Основы алгоритмизации и программирования

Этапы разработки информационных технологий

1. Постановка задачи
2. Математическая модель задачи
3. Разработка алгоритма
4. Разработка визуальной части проекта
5. Код приложения
6. Отладка программы

1 этап. Постановка задачи

Разработать информационную технологию, позволяющую вычислить длину окружности заданной радиусом

Выходные данные: L – длина окружности

Входные данные: π – константа

R – радиус окружности.

2 этап. Математическая модель задачи

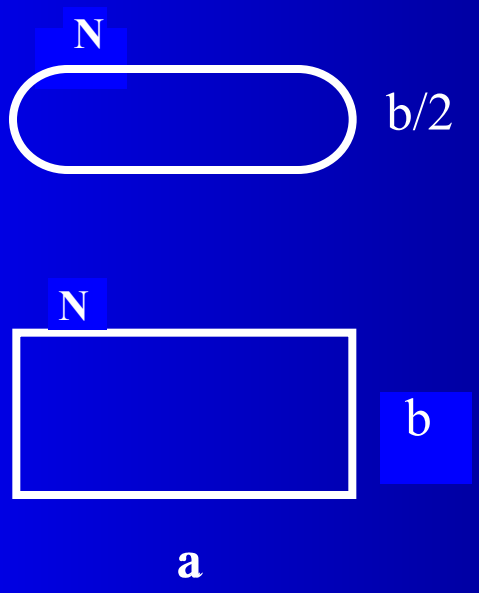
$$L=2\pi R$$

3 этап. Разработка алгоритма

- Алгоритм - последовательность арифметических и логических действий над входными данными, однозначно приводящая к решению задачи.

- Схема алгоритма - графическая интерпретация алгоритма с помощью специальных стандартных блоков.

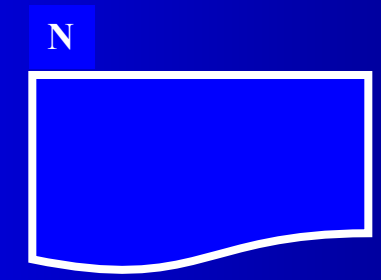
Стандартные блоки для отображения алгоритмов



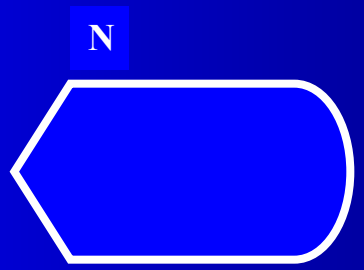
Начало и конец – для отображения вычислительного процесса, а также входа-выхода процесса

Процесс - для отображения операций по обработке данных (присваивание переменной численного значения, значения другой переменной или значения вычисленного выражения)

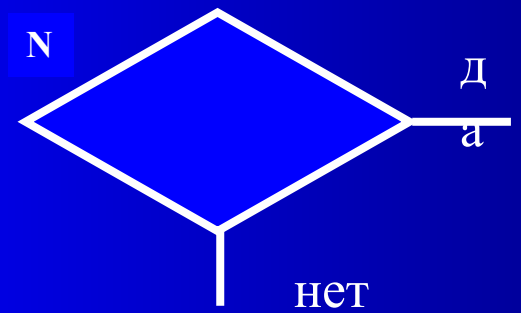
Ручной ввод - ввод значений с клавиатуры



Документ – вывод результатов на печать



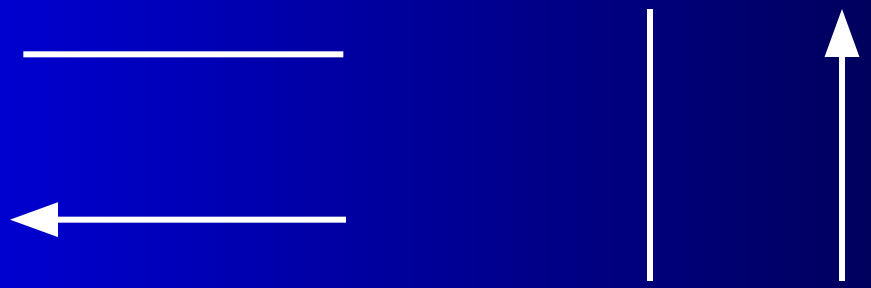
Дисплей – вывод данных на
экран



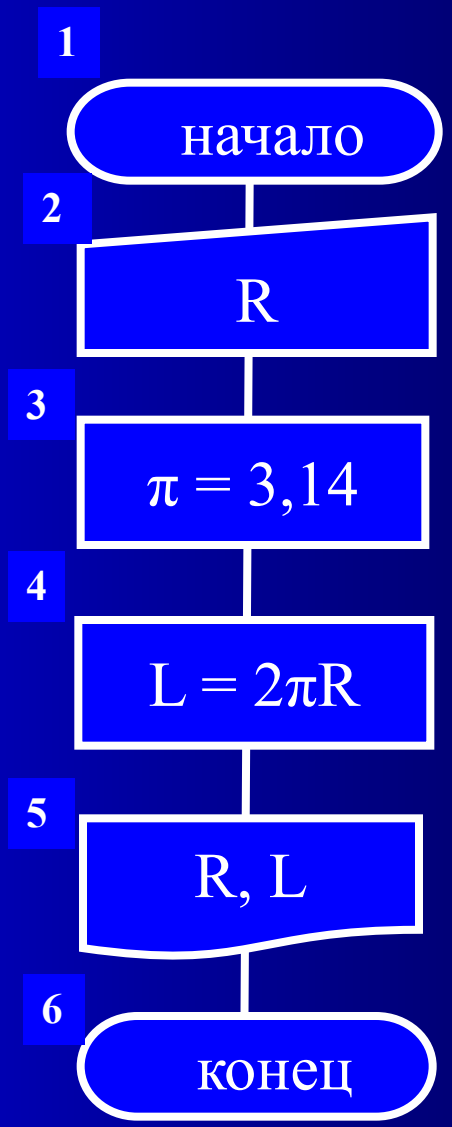
Решение – для записи логических выражений, имеет два вы ода

x

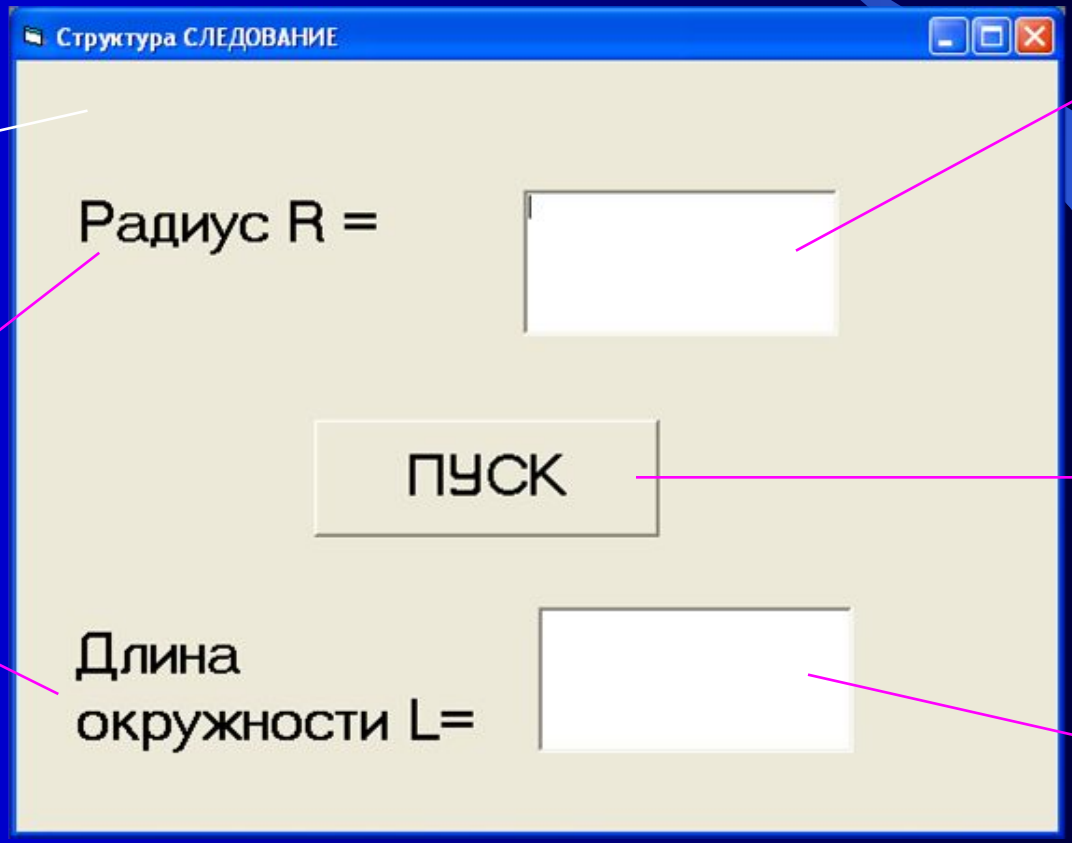
- Передача управления между блоками показывается линиями связи, причем при движении сверху вниз и слева направо стрелки на линиях связи не ставятся, а в обратном направлении стрелки ставятся



- Схема алгоритма для нашего примера



4 этап. Разработка визуальной части проекта



Форма

Label1

Label2

Text1

Command1

Text2

- На форме размещены:

  - два текстовых поля Text1, Text2 для ввода значения радиуса и вывода результата;
  - две надписи Label1, Label2 для пояснений к текстовым полям;
  - одна командная кнопка Command1 для выполнения вычислений

