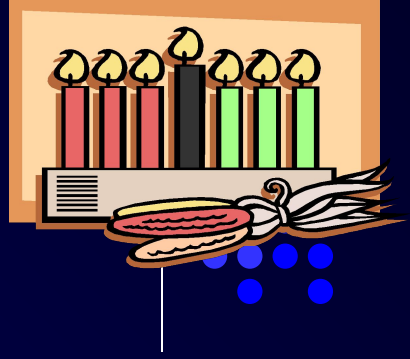
Б) Напечатайте случайное число в пределах от A до B.

P = RND*(B-A) + A *p:= random(b-a+1)+a;*

В) Напечатайте значения переменной P в формате: 3 позиции для целой части и 2 для дробной.

PRINT USING "###.##" ;P *write(p:6:2);*

(6=3+2+1 на точку)



Домашнее задание

Подготовить ответы на вопросы:

Как записывается оператор вывода?

Как записывается оператор ввода?

Как записывается ооператор присвоения?

Чем заканчивается программа?

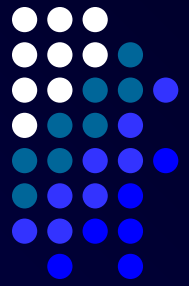
Какая функция используется при записи выражения y=x2+3x—7 на языках QBasic и Turbo Pascal 7.0?

Записать на языках QBasic и Turbo Pascal 7.0 следующие выражения:

$y = 5x^5 - 10x + 2;$

$z = 14x^4-5x^3+11x-17.$

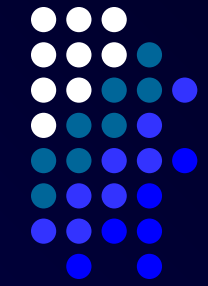
Какие операции можно применять к переменным целого типа?



Урок 4

Условный оператор

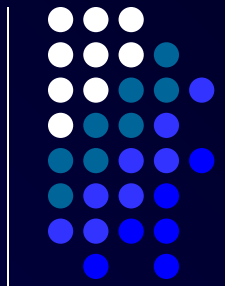
Цель урока: **Показать сходство и различие условного оператора в языках программирования QBasic и Turbo Pascal 7.0.**



План урока:

1. Проверка домашнего задания.
2. Условный оператор.
3. Оператор варианта.
4. Домашнее задание.

Ход урока



1. Проверка домашнего задания.

Вопросы.

Как записывается оператор вывода?

Как записывается оператор ввода?

Как записывается оператор присвоения?

Чем заканчивается программа?

Какая функция используется при записи выражения y=x2+3x—7 на языках QBasic и Turbo Pascal 7.0?

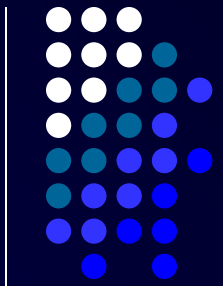
Записать на языках QBasic и Turbo Pascal 7.0 следующие выражения:

y = 5x5 - 10x + 2;

z = 14x4-5x3+11x-17.

Какие операции можно применять к переменным целого типа?

Условный оператор



Условные операторы в QBasic и Turbo Pascal 7.0 помогают нам осуществить "ветвление" программы, т.е. передать управление по условию.

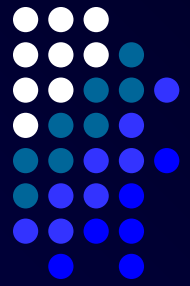
Условный оператор имеет вид:

IF *условие* THEN <*операторы1*> [ELSE <*операторы2*]>

Выполнение условного оператора начинается с вычисления значения логического выражения, записанного в условии. Простые условия записываются в виде равенств или неравенств. Сложные условия составляют из простых с помощью логических операций.

Если условие истинно, то выполняется <операторы1>, в противном случае -<операторы2>.

Даны два числа A и B. Найти наибольшее из них.



```
' prim1                Program prim1;                          uses
   crt;
    Описываем переменные A и B как целые
   DEFINT  A - B       Var a,b: integer;
                       Begin
   CLS                                  Clrscr;
              Вводим два целых числа
INPUT"A=,B="; A,B              Write('введите 2числа');
          Readln(a,b);
                  Если A>B, то   выводим на экран A, иначе B.
IF A>B  THEN PRINT A     If a>b Then Writeln(a)  Else
 ELSE PRINT B            Writeln(b);

End                      End.
```

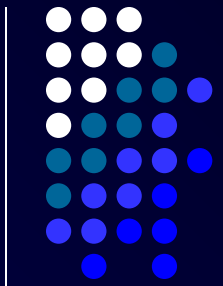
QB TP

Если после THEN или после ELSE располагается целая группа операторов, то можно использовать IF блок, который имеет следующую структуру:

IF условие THEN
Операторы
ELSE операторы
END IF

- В Turbo Pascal 7.0, если в качестве оператора должна выполниться серия операторов, то они объединяются в операторные скобки
- Begin-End

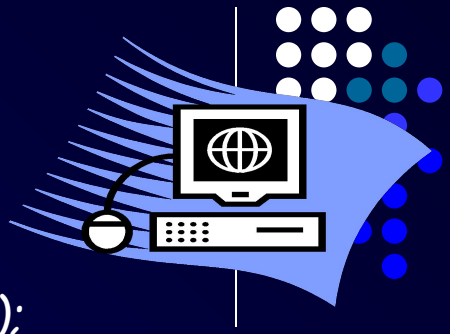
Оператор варианта



Если необходимо осуществить проверку более сложных условий, чем ДА/НЕТ, целесообразно использовать условный оператор