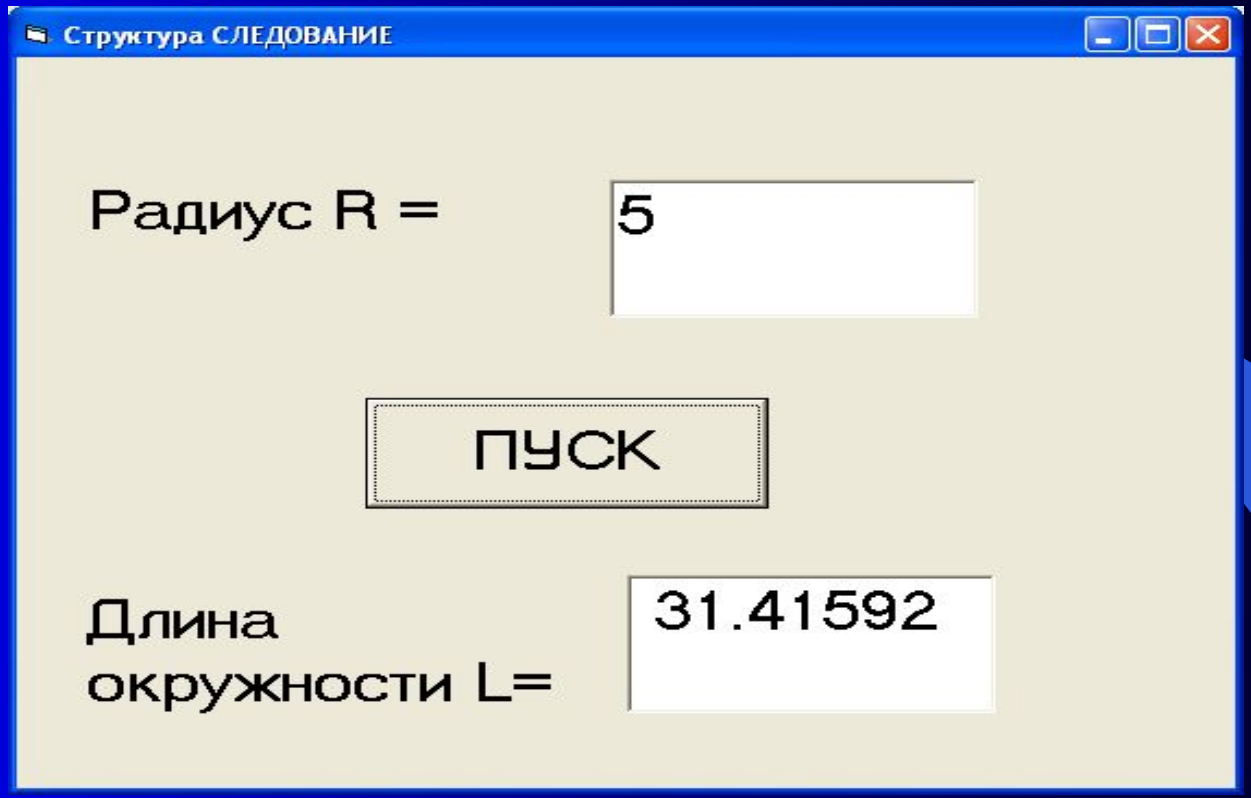
Двойным щелчком по форме открывается окно редактора кода приложения.

5 этап. Код приложения

```
Private Sub Command1_Click()
Const pi As Single = 3.1415926
Dim R As Single, L As Single
R = Val(Text1.Text)
L = 2 * pi * R
Text2.Text = Str(L)
End Sub
```

NB! Код приложения копируется в отчет через буфер обмена

6 этап. Отладка проекта



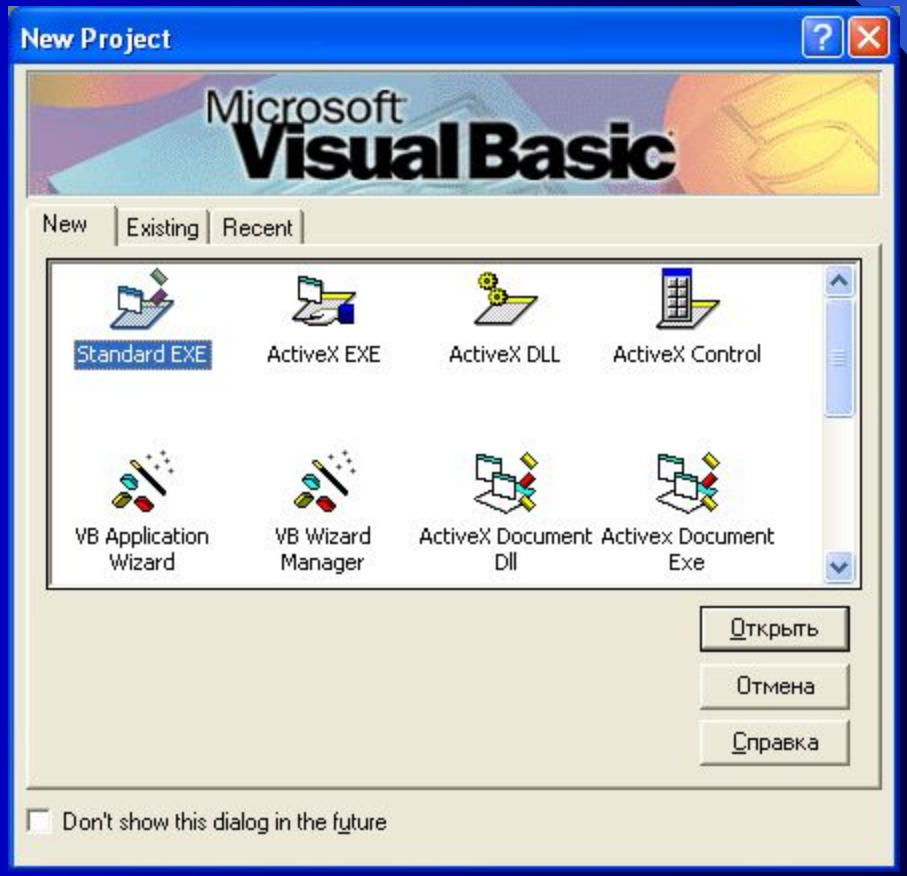
- В отчете приводится форма с исходными данными и результатами.
- Alt+Print Screen копирует в буфер обмена активное окно.

СРЕДА ПРОГРАММИРОВАНИЯ

VISUAL BASIC

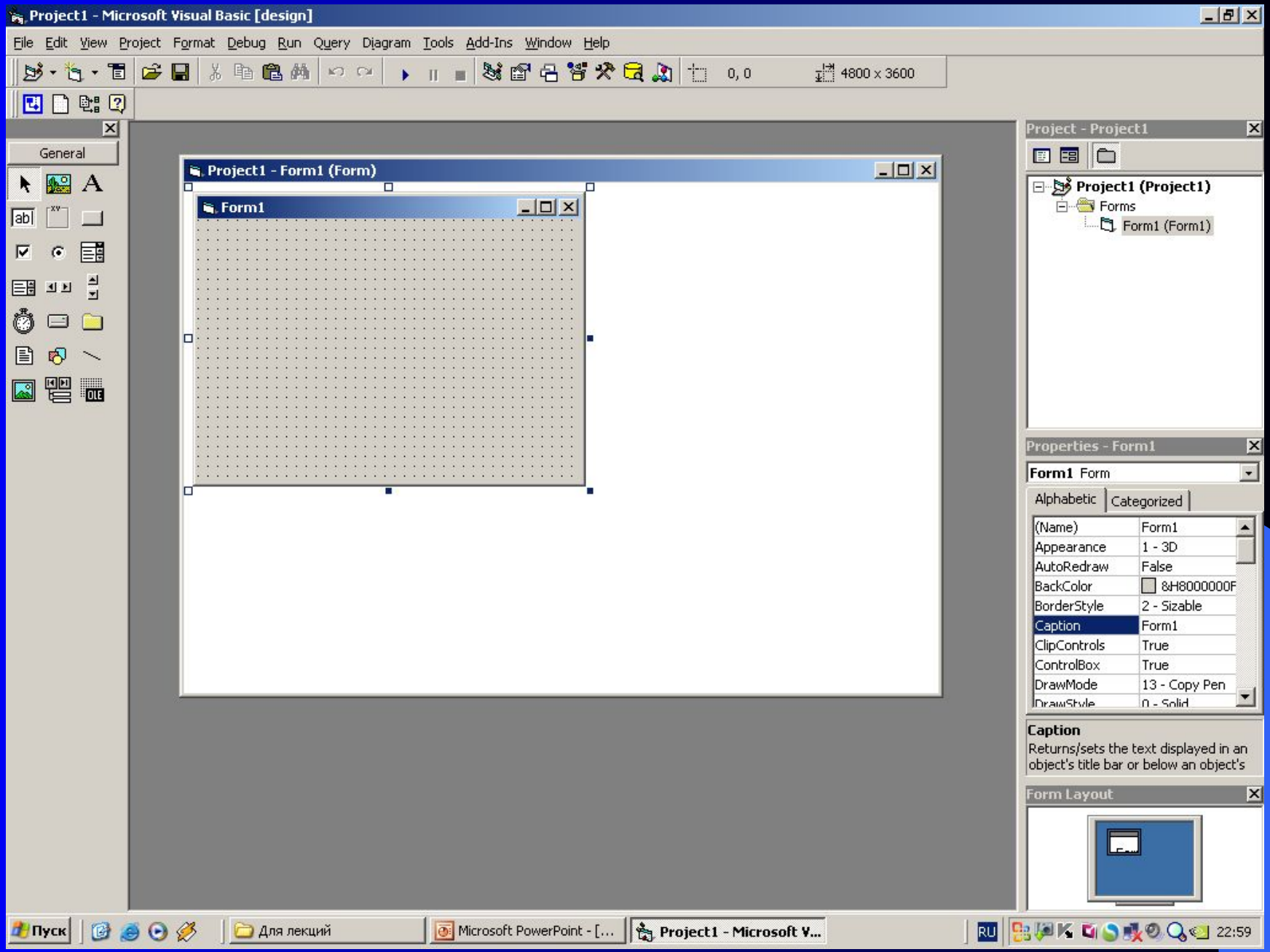
При запуске VB

- на экране отображается диалоговое окно **New Projekt**, в котором можно выбрать один из нескольких типов шаблонов проектов.



- Выбираем **Standart Exe** - стандартный проект (по умолчанию).

- На экране появляется рабочая поверхность **IDE –** Интегрированной Среды Разработки.


Project1 - Microsoft Visual Basic [design]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
0, 0
4800 x 3600
General
Project1 - Form1 (Form)
Form1
Project - Project1
Project1 (Project1)
Forms
Form1 (Form1)
Properties - Form1
Form1 Form
Alphabetic
Categorized
(Name) Form1
Appearance 1 - 3D
AutoRedraw False
BackColor &H8000000F
BorderStyle 2 - Sizable
Caption Form1
ClipControls True
ControlBox True
DrawMode 13 - Copy Pen
Caption
Returns/sets the text displayed in an object's title bar or below an object's
Form Layout
Пуск
Для лекций
Microsoft PowerPoint - [...
Project1 - Microsoft V...
RU
22:59

8. Панель элементов управления - General

Содержит стандартные элементы управления, т.е. объекты системы программирования, которые можно использовать для создания приложения.