SELECT … END SELECT. ***Case*** *переменная* ***Of***

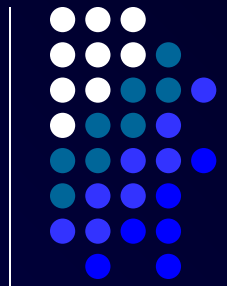
Если выражение выбора отвечает условиям списка выражений данного блока CASE, выполняются операторы из этого блока.



```
'prim2                          Program prim2;
DEFINT N                        var N:integer;
          Вводим целое число N.
INPUT"N=";N                          begin writeln('n='); readln(n);
SELECT CASE N                         Case n of
            Если N<=0, то выводим текст "<=0"-
 CASE IS<=0: ?"<=0"                          -32768..0:  writeln('<=0');
     Если N находиться в пределах от 2 до 9, то выводим текст "2-9"
CASE 2 TO 9: ?"2-9"                    2..9:   writeln('2-9');
          Если N=1 или N=10, то выводим текст "1,10"
 CASE IS=1, 10 : ?"1,10"                      1,10 : writeln('1,10');
CASE ELSE                          else
                      Иначе выводим текст ">10"
    PRINT">10"                           writeln('>10')
               Окончание ветвления.
END SELECT                                    end;
            Окончание программы.
END                                  End.
```

QB TP

Домашнее задание



Имеется условный оператор:

*if D<>10 **Then** writeln('ура!') **Else** Writeln(' плохо...');*

Можно ли заменить его следующими операторами:

*if D=10 **Then** Writeln('ура!') **Else** Writeln('плохо...');*

*if **Not**(D=W) **Then** Writeln('ура!') **Else** Writeln('плохо...');*

*if **Not**(D=10) **Then** Writeln('плохо...') **Else** Writeln('ура!');*

*if **Not**(D<>10) **Then** Writeln('плохо...') **Else** Writeln('ура!').*

Какими будут значения переменных *j, k* после выполнения условного оператора:

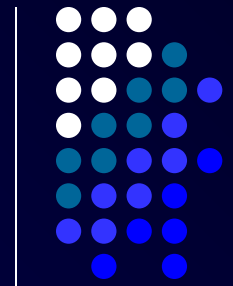
if j>k **Then** j = k-2 **Else** k=k-2 *if j>k **Then** j:=k - 2 **Else** dec(k,2);*

если исходные значения переменных равны:

j=3,k=5;

j=3,k=3;

j=3,k=2.



Урок 5

цикл с параметром

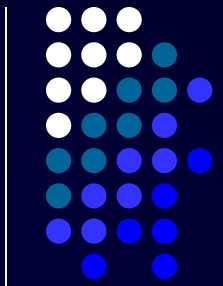
Цель урока:

Показать сходство и различие цикла с параметром в языках программирования QBasic и Turbo Pascal 7.0.

План урока:

1. Проверка домашнего задания.
2. Цикл с параметром.
3. Решение задач.
4. Домашнее задание.

Проверка домашнего задания



1. Имеется условный оператор:

if D<>10 Then writeln('ура!') Else Writeln(' плохо...');

Можно ли заменить его следующими операторами:

if D=10 Then Writeln('ура!') Else Writeln('плохо...');
(НЕТ)

if Not(D=W) Then Writeln('ура!') Else Writeln('плохо...');
(Да, если W=10) if Not(D=10) Then Writeln('плохо...') Else Writeln('ура!'); (НЕТ)

if Not(D<>10) Then Writeln('плохо...') Else Writeln('ура!').
(ДА)

2. Какими будут значения переменных j, k после выполнения условного оператора:

QBasic

```
if j>k Then j = k-2  Else k=k-2
```

Turbo Pascal 7.0

```
if j>k Then j:=k - 2 Else dec(k,2);
```

если исходные значения переменных равны:

j=3, k=5; Ответы: (j=3, к=3)

j=3, k=3; (j=3, к=1)

j=3, k=2. (j=0, k=2)

Цикл с параметром

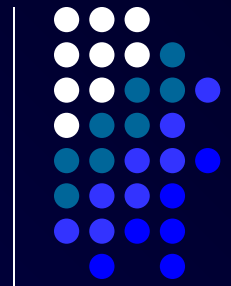
FOR I=A TO B [STEP h] For <параметр>:=A To B Do
<тело цикла>

Если h=1, то шаг можно опустить.

Оператор цикла с параметром применяют тогда, когда известно число повторений одного и того же действия.

Начальное и конечное значения параметра цикла могут быть представлены константами, переменными или арифметическими выражениями.

Цикл с параметром



Рассмотрим, как выполняется оператор цикла с параметром вида

FOR I=A TO B, For <параметр>:=A To B Do <тело цикла>

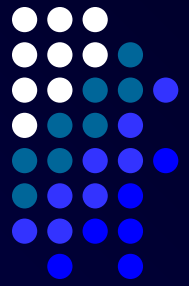
Сначала вычисляются значения выражений A и B. Если A<=B, то <параметр> последовательно принимает значения, равные A, A+1,..., B-1, B, и для каждого из этих значений выполняется <тело цикла>.

Если A>B, то <тело цикла> не будет выполнен ни разу и выполнение цикла с параметром сразу же закончится.

Оператор цикла с параметром

FOR I=B TO A STEP -1 и For <параметр>:=A DownTo B Do <тело цикла> выполняется аналогичным образом, но значение <параметра> изменяется с шагом, равным -1.

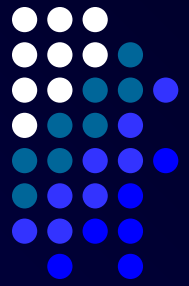
Составить программу вычисления значения выражения y=1+1/2+1/3+… +1/20.



В данном случае целесообразно организовать цикл с параметром, изменяющимся от 1 до 20, то есть шаг изменения параметра равен +1. Обозначим: y — очередное значение суммы дробей; n — параметр цикла. Учитывая это, составим программу:

```
'prim1                  Program prim1;
                      uses crt;
DEFINT N                    Var  n: Integer;
DEFSGN Y                  y: real;
                      Begin
CLS                         clrscr;
                    Задаем начальное значение равное нулю.
Y=0                           y:=0;
```

QB TP



Организовываем цикл с параметром от 1 до 20

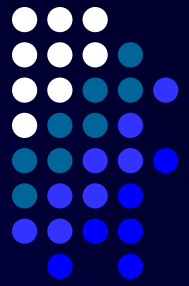
FOR n=1 TO 20	For n:=1 to 20 Do begin
Находим очередную сумму.	
У=У +1/N	y:=y + 1/n;
Выводим на экран очередную сумму.	
?"У=";У	Writeln('y=',y);
NEXT	End.;
End	end.

Самостоятельное задание.

А) Переставьте строки вывода результата за NEXT (End;)

Б) Измените цикл с 20 до 1.

Из чисел от 10 до 99 вывести те, сумма цифр которых равна S(0<S<18).

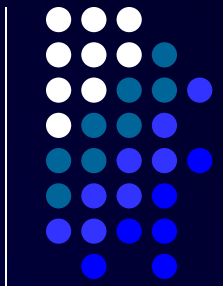


Вопросы для обсуждения

1. Каким действием можно выделить последнюю цифру числа?
2. Каким действием можно выделить первую цифру числа?

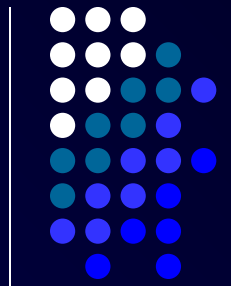
Обозначим: k — это просматриваемое число; p1 — это первая цифра числа k, p2 — это вторая цифра числа k; s — это сумма цифр данного числа k. Число k будем выписывать только в том случае, когда сумма p1 и p2 будет равна s.

QB TP



PRIM2	Program prim2;
	uses crt;
DEFINT K,N,P,SCLS	Var k,n,pl,p2,s:Integer;
	Begin
CLS	clrscr;
Задаем целое число от 10 до 99.	
INPUT"целое число=";N	Writeln(' целое число n='); Readln(n);
Организовываем цикл с параметром	
FOR K=10 TO 99	For k:=10 To 99 Do Begin
Выделяем первую цифру.	
P1=K \10	P1:=k Div 10;
Выделяем вторую цифру	
.P2=K MOD 10	p2:=k Mod 10;
Находим сумму цифр	
S=P1+ P2	s:=p1+p2;
Если сумма цифр равна заданному числу N, то выводим К	
IF S=N THEN PRINT "k=";K	if s=n Then Writeln('k=',k);
NEXT	End;
END	End.

Дано натуральное число n (1000<=N<=9999). Определить, является ли оно палиндромом ("перевертышем"), с учетом четырех цифр. Например, палиндромами являются числа: 2222, 6116, 1441.



Вопросы для обсуждения

Дано число n. Каким образом можно построить "перевертыш" данного числа?

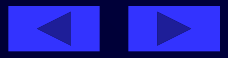
Сколько переменных необходимо для решения данной задачи? Объясните назначение каждой переменной.

Обозначим: n — вводимое число; t — дубликат числа n;

a — перевертыш числа n;

i — переменная цикла для создания перевертыша.

'PRIM3	Program Prim3
	uses crt;
DEFINT A,I, M-N	Var n, m, a, i: Integer;
	Begin
CLS	Clrscr;
Введем четырехзначное целое число	
INPUT"N<=9999";N	Writeln('N<=9999'); Readln(n);
Запоминаем введенное целое число и задаем начальное значение перевертыша	
M=N:A=0	M:=n; a:=0;
Организуем цикл с параметром от 1 до 4.	
FOR I=1 TO 4	For i:=1 To 4 Do; Begin
Находим перевертыш числа N.	
A=A*10+ M mod 10: M=M \ 10	a:=a*10+M Mod 10; m:=m Div 10;
NEXT	End;
Если A=N, то данное число является перевертышем.	
IF A=N THEN PRINT"DA"	If a=n Then Writeln('DA!')
ELSE ?"NO"	Else Writeln('NO');
END	Readln; End



QB TP

Домашнее задание

1. Определить значение переменной S после выполнения следующих операторов:

s=0:n=5 For i=2 To n: s=s+100 \ i :next

s:=0; n:=5 For i:=2 To n Do s:=s+100 Div i

2. Какие из приведенных операторов правильные и почему?