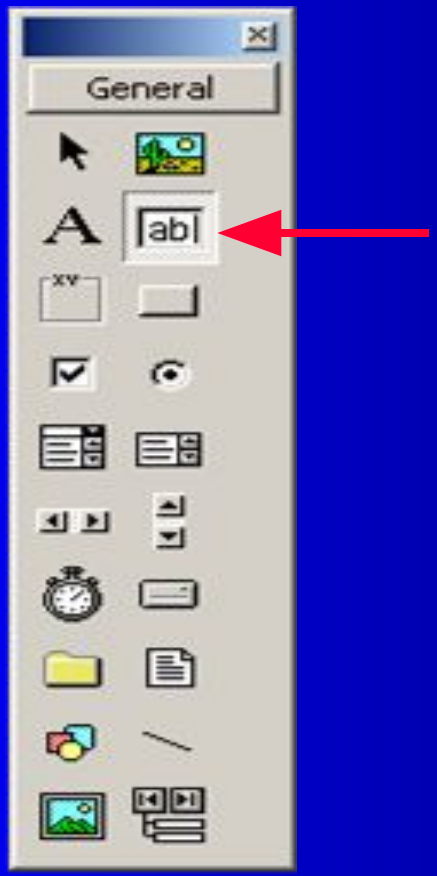
Label – надпись (этикетка)

- Используется для отображения на форме (вывода на форму) текста, заголовков, комментариев, названия объектов и др.

Command Button (командная кнопка)

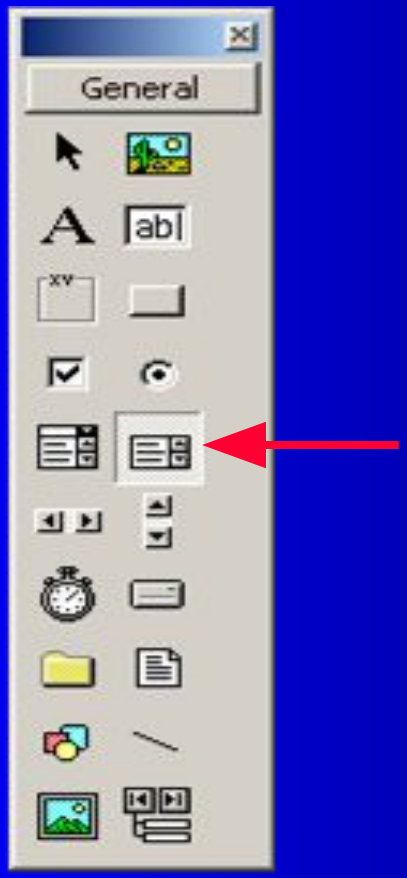
- На рабочей поверхности приложения кнопки играют ту же роль, что и кнопки в жизни.
- Нажатие на кнопку инициирует некоторое действие.
- Кнопку можно помещать в любое место формы. Ее событием является щелчок – Click. Визуально отображается «нажатием» кнопки.

TextBox (Текстовое поле ввода)



- Обеспечивает возможность ввода и вывода текста пользователем.
- Текстовое поле может работать в режиме одной строки или в многострочном, как поле ввода пароля или в режиме «только ввод».

ListBox (окно списка)



- Используется для вывода большого числа элементов списка.
- Мы будем использовать для вывода результатов циклических задач.

- Свойства:
  Name
  По умолчанию List1, List2 и т.д.

Основы языка

Visual Basic

Константа

- Область памяти, имеющая имя.
- Получает своё значение на этапе разработки программы и в процессе её выполнения значение константы изменить нельзя.

- Константа д.б. объявлена:

Const <имя константы>[As<имя типа>]= Значение

Например:

Const Pi As Single = 3.1416

Переменная

- Область памяти, имеющая имя.
- Получает своё значение на этапе выполнения программы и сохраняет его пока ему не будет присвоено новое значение.
- Переменную необходимо объявить, указав её имя и тип:

  Dim <Имя переменной>[As <Имя типа>]

  В одной строке м.б. объявлено несколько переменных:

  Dim x As Integer, b As Single, fi As Double

Математические операции:

^	возведение в степень	
-	отрицание	
*	умножение	
/	деление	7 / 2 = 3.5
\	целочисленное деление	7 \ 2 = 3
mod	остаток от деления на целое	7 mod 2 = 1
+	сложение	
-	вычитание	

Запись арифметических выражений

Выражение – это константа, переменная, функция, числовое или строковое значение или их комбинация, образованная при помощи знаков операций и круглых скобок.

Например:

$\frac{a}{-b}$ → a / - b

ab → a * b

$\frac{a+b}{cd}$ → (a + b) / (c * d) или (a+b)/c/d

Числа с множителем 10 в степени представляются в экспоненциальной форме

$5{,}25*10^{8}$ - 5.25E8

Операции выполняются слева направо с учётом приоритетов и круглых скобок

Оператор присваивания

- <Имя переменной> = <Выражение>
- Символ “ = “ в информатике понимается не как равенство а как процесс присвоения значения, полученного в результате вычисления выражения, записанного справа от знака “ = “ переменной записанной слева от этого знака.
- При этом прежнее значение переменной слева от знака присваивания замещается значением вычисленного выражения.

Например:

1. Pi = 3.1415926
   R = 10.0
   L = 2 * Pi * R

2. a = 2
   a = L

3. Flag = true

4. text = “Информатика”

Функции в языке VB

- Аргументы записываются после имени функции в круглых скобках и отделяются друг от друга запятыми.
- Функции м.б. математические, строковые, финансовые, даты и др.

Математические функции

Sin(x)

Cos(x)

Tan(x) - tg x

Atn(x) - arctg x

Sqr(x) - квадр. корень

Log(x) - нат. логарифм

Exp(x) - e^x

Abs(x) - | x |, абсолютное значение

Аргумент тригонометрических функций должен быть представлен в радианной мере!

Вспомним:

Lg x = Log (x) / Log (10)

Радиан = Градус * Pi / 180

Градус = Радиан * 180 / Pi

Организация ввода данных

Ввод данных
можно осуществлять с помощью

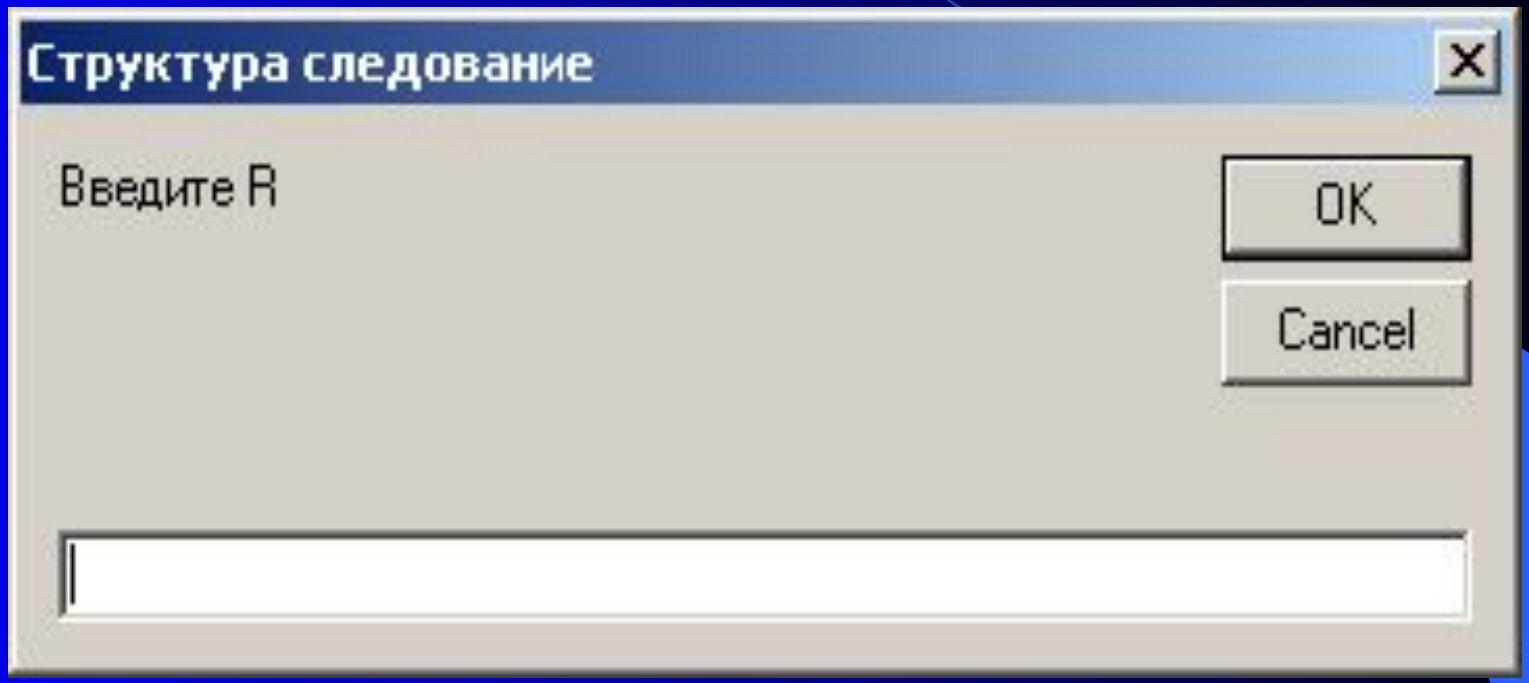
- текстового поля
- с помощью функции *InputBox()*

Функция ввода – *InputBox()*

- Эта функция инициирует создание диалогового окна с текстовым полем для ввода данных
- Имеет три аргумента

InputBox(<Приглашение>,<Заголовок>,[По умолчанию])

R=InputBox("Введите R", "Структура следование")



Метод Print – выводит результаты на форму

- Если выводимые значения в списке разделяются символом « ; », то они печатаются через один пробел.
- Если символом « , », то каждое следующее значение печатается через 14 пробелов.
- Также выводимые значения могут разделяться символом & (конкатенация). В этом случае они печатаются слитно

Операции отношения и логические операции

Операции отношения – < , > , <= , >= , = , <> ,

их результатом всегда является логическое (булево) значение, выражающее истинность некоторого отношения между данными (операндами)

Любая операция отношения может иметь значения:

True – истинно или **False – ложно**.