А) FOR I=12 TO 15:S+S+I:NEXT

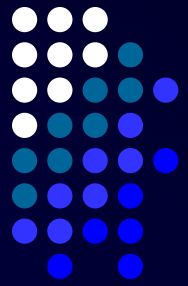
For i:=12 To 15 Do s:=s+i;

Б) FOR A=30 TO 20 For a:=30 To 20 Do

if A Mod 3=0 Then d=d+1 if a Mod 3=0 Then d:=d+1;

NEXT

3. Как выглядит оператор цикла с параметром? Как он работает?



Урок 6

Оператор цикла с предусловием. Оператор цикла с постусловием

Цель урока: Показать сходство и различие вложенных циклов в языках программирования QBasic и Turbo Pascal 7.0.

План урока:

Проверка домашнего задания.

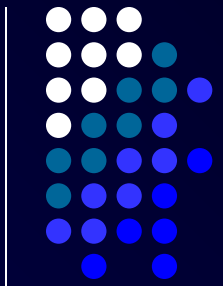
Цикл с предусловием.

Оператор цикла с постусловием

Решение задач.

Домашнее задание.

Проверка домашнего задания



1. Определить значение переменной S после выполнения следующих операторов:

s=0:n=5 For i=2 To n: s=s+100 \ i :next

s:=0; n:=5 For i:=2 To n Do s:=s+100 Div i Ответ:S=128

2. А) правильно Б) неправильно, так как начальное значение меньше конечного.

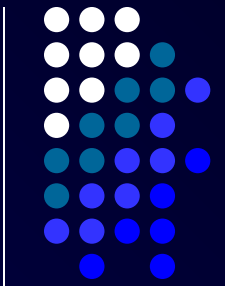
Вопросы.

Как записывается оператор цикла с параметром? Как он работает?

Как записывается полный условный оператор? Как он работает?

Как записывается неполный условный оператор? Как он работает?

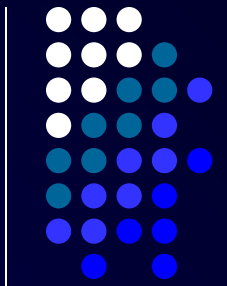
Цикл с предусловием.
While ... Wend While ... Do



Цикл с предусловием используется для программирования процессов, в которых число повторений оператора цикла не известно, а задается некоторое условие его окончания.

Выполнение оператора цикла с предусловием начинается с проверки условия, записанного после слова while. Если оно соблюдается, то выполняется <тело цикла>, а затем вновь проверяется условие и т.д. Как только на очередном шаге окажется, что условие не соблюдается, то выполнение <тела цикла> прекратится.

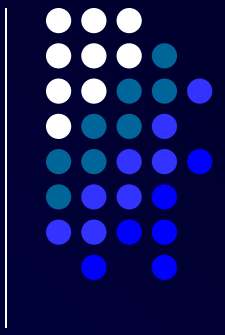
Дано натуральное число n. Посчитать количество цифр в числе



Подсчет количества цифр начнем с последней цифры числа. Увеличим счетчик цифр на единицу. Число уменьшим в 10 раз (тем самым мы избавляемся от последней цифры числа). Далее с получившимся числом проделаем ту же последовательность действий и т.д., пока число не станет равным нулю.

Примечание.

В теле цикла обязательно должен быть оператор, влияющий на соблюдение условия, в противном случае произойдет зацикливание.



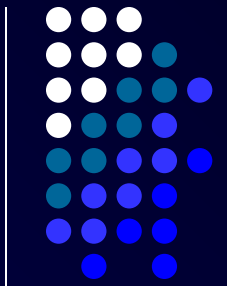
```
'PRIM1

DEFLNG M - N
DEFINT k

CLS
                Вводим целое число.
INPUT"N="; N
     Запоминаем его и счетчику цифр
M=N: K=0
               Пока  m<>0 делать цикл.
WHILE M<>0
 "уменьшаем" число на последнюю
K=K+1: M=M  \ 10
WEND
                          Вывод количества цифр
PRINT " В числе ";N;" - ";K ;" цифр"

END
```

```
Program  prim1;
uses crt;
Var m, n: Longint;
 k: Integer; {счетчик цифр}
Begin
clrscr;

Writeln(' Введите N='); Readln(n);
присваиваем начальное значение.
m: = n; k:=0;

While m<>0 Do    Begin
цифру, т.е. в 10 раз.
k:=k+1;  m:= m Div 10; { Inc(k) }
End;

Writeln('В числе ',n,' - ',k,' цифр!');
Readln;
End.
```

QB TP

Оператор цикла с постусловием DO-LOOP

Repeat (повторять) Until (до тех пор, пока)



Наибольшими в QBasic возможностями обладает следующая конструкция циклов типа DO-LOOP.

DO и LOOP - верхная и нижняя границами цикла, позволяют тестировать условие завершения цикла в верху цикла, в низу цикла, в обоих местах или нигде.

Если цикл должен повториться по TRUE, то используйте управляющее слово WHILE.

Если цикл должен повториться по FALSE, то используйте управляющее слово UNTIL.

Для программной реализации в Turbo Pascal 7.0 циклических процессов с неизвестным числом повторений существует еще один оператор — оператор цикла с постусловием, который имеет следующий вид:

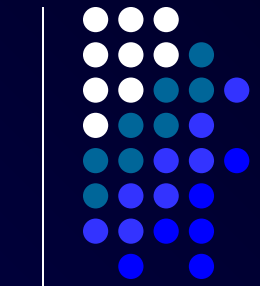
```
Repeat <оператор 1>;
    <оператор 2>;
    <оператор n>;
Until <условие>;
```

где Repeat (повторять), Until (до тех пор, пока …)

- **Отличие** этого оператора от оператора цикла с предусловием: проверка условия производится после очередного выполнения тела цикла. Это обеспечивает его выполнение хотя бы один раз.
- Данный оператор цикла предполагает наличие нескольких операторов в теле цикла, поэтому служебные слова Begin и End не нужны.
- Последовательность операторов, входящих в тело цикла выполняется один раз, после чего проверяется соблюдение условия, записанного следом за служебным словом Until. Если условие не соблюдается, цикл завершается. В противном случае — тело цикла повторяется еще раз, после чего снова проверяется соблюдение условия.

При описании циклов с постусловием необходимо принимать во внимание следующее:

- перед первым выполнением цикла условие его окончания (или продолжения) должно быть определено;
- тело цикла должно содержать хотя бы один оператор, влияющий на условие окончания (продолжения), иначе цикл будет бесконечным;
- условие окончания цикла должно быть в результате выполнено.
- Для досрочного выхода из цикла используют в QBasic операторы EXIT DO (LOOP), в Turbo Pascal 7.0 функции EXIT, BREAK.



Составить программу планирования закупки товара в магазине на сумму, не превышающую заданную величину.

Решение

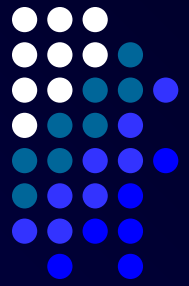
x, k — соответствующие цена и количество товара,

p — заданная предельная сумма,

s — общая стоимость покупки. Начальное значение общей стоимости покупки (***s***) равно нулю. Значение предельной суммы считывается с клавиатуры. Необходимо повторять запрос цены и количества выбранного товара, вычислять его стоимость, суммировать ее с общей стоимостью и выводить результат на экран до тех пор, пока она не превысит предельную сумму ***p***. В этом случае на экран нужно вывести сообщение о превышении:

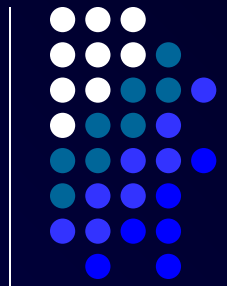
```
'prim9;
DEFINT C, K, P, S


CLS
INPUT "пред сумма = "; P
S = 0
DO
INPUT "цена тов и его кол-во = "; C, K
S = S + C * K
PRINT "стоимость покупки ="; S

LOOP UNTIL S > P
PRINT "суммарная стоим. покупки >
    предел. суммы "

END
```

```
program prim9;
uses crt;
var c,k,p,s:integer;
begin
clrscr;
write('пред сумма = ');readln(p);
s:=0;
repeat
 writeln('цена тов и его кол-во =');
readln(c,k);
s:=s+c*k;
writeln('стоимость покупки =',s);
 until s>p;
writeln('суммарная стоим. покупки >
    предел. суммы ');
readln;
end.
```



QB TP

Самостоятельно



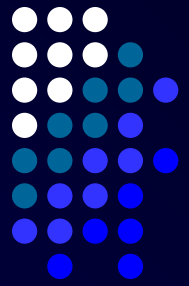
Осуществите досрочный выход из цикла при условии, что Вы приобрели вещи на сумму P/2.

If S>p/2 then exit do

if (s>p/2) then break;

- Алгоритм Евклида — это алгоритм нахождения наибольшего общего делителя (НОД) двух целых неотрицательных чисел.
- Алгоритм Евклида нахождения НОД основан на следующих свойствах этой величины. Пусть x и y одновременно не равные нулю целые неотрицательные числа и пусть x >= y, тогда если y=0, то НОД(x,y) = x, а если y<>0, то для чисел x, y и r, где r — остаток от деления x на y выполняется равенство НОД(x, y)=НОД(y, r).
- Например, пусть x=48, а y=18, найдем их наибольший общий делитель.

X	Y		Результаты
48	18	x>y	
48 mod 8=12	18	x<y	НОД(48,18)=НОД(12,18)
12	18 mod 12=6	x>y	НОД(12,18)=НОД(12,6)
12 mod 6=0	6	x=0	НОД(12,6)=НОД(0,6)
0	6		НОД(0,6)=6



```
'prim3                          Program prim3;
                              uses crt;
DEFINT X,Y                        Var x, y: Integer;
                              Begin
CLS                           clrscr;
```

Вводим два целых неотрицательных числа.

```
INPUT"X=,Y=";X,Y                    Writeln('x:=,y:=');Readln(x,y);
DO                            Repeat
IF X>Y THEN X=X mod Y ELSE Y=Y mod X
                    If x>y Then x:=x Mod y Else y:=y Mod x;
```

До тех пор, пока одно из чисел не станет равно нулю.