- Логические операции – выполняются над логическими значениями или логическими выражениями. В результате получаются также логические значения:

Not – логическое **Не** - отрицание
And – логическое **И** - конъюнкция
Or – логическое **Или** - дизъюнкция
Xor – исключающее **Или** когда истинно одно
Eqv – эквивалентность (одинаковость)
Imp – импликация (включение)

A	B	Not A	A And B	A Or B
T	T	F	T	T
T	F	F	F	T
F	T	T	F	T
F	F	T	F	F

Вычислительные процессы и структуры

1. Линейные – структура СЛЕДОВАНИЕ

2. Разветвляющиеся – структура РАЗВИЛКА

3. Циклические – структура ЦИКЛ

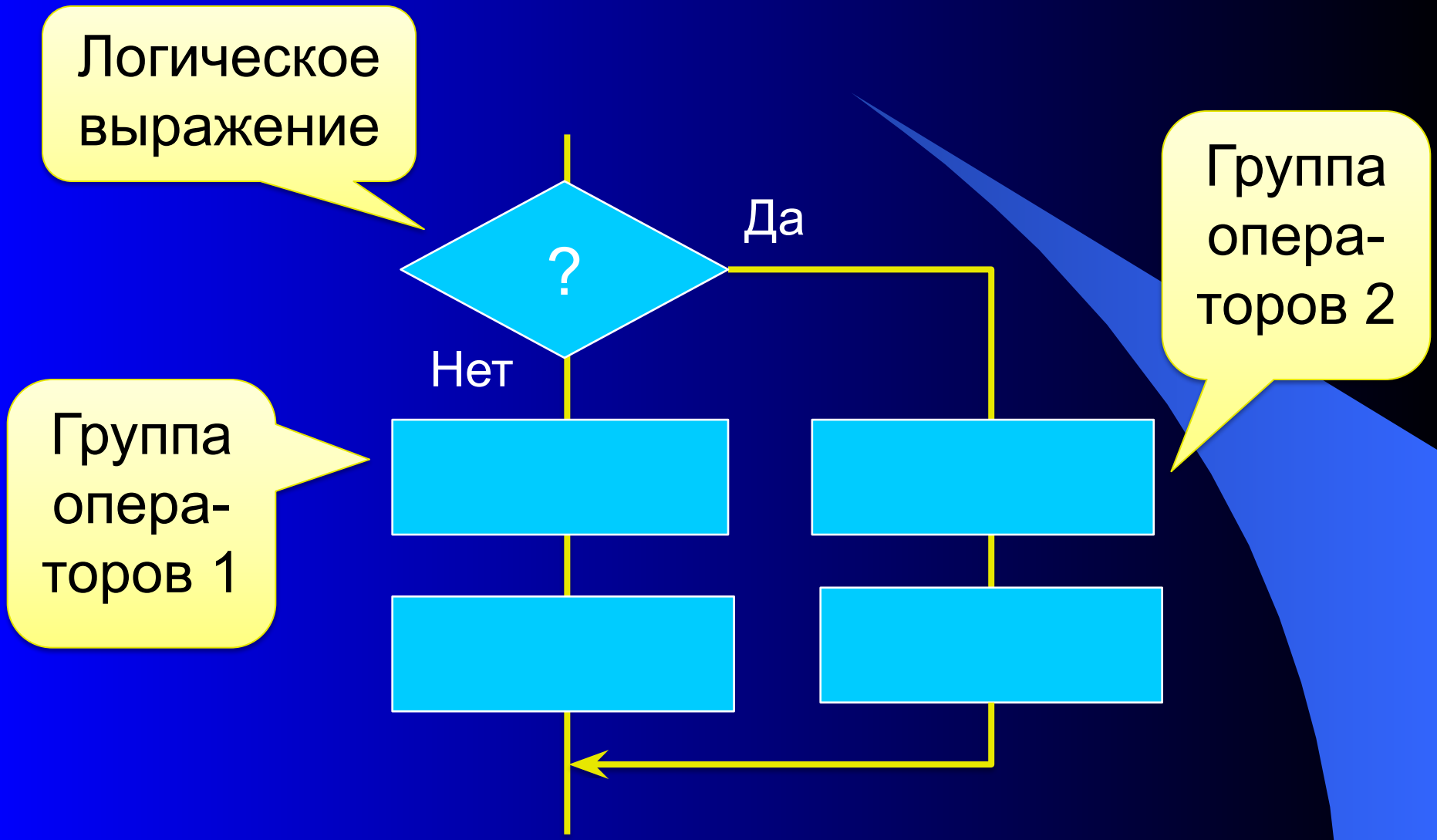
1. Структура СЛЕДОВАНИЕ

- Это структура, действия в которой выполняются последовательно друг за другом от первого до последнего.
- Эта структура была рассмотрена на примере вычисления длины окружности.

2. Структура РАЗВИЛКА

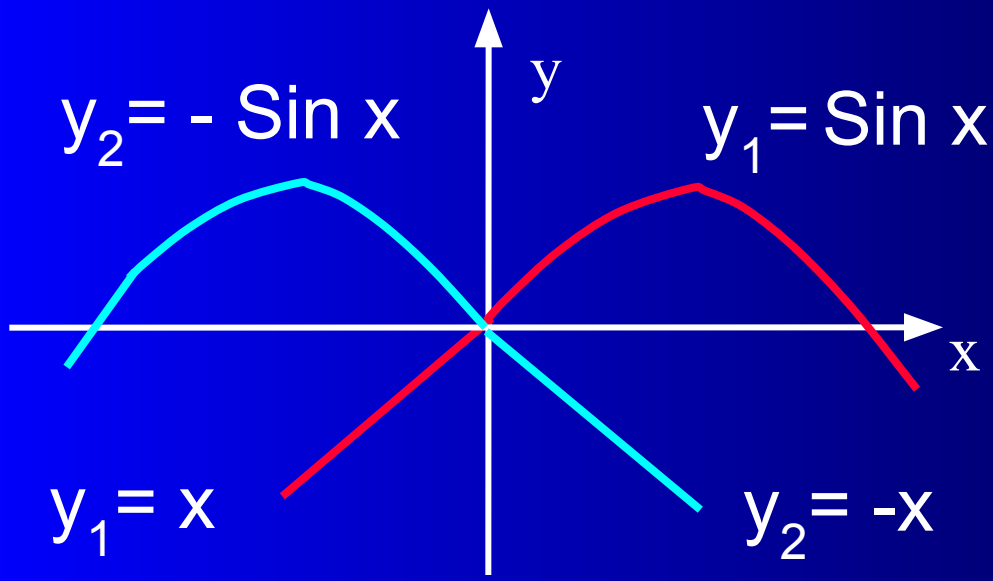
Под развилкой понимается структура, в которой продолжение вычислительного процесса зависит от выполнения или не выполнения некоторого логического выражения (условия).

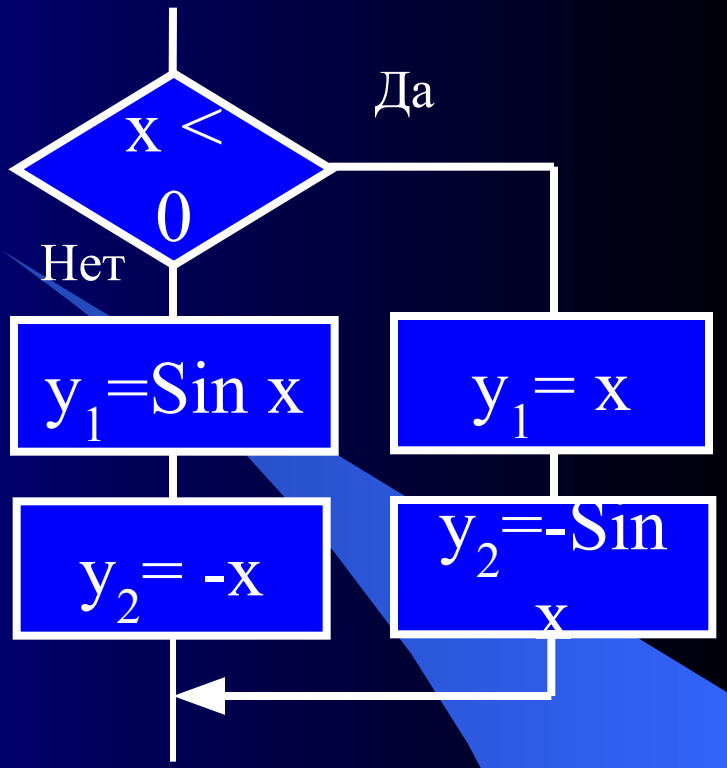
Классическая развилка



Условный оператор If…Then…Else…End If

If <логическое выражение>
Then
<группа операторов 1>
Else
<группа операторов 2>
EndIf





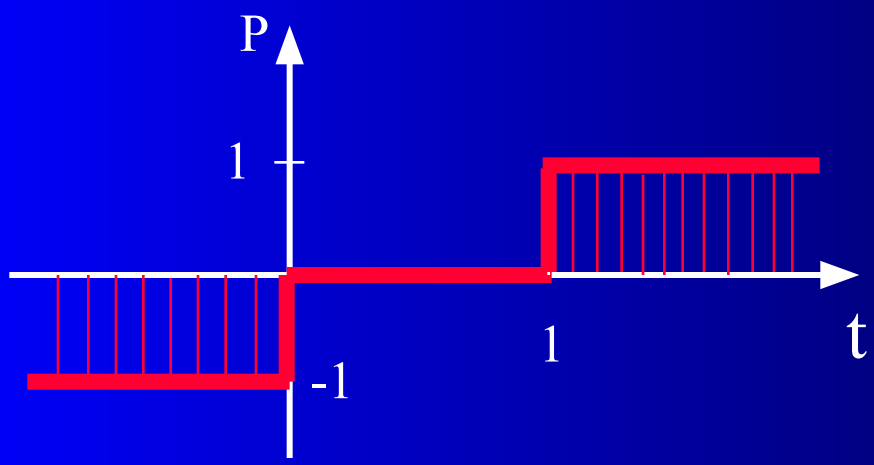
$$\begin{cases} y_1 = x, & y_2 = -\text{Sin } x, \text{ если } x<0 \\ y_1 = \text{Sin } x, & y_2 = -x, \quad \text{в ост. случаях} \end{cases}$$

```
Private Sub Command1_Click()
Dim Y1 As Single, Y2 As Single, X as Single
X=Inputbox("Введите X", "Развилка")
   If x<0 Then
        y1 = x
        y2 = -Sin(x)
         Else
        y1 =Sin(x)
        y2 = -x
   End If
Print "При X=" & X & " Y1=" & Y1 & " Y2=" & Y2
End Sub
```