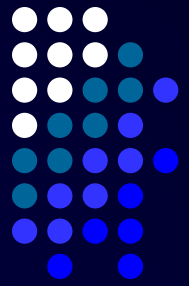
```
LOOP UNTIL  x=0 or Y=0              Until (x=0) Or (y=0);
```

Вывод НОД - без условного оператора, так как одно из чисел равно нулю.

```
PRINT"NOD(A,B)=";X+Y                  Writeln('НОД=',x+y));
                              Readln;
End                               End.
```

QB TP

Домашнее задание.



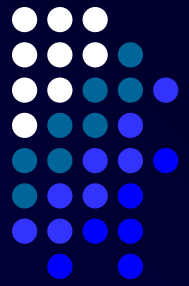
Дана последовательность операторов:

```
a =1 : b=1                          a:=1; b:=1;
while a+b<8                         while a+b<8 do
a=a+1: b=b+2                        Begin a:=a+1; b:=b+2 End;
wend: s:=a+b                        s:=a+b
```

Сколько раз будет повторен цикл, и какими будут значения переменных a, b, и s после завершения этой последовательности операторов?

Определить значение переменной s после выполнения следующих операторов:

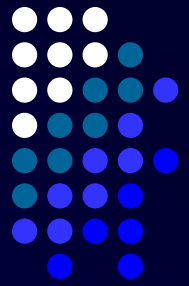
```
s=0 : i:=l                          s:=0; i:=l;
DO                                  Repeat
 s=s+5 Div i : i=l-l                s:=s+5 Div i; i:=l-l;
LOOP Until i<=1                     Until i<=1;
```



Урок 7

Вложенные циклы

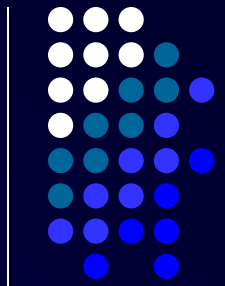
Цель урока: **Показать сходство и различие операторов цикла с предусловием и постусловием в языках программирования QBasic и Turbo Pascal 7.0.**



План урока:

1. Проверка домашнего задания.
2. Вложенные циклы.
3. Решение задач.
4. Домашнее задание.

Проверка домашнего задания



1.1 Дана последовательность операторов:

a =1 : b=1 a:=1; b:=1;

while a+b<8 while a+b<8 do Begin

a=a+1: b=b+2 a:=a+1; b:=b+2;

wend: s=a+b End; s:=a+b

Сколько раз будет повторен цикл, и какими будут значения переменных a, b, и s после завершения этой последовательности операторов?

Ответ: 2 раза, s=8, a=3, b=5

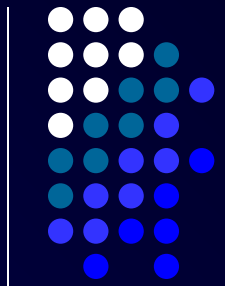
1.2. Определить значение переменной s после выполнения следующих операторов:

s=0 : i=l s:=0; i:=l;

DO : s=s+5 \ i : i= i - l: LOOP Until i<=1 Repeat s:=s+5 Div i; i:= i -l; Until i<=1;

Ответ: s=5

Вложенные циклы



При решении некоторых задач приходится использовать вложенные циклы. Внутренний и внешний циклы могут быть любыми из трех рассмотренных ранее видов: циклами С параметром, циклами с предусловием или циклами с постусловием. Правила организации как внешнего, так и внутреннего циклов такие же, как и для простого цикла каждого из этих видов. Но при использовании вложенных циклов необходимо соблюдать следующее условие: внутренний цикл должен полностью укладываться в циклическую часть внешнего цикла.

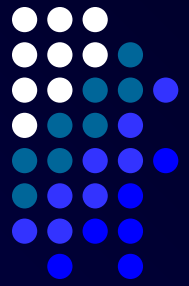
Вложенные циклы

Например.

Даны натуральные числа n и k. Составить программу вычисления выражения 1k+2k+...+nk.

Решение

Для вычисления указанной суммы целесообразно организовать цикл с параметром i, в котором, во-первых, вычислялось бы очередное значение y=ik и, во-вторых, осуществлялось бы накопление суммы прибавлением полученного слагаемого к сумме всех предшествующих (s= s+y).



```
' PRIM 1                    Program Prim1;
                         uses crt;
DEFINT I, K, M-N,S,Y       Var n, k, y, i, s, m: Integer;
                        Begin
CLS                          clrscr;
INPUT" N=,K="; N,K:S=0     Writeln ('n=  k='); ReadLn(n, k); s:=0;
FOR I=1 TO N                For i:=1 To n Do Begin
Y=1                       y:=1;
For M=1 To K                         For m:=1 To k Do Begin
                      Нахождение степени k числа i
Y= Y*I                             y:= y*i;
Next                              End;
                     Нахождение промежуточной суммы.
S=S + Y                          s:=s+y;
Next                              End;
PRINT " Ответ=";S            WriteLn(' Ответ: ',s);
                        Readln;
END                                  End.
```

QB TP

Модифицировать предыдущую программу так, чтобы она вычисляла сумму 11+22+...+nn.

Решение

Данная задача отличается от предыдущей тем, что показатель степени очередного слагаемого совпадает со значением ее основания, следовательно, параметры внутреннего цикла (цикла, в котором вычисляется очередное слагаемое) совпадают с параметрами внешнего цикла.

```
For m=1 To i          For m:=1 To i Do
```

Пример.

Старинная задача. Сколько можно купить быков, коров и телят, если плата за быка 10 рублей, за корову — 5 рублей, за теленка — полтинник (0,5 рубля), если на 100 рублей надо купить 100 голов скота.

Решение

Обозначим через b — количество быков; k — количество коров; t — количество телят. После этого можно записать два уравнения: 10b+5k+0.5t=100 и b+k+t=100. Преобразуем их: 20b+10k+t=200 и b+k+t=100. На 100 рублей можно купить: не более 10 быков, т.е. 0<=b<=10

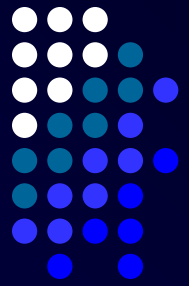
- не более 20 коров, т.е. 0<=k<=20

- не более 200 телят, т.е. 0<=t<=200.

Таким образом, получаем:

```
' PRIM3

DEFINT b,k,t

CLS
For b:=0 To 10
For k:=0 To 20
For t:=0 To 200
If (20*b+10*k+t=200) And
    (b+k+t=100) Then
PRINT"Быков ";b;"коров ";k;"
    телят ";t

NEXT t,k,b

END
```

```
Program prim3
uses crt;
Var b, k, t: Integer;
 Begin
   clrscr;
For b:=0 To 10 Do
For k:=0 To 20 Do
For t:=0 To 200 Do
If (20*b+10*k+t=200) And (b+k+t=100)
   Then
WriteLn('Быков ',b,'коров ',k,' телят
   ',t);

Readln;

End.
```



QB TP

Написать программу, которая находит и выводит на печать все четырехзначные abcd, числа a, b, c, d— различные цифры, для которых выполняется: ab—cd=a+b+c+d.

Решение

Задачу можно решать несколькими способами. Одним из возможных способов является перебор всех четырехзначных чисел и проверка для каждого из них выполнения условий. Попробуем сократить перебор, для этого преобразуем второе условие:

10a+b-(10c+d) = a + b + c + d;

9(a-c)=2(c+d);

(a-c)/(c+d)=2/9

Проанализировав первое условие, получаем, что a=c+2, d=9-c, следовательно 0<= c <=7.

```
'Prim 4                 Program Prim4;
                      uses crt;
DEFINT A-D                  Var a, b, c, d: Integer;
                      Begin
CLS                         clrscr;
For c = 0 To 7                  For c:=0 To 7 Do Begin
A = c+2: d = 9-c                    a:=c+2; d:=9-c;
For b = 0 To 9                      For b:=0 To 9 Do Begin
If b<>c And b<>a And b<>d Then Print a, b, c, d
      If (b<>c) And (b<>a) And (b<>d) Then    Write (a, b, c, d');
Print                               Writeln
NEXT                                End;
NEXT                          End;
                              Readln;
End                           End.
```

QB TP

Если мы сложим все цифры какого-либо числа, затем все цифры найденной суммы и будем повторять много раз, мы, наконец, получим однозначное число (цифру), называемое цифровым корнем данного числа. Например, цифровой корень числа 34697 равен 2 (3+4+6+9+7=29; 2+9=11; 1+1=2).

Составим программу для нахождения цифрового корня натурального числа.

Решение

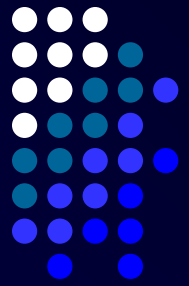
Сколько переменных потребуется для решения задачи, какого типа будут эти переменные?

Всегда ли необходимо вычислять сумму цифр числа (а если введенное число является однозначным)?

Вычислим сумму цифр числа: для этого будем выделять цифры числа и увеличивать текущую сумму. Какую конструкцию необходимо использовать для этого?

В результате выполнения цикла мы получили число. Является ли оно однозначным (корнем данного числа)? Какую конструкцию необходимо использовать для нахождения корня числа? Какие действия должна выполнять программа внутри этой конструкции?

Программа, вычисляющая корень данного числа, может выглядеть следующим образом:



```
'PRIM 5                 Program prim5;
                      uses crt;
DEFLNG N,K,S                Var n,k,s: Longint;
                      Begin
CLS                         clrscr;
INPUT" число N=";N:S=N    Writeln(' число='); Readln(n); s:=n;
                       Пока сумма является двузначным числом.
While S>9                   While s>9 Do    Begin
K=S: S=0                    k:=s; s:=0;
                             Вычисляем сумму цифр числа.
DO                        Repeat
S=S+K Mod 10: K = K \ 10   S:=s+k Mod 10; k:=k Div 10;
LOOP Until K=0                        Until k=0;
Wend                        End;
PRINT "цифр. корень числа  Writeln(' цифр. корень числа ',n,'
 равен ";S                 равен ',s);
                          Readln;
End                       End.
```

QB TP

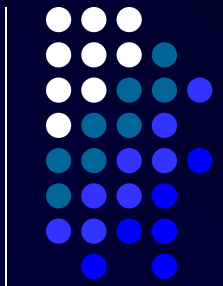
Домашнее задание

Что будет выведено на экране монитора после выполнения следующего фрагмента программы:

```
A=28                    a:=28;
FOR I = 1 TO  A\2       For i:=1 To a div 2 Do   Begin
IF A MOD I=0 THEN PRINT I   if a mod i =0 then
                            Writeln(i);
 NEXT                   End;
```

Решение какой задачи выражает этот фрагмент программы?