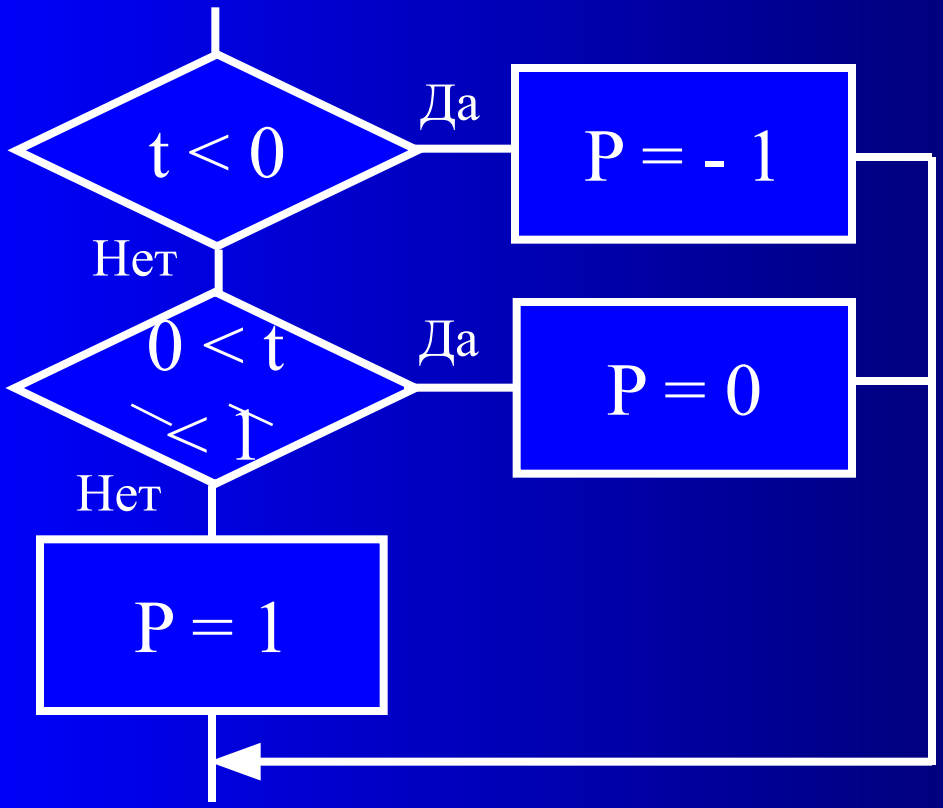
Вложенная развилка

Условный оператор **If…ElseIf…End If**

If <логическое выражение 1> Then
<группа операторов 1>
ElseIf <логическое выражение 2> Then
<группа операторов 2>
Else
<группа операторов 3>
End If



$$P = \begin{cases} -1, & \text{если } t < 0 \\ 0, & \text{если } 0 \le t \le 1 \\ 1, & \text{если } t > 1 \end{cases}$$




```
If t < 0 Then
P = -1
ElseIf t >= 0 And t <= 1
Then
P = 0
Else
P = 1
End If
```

3. Структура цикл

- Наиболее эффективно проявляются возможности компьютера при многократном выполнении одних и тех же действий с изменяющимися данными.
- Такой вычислительный процесс называется циклическим, а описывающая его структура – циклом.

Будем использовать следующие термины и обозначения:

- параметр цикла – X,
- начальное значение параметра цикла – Xo,
- конечное значения параметра цикла – Xk,
- шаг изменения параметра цикла – dX,
- условие выполнения цикла – X <= Xk,
- тело цикла – группа повторяющихся операторов.

Различают циклы с параметром и итерационные:

- в циклах с параметром число его повторений (N) заранее известно и зависит от начального значения параметра цикла, его конечного значения и шага

$$N = (Xk – Xo) / dX + 1;$$

- в итерационных циклах повторения заканчиваются когда достигается заданная точность вычислений (нахождение предела функции, корней уравнений и т.п.).

Циклы с параметром различают :

- циклы с предусловием, когда проверка на окончание цикла осуществляется до повторяющихся действий (в этом случае тело цикла может не выполниться ни разу)

и

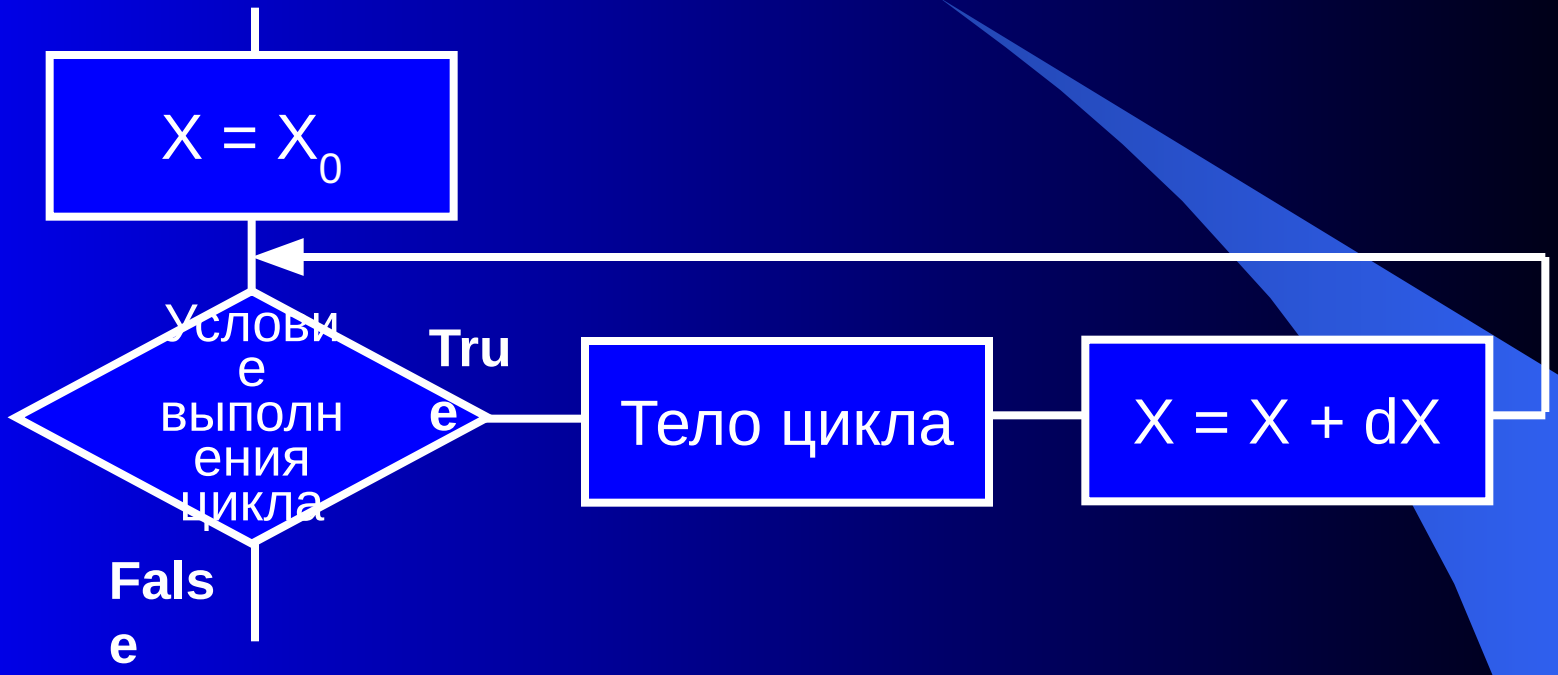
- циклы с постусловием, когда проверка на окончание цикла осуществляется после выполненных действий (в этом случае тело цикла выполнится хотя бы один раз).

Цикл с предусловием

Организуется следующим образом:

1. Параметру цикла X присваивается начальное значение $X_{0.}$
2. Проверяется условие выполнения цикла.
3. Если это условие истинно, то выполняется тело цикла, если ложно, осуществляется переход к оператору следующему за циклом.
4. Значение параметра цикла изменяется на величину шага и далее снова выполняется пункт 2.

Схема алгоритма:



Цикл с предусловием реализуется несколькими способами.

Оператор For . . . Next
(используется только для цикла с предусловием)

For < Параметр цикла > = < Начальное значение >
    To < Конечное значение > [Step < Шаг >]
< Тело цикла >
Next [Параметр цикла]

For X = X0 To Xk Step dX
< Тело цикла >
Next X

Пример 1.

1. Постановка задачи

Вычислить значение функции Y = Sin X при значениях аргумента меняющегося от 0 до 1 с шагом 0,1

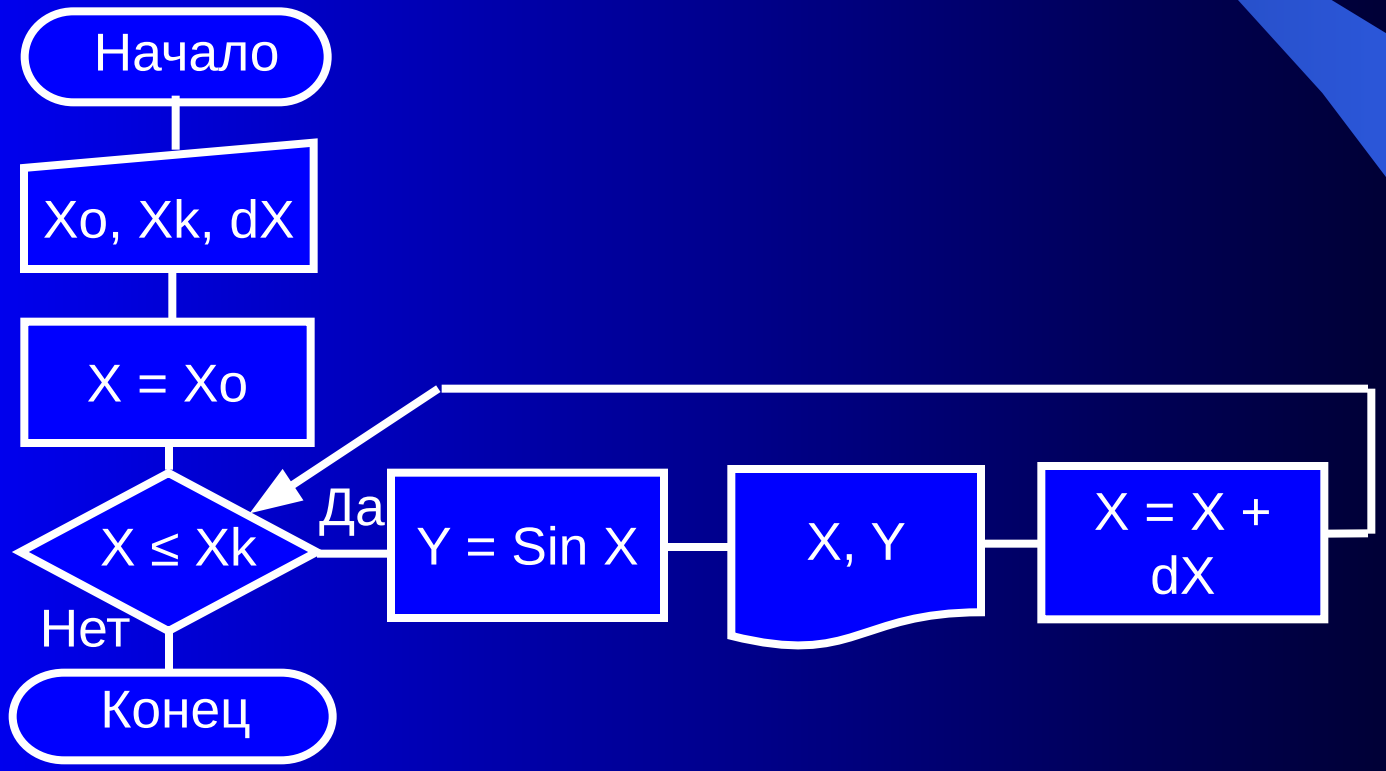
Входные данные: X_0, Xк, dX

Выходные данные: X, Y

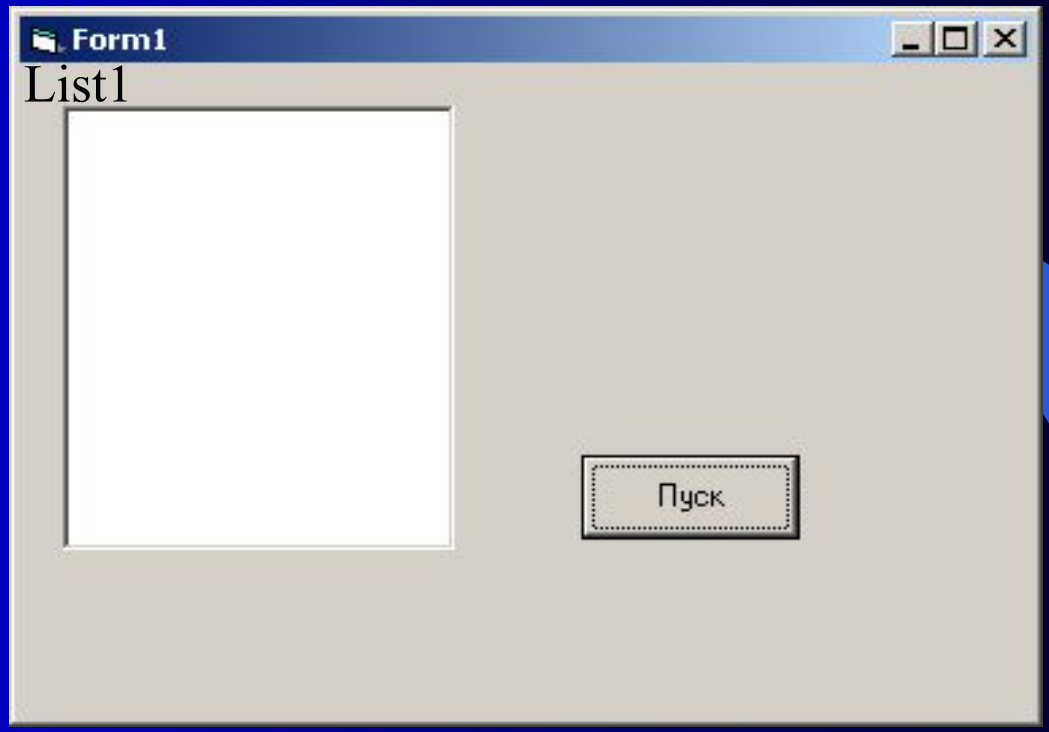
2. Математическая модель

y = Sin x для всех 0 ≤ x ≤ 1 с шагом 0,1

3. Схема алгоритма



4. Разработка визуальной части проекта

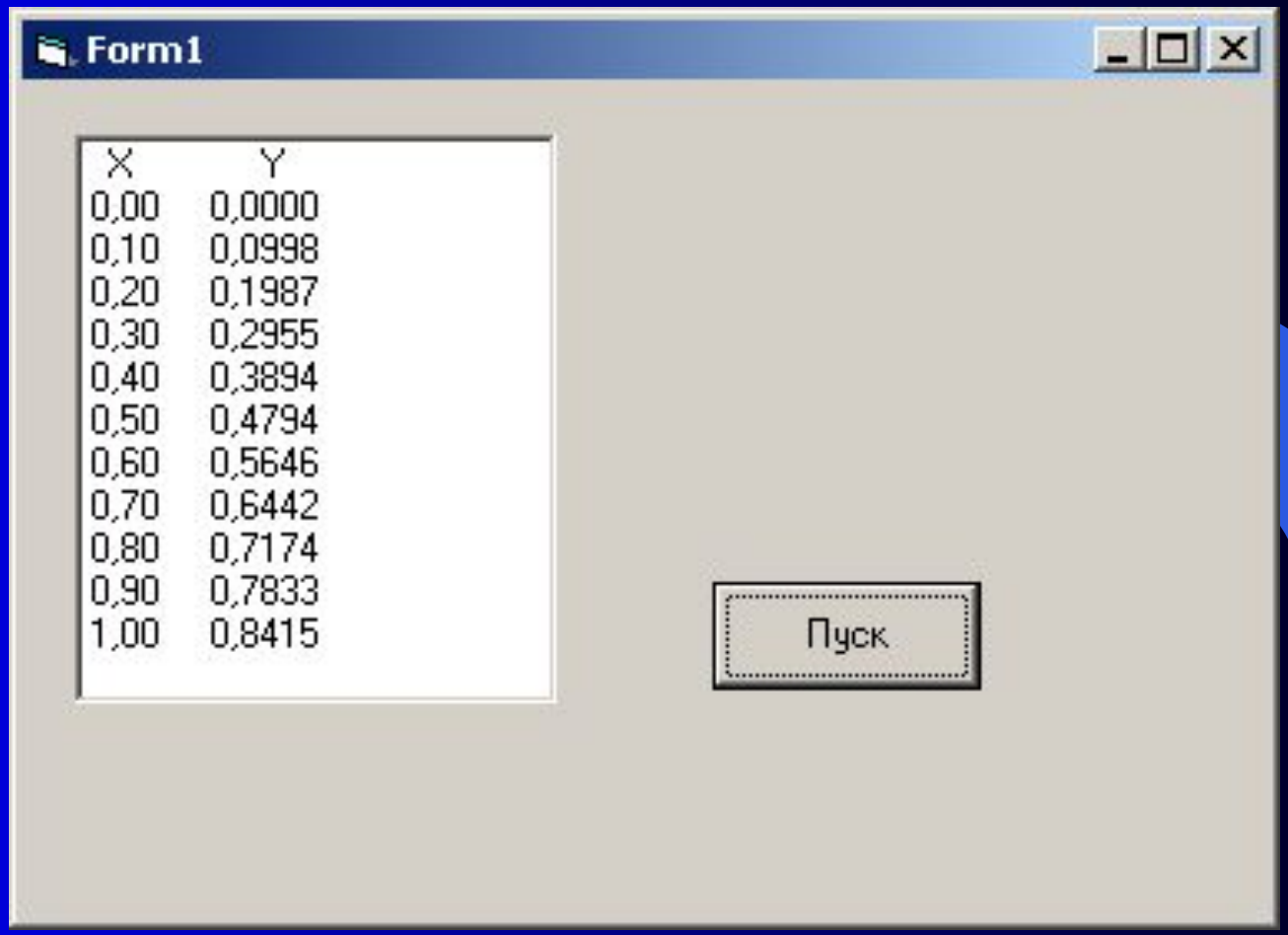


ListBox – поле списка
(см. Основные элементы управления панели General)

5. Код приложения

```
Private Sub Command1_Click()
Dim x0 As Single, xk As Single, dx As Single
Dim y As Single
x0 = InputBox("Введите x0")
xk = InputBox(" Введите xk")
dx = InputBox(" Введите dx")
List1.AddItem "  X              Y "
For x = x0 To xk + dx / 2 Step dx
y = Sin(x)
List1.AddItem Format(x, "0.00") & "      " & Format(y, "0.0000")
Next
End Sub
```

6. Отладка программы



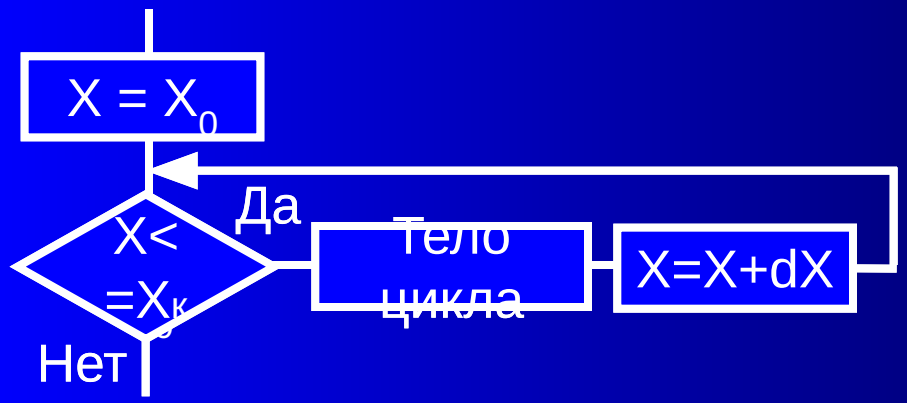
Цикл с постусловием

Организуется следующим образом:

1. Задается начальное значение параметра цикла $X=X_0$
2. Выполняется тело цикла.
3. Значение параметра цикла изменяется на величину шага.
4. Проверяется условие продолжения цикла.
5. Если условие истинно, то вновь выполняется тело цикла (переход к пункту 2),
если же условие ложно, то выполняется следующий после цикла оператор.