Контрольная работа №1



Вариант №1

1. Дано натуральное число:

(a) найти сумму цифр этого числа;

(b) верно ли, что число начинается и заканчивается одной и той же цифрой.

2. Найти все трехзначные числа, такие, что сумма цифр равна A, а само число делится на B (A и B вводятся с клавиатуры).
3. Дано натуральное число. Приписать к нему такое же число.

Вариант №2

1. Дано натуральное число:

(a) найти произведение цифр числа;

(b) верно ли, что в данном числе нет данной цифры A (цифру A вводить с клавиатуры).

2. Найти все трехзначные числа, которые при увеличении на 1 делятся на 2, при увеличении на 2 делятся на 3, при увеличении на 3 делятся на 4, а при увеличении на 4 делятся на 5.
3. Из данного натурального числа удалить все цифры A (A вводится с клавиатуры).

Вариант №3

1. Дано натуральное число:

(a) найти количество цифр данного числа;

(b) верно ли, что данное число заканчивается на нечетную цифру.

2. Найти количество трехзначных чисел, сумма цифр которых равна A, а само число заканчивается цифрой B (A и B вводятся с клавиатуры).

3. Найти все трехзначные симметричные натуральные числа из промежутка от A до B (A и B вводятся с клавиатуры).

Вариант №4

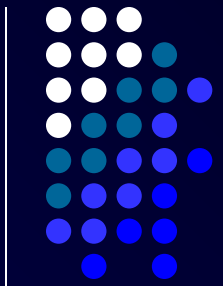
1. Дано натуральное число:

(a) найти количество четных цифр числа;

(b) верно ли, что данная цифра A встречается в числе более двух раз (A вводить с клавиатуры).

2. Найти все четырехзначные числа, у которых сумма крайних цифр равна сумме средних цифр, а само число делится на 6 и 27.
3. Найти количество различных цифр данного натурального числа

Ответы к контрольной работе №1



Вариант №1

N=121, сумма равна 4, верно.

A=15, B=17, 357, 663,816

45, 4545

Вариант №2

A=5, N=121, нет

121, 181, 241,301, 361, 421, 481, 541, 601, 661, 721, 781, 841, 901, 961

N=1234, A=3, N=124

Вариант №3

N=1235, K=4, да

A=17, B=5, 395, 485, 575, 665, 755, 845, 935

A=478, B=535, 484, 494, 505, 515, 525, 535

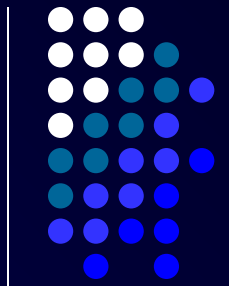
Вариант №4

N=12242, A=2 , цифра 2 встречается более двух раз.

1185, 1458, 1728, 3186, 3456, 3726, 5184, 5454, 5724, 7182, 7452, 7722, 9180, 9450, 8720

N=13234, различных цифр 4.

Контрольная работа №2



Вариант № 1

1. Найти количество делителей натурального числа. Сколько из них четных?
2. Найти все натуральные числа a, b и c из интервала от 1 до 20, для, которых выполняется равенство: a2+b2=c2.

Вариант № 2

1. Найти сумму нечетных делителей натурального числа.
2. Найти все равновеликие прямоугольники, стороны которых выражены целыми числами a и b, а площадь равна S (a и b принадлежат интервалу от 1 до 20, а S вводится с клавиатуры).

Вариант №3

1. Найти все натуральные числа из промежутка от 1 до 200, у которых количество делителей равно N (N вводить с клавиатуры).
2. Найти все натуральные числа a, b и c из интервала от 1 до 20, для которых выполняется равенство: a+b2=c2.

Вариант № 4

Найти количество делителей натурального числа, больших К (К ввод. с клавиатуры).

Найти все такие тройки натуральных чисел x, y и z из интервала от 1 до 20, для которых выполняется равенство: x2—y =z2.

Ответы к контрольной работе №2

Вариант №1

1. N=24, K=7, четных 5.
2. 3, 4, 5 4, 3, 5 5, 12, 13 6, 8, 10 8, 6, 10 8, 15, 17 9, 12, 15 12, 5, 13

12, 9, 15 12, 16,20 15, 8, 17 16, 12, 20.

Вариант №2

1. N=27, нечетных 3.
2. S=42, 3,14 6,7 7,6 14,3

Вариант №3

N=9, 49, 80, 112, 162, 176

3 1 2 5 2 3 7 3 4 8 1 3 9 4 5 11 5 6 12 2 4
13 6 7 15 1 4 15 7 8 16 3 5 17 8 9 19 9 10
20 4 6

Вариант №4

N=196, K=6 , количество делителей больших K пять. Это 7, 14, 28, 49, 98.

2,3,1 3,5,2 3,8,1 4,7,3 4,12,2 4,15,1 5,9,4
5,16,3 6,4,5 6,20,4 7,13,6 8,15,7 9,17,8
10,19,9