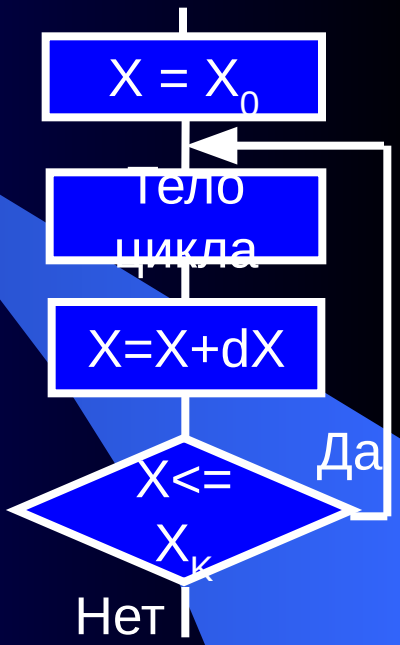
Оператор Do While . . . Loop

С предусловием:




```
X = X0
Do While X <= Xk
< Тело цикла >
X = X + dX
Loop
```

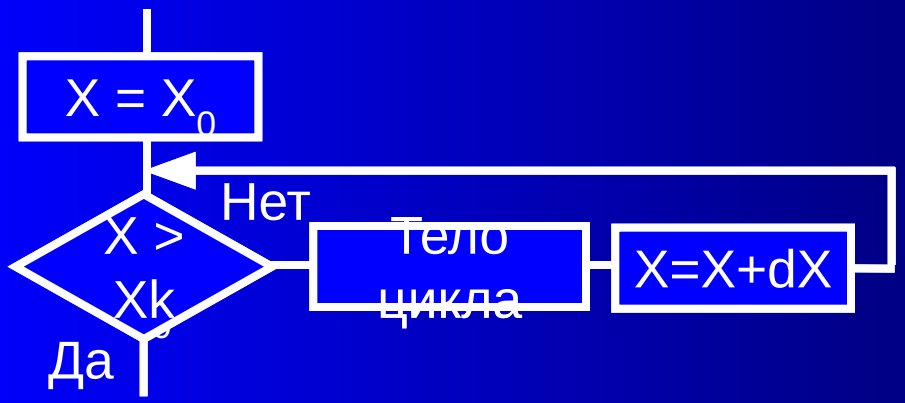
С постусловием:




```
X = X0
Do
< Тело цикла >
X = X + dX
Loop While X <= Xk
```

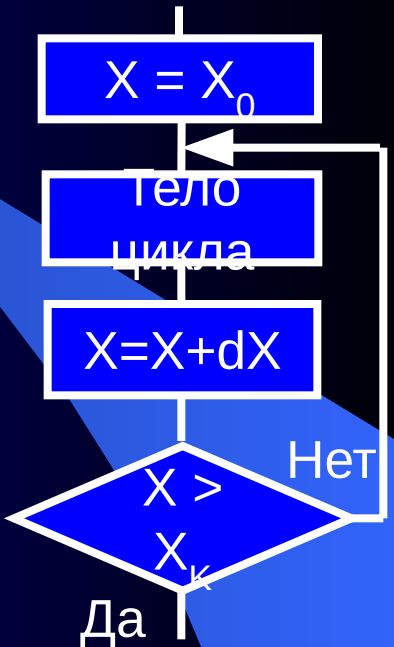
Оператор Do Until . . . Loop

С предусловием:



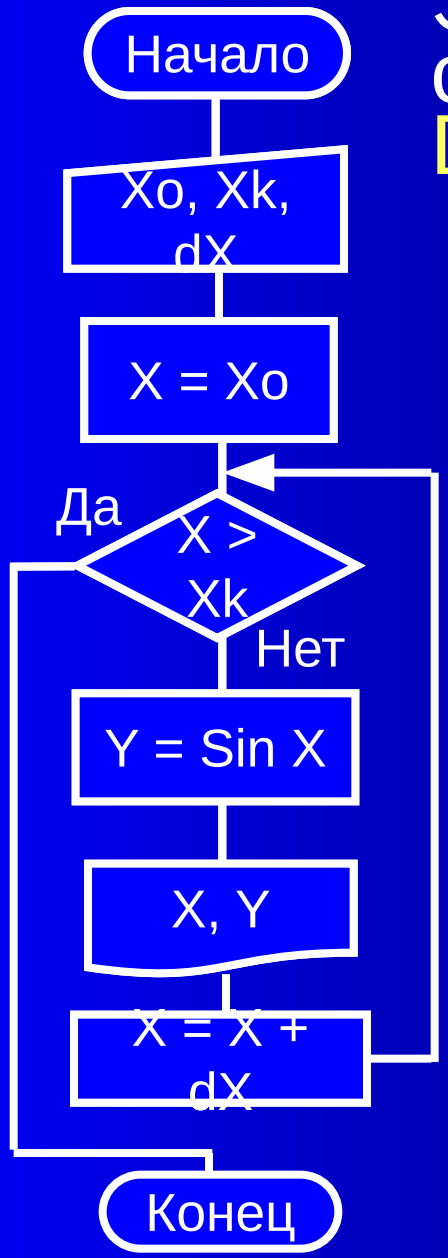
X = X0
Do Until X > Xk
< Тело цикла >
X = X + dX
Loop

С постусловием:



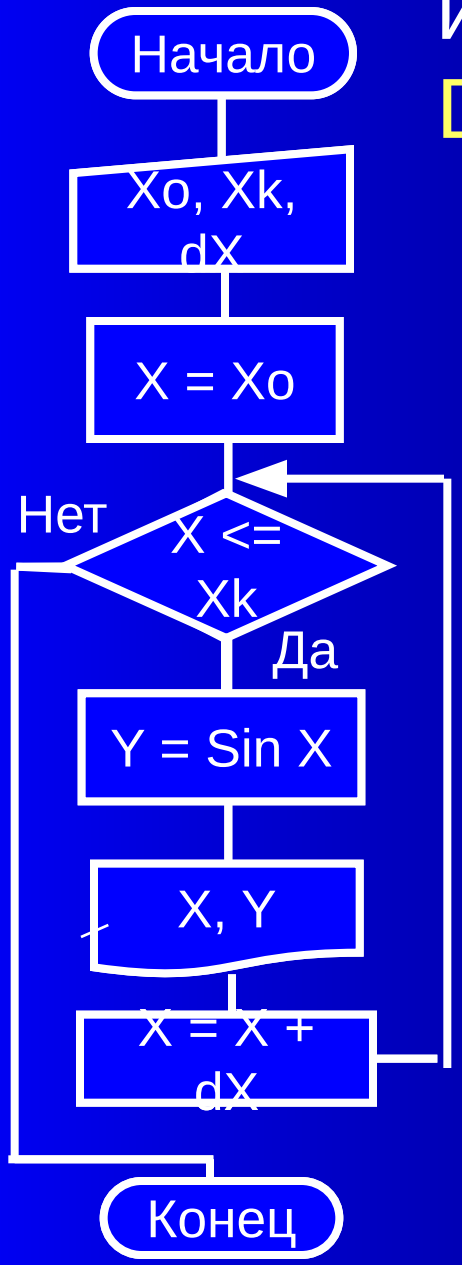
X = X0
Do
< Тело цикла >
X = X + dX
Loop Until X > Xk

Эту же задачу можно реализовать с помощью оператора Do Until … Loop (с предусловием)



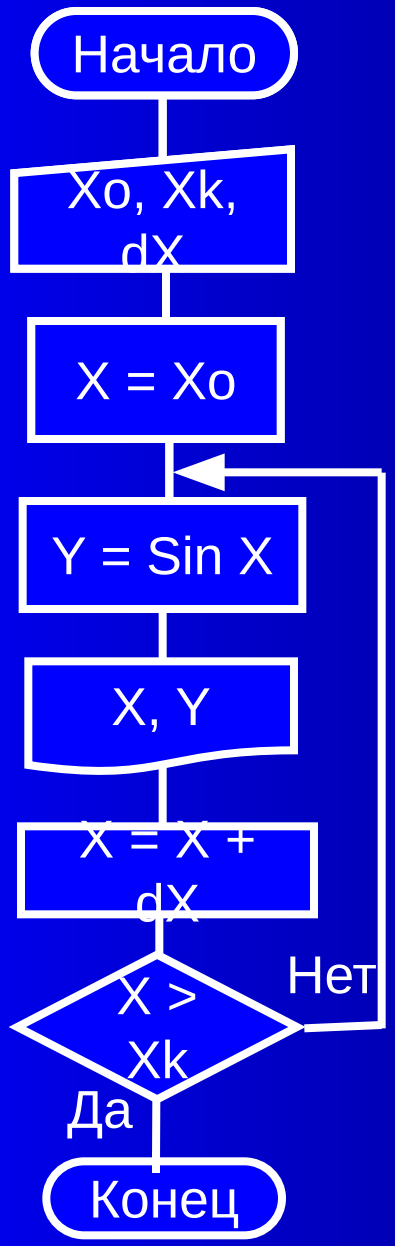

```
. . .
X = X0
Do until X > Xk
Y = Sin (X)
List1.AddItem …
X = X + dX
Loop
. . .
```

или с помощью оператора Do While…Loop (с предусловием)

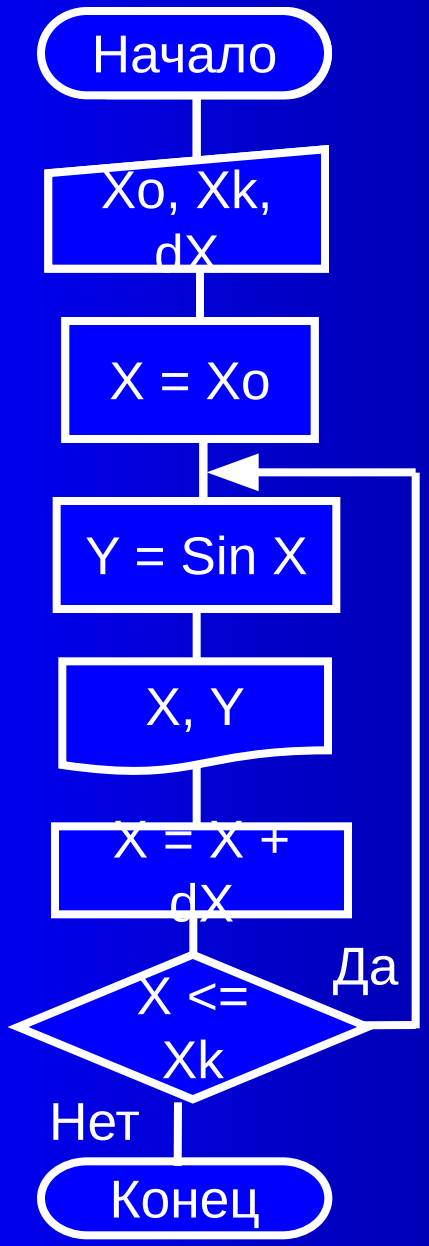



```
. . .
X = X0
Do While X <= Xk+dX/2
Y = Sin (X)
List1.AddItem …
X = X + dX
Loop
. . .
```

А теперь с помощью оператора Do…Loop Until (с постусловием)




```
. . .
X = X0
Do
Y = Sin (X)
List1.AddItem …
X = X + dX
Loop until X > Xk
. . .
```



или с помощью оператора Do…Loop While (с постусловием)

```
. . .
X = X0
Do
Y = Sin (X)
List1.AddItem …
X = X + dX
Loop while X <= Xk+dX/2
. . .
```

Структура «Цикл в цикле»

- это структура с одним или несколькими вложенными циклами.

Рассмотрим пример:

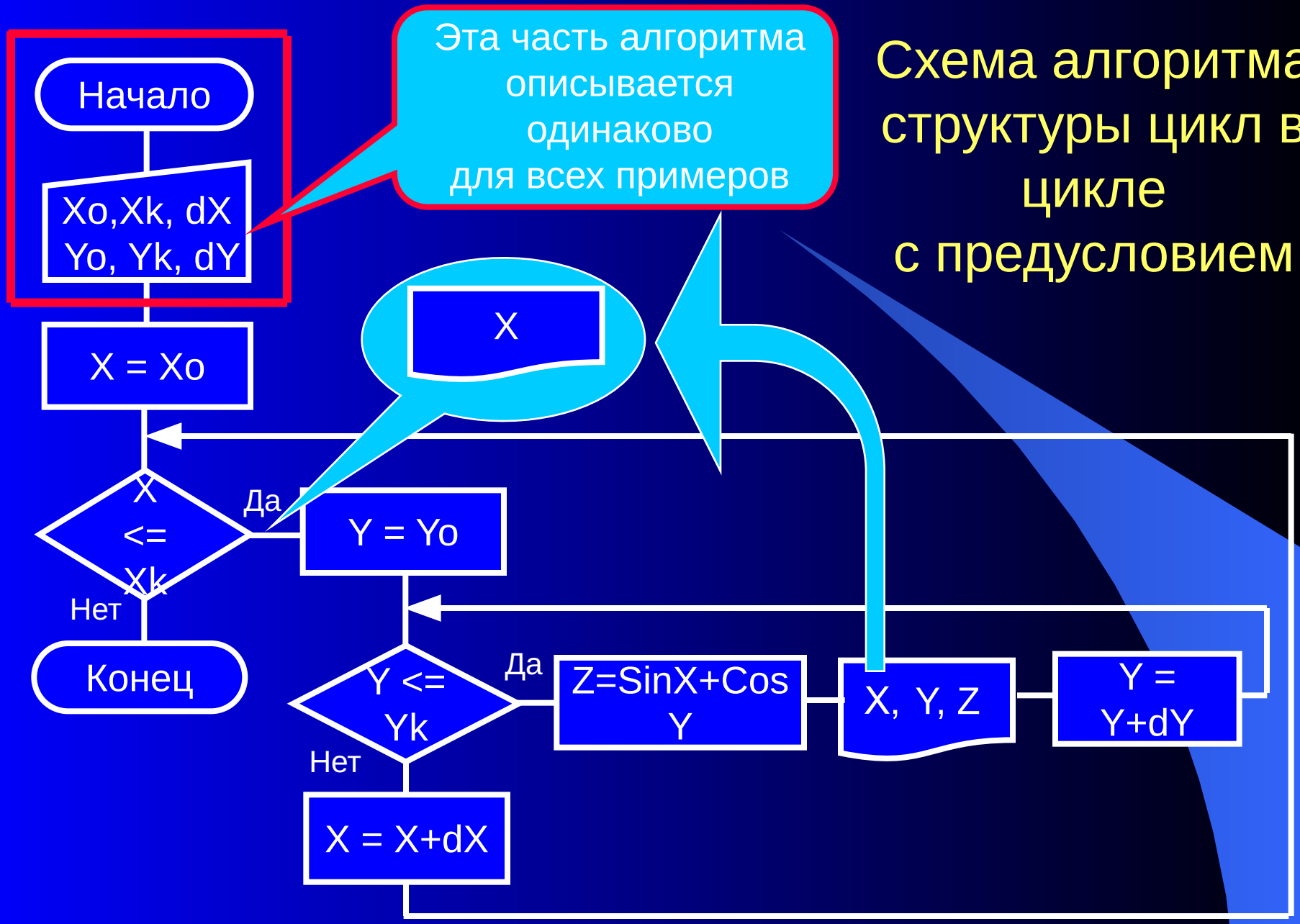
$$Z = \sin X + \cos Y ,$$

где

$Xo <= X <= Xk$ с шагом dX

$Yo <= Y <= Yk$ с шагом dY

Схема алгоритма структуры цикл в цикле с предусловием



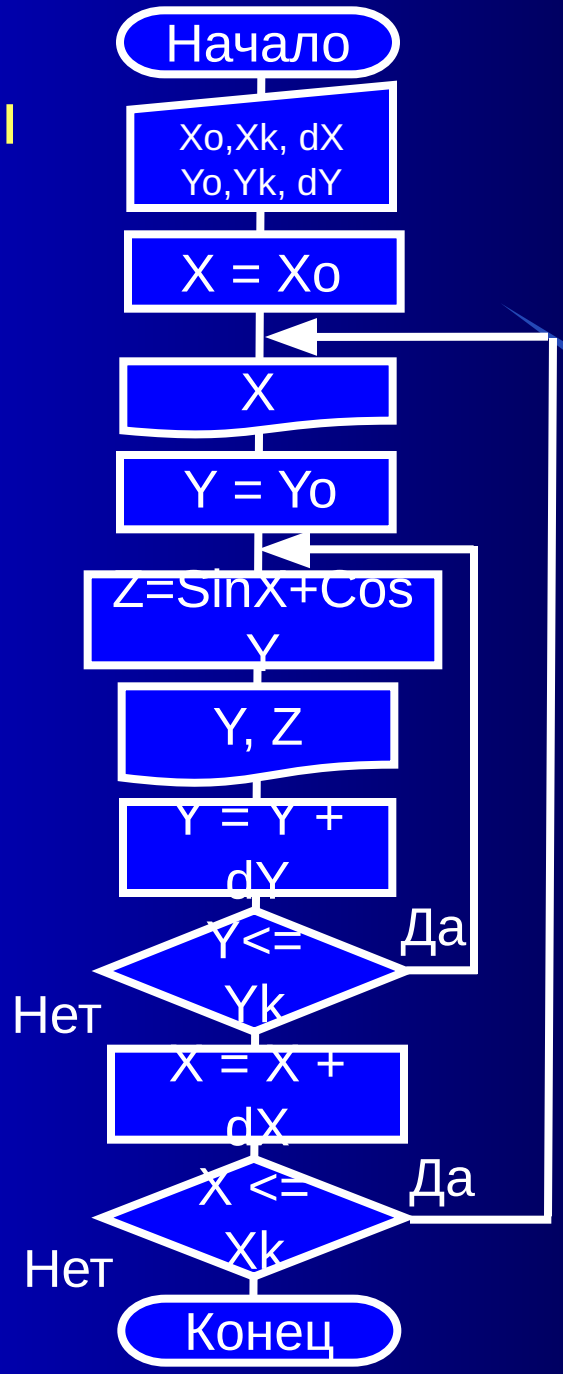
```
Private Sub Command1_Click()
Dim X0 As Single, Xk As Single, dX As Single
Dim Y0 As Single, Yk As Single, dY As Single
Dim Z As Single
X0 = InputBox (“Введите начальное значение X”)
Xk = InputBox (“Введите конечное значение X”)
dX = InputBox (“Введите шаг изменения X”)
Y0 = InputBox (“Введите начальное значение Y”)
Yk = InputBox (“Введите конечное значение Y”)
dY = InputBox (“Введите шаг изменения Y”)
. . .
```

Варианты продолжения кода процедуры

```
. . .
X = X0
Do While X <= Xk
Print  X
Y = Y0
Do While Y <= Yk
Z = Sin (X) + Cos (Y)
Print Y, Z
Y = Y + dY
Loop
X = X + dX
Loop
End Sub
```

```
. . .
For X=X0 To Xk Step dX
Print X
For Y=Y0 To Yk Step dY
Z = Sin (X) + Cos (Y)
Print  Y,  Z
Next Y
Next X
End Sub
```

Схема структуры цикл в цикле с постусловием



На следующем слайде представлен фрагмент кода процедуры этой части алгоритма

```
. . .
X = Xo
Do
Print “X=” & X
Y = Yo
Do
Z = Sin (X) + Cos (Y)
Print “Y = ” & Y & “   Z = ” & Z
Y = Y + dY
Loop While Y <= Yk
X = X + dX
Loop While X <= Xk
End Sub
```

Фрагмент кода процедуры без описаний переменных и ввода значений входных переменных