Информатика

Лекция 7

Основы построения ЭВМ

Принципы фон Неймана

1. Информация кодируется в двоичном формате и разделяется на единицы (элементы) информации – ***слова***. ***Слово*** обрабатывается как единое целое (машинный элемент информации).
2. Разнотипные слова хранятся в одной и той же ***памяти*** и различаются только **по способу их использования** (числа, команды и т.д.). Все ***слова*** по своей сути одинаковы и неразличимы. Такое «**однообразие**» слов позволяет использовать ***одни и те же операции для обработки слов различной природы***.
3. ***Слова*** информации размещаются в ***ячейках памяти*** машины и идентифицируются ***номерами ячеек*** (***адресами***). Основная память состоит из пронумерованных ячеек. ***Адрес*** ячейки используется для ***чтения*** или ***записи*** слов.

Принципы фон Неймана

4. Алгоритм представляется в виде последовательности управляющих слов (***команд***), которые определяют наименование операции и слова информации, участвующие в этой операции.

Алгоритм, представленный в терминах машинных команд, называется ***программой***.

Четырехадресная команда:

КОП – это Код Операции

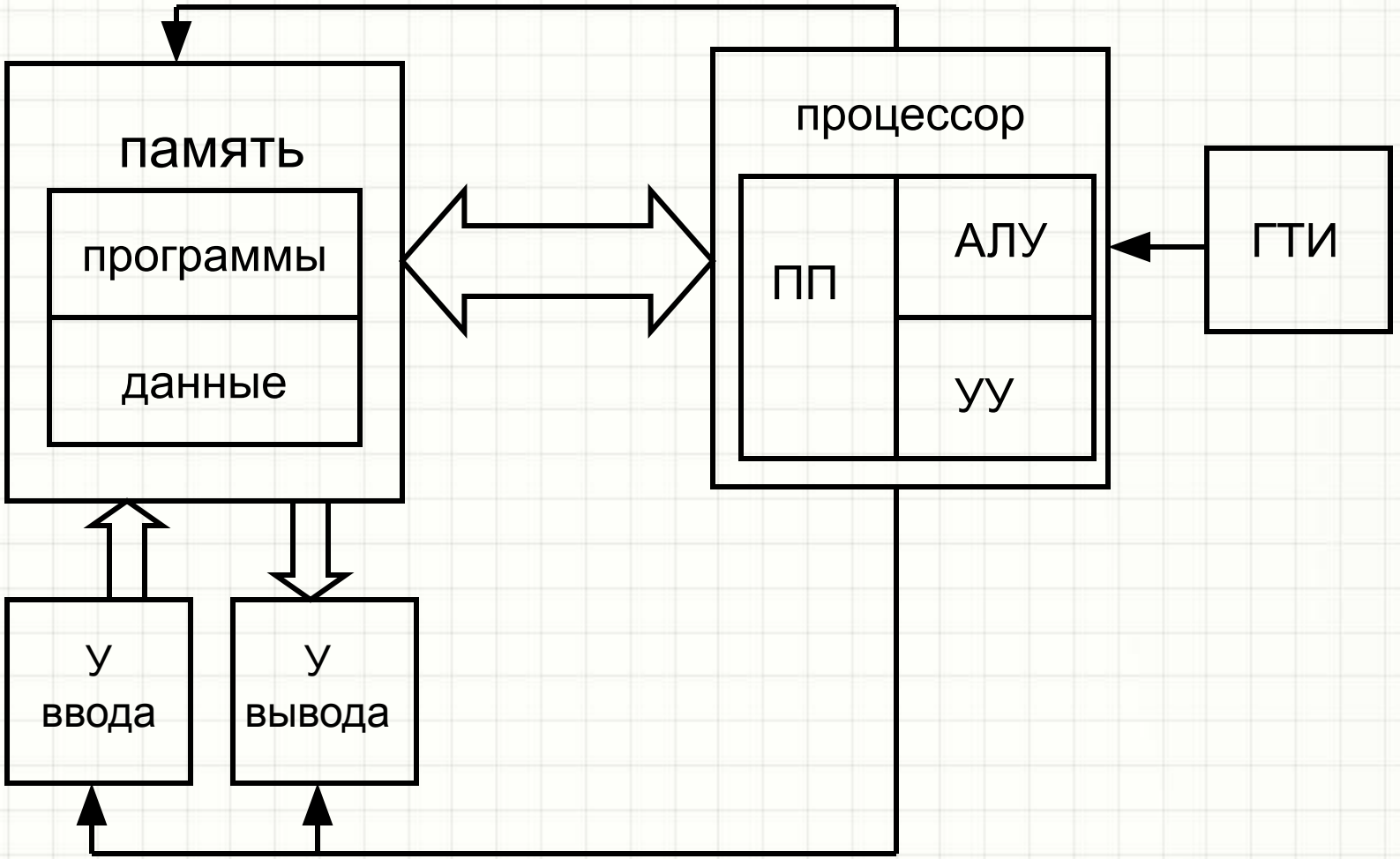
5. Выполнение вычислений, предписанных алгоритмом, сводится к последовательному выполнению команд в порядке, однозначно определяемом программой.

Обобщенная структура ЭВМ

Пять принципов фон Неймана предполагают, что основными составными частями ЭВМ должны быть:

- **АЛУ** – арифметико-логическое устройство – для выполнения арифметических и логических операций
- **УУ** – устройство управления – для организации выполнения программы
- **ЗУ** – запоминающее устройство – память
- **ВУ** – внешнее устройство – устройство ввода-вывода

Обобщенная структурная схема ЭВМ



Архитектура и структура ЭВМ

Архитектура ЭВМ – это ***логическая организация*** вычислительного устройства, состав и назначение ее ***функциональных*** средств, ***принципы кодирования*** и т.п., т.е. все, что определяет процесс обработки информации.

Архитектура, построенная на принципах фон Неймана – ***классическая архитектура***.

Структура ЭВМ – это ***совокупность элементов*** компьютера ***и связей*** между ними.

Структура ЭВМ – это «железо», которое не будет «работать» без программ, задающих алгоритм обработки информации.

Необходимость программного обеспечения (ПО).

Программное обеспечение

Программное обеспечение ЭВМ

Программное обеспечение (ПО) делится на две группы:

- ***Системное*** ПО (**СПО**)
- ***Прикладное*** ПО (**ППО**)

Системное ПО можно разделить на:

- ***Базовое*** ПО (BIOS, OS, операционные оболочки)
- ***Сервисное*** ПО (диагностическое, обслуживающее, антивирусное ….)

Прикладное ПО (ППО):

- Пакеты прикладных программ

Подробности рассмотреть самостоятельно.

Языки программирования

Языки программирования

Команда в памяти ЭВМ ***записана*** в виде ***машинного слова***

Для записи программ используются языки программирования

Язык программирования (ЯП) – формализованный язык для описания алгоритмов решения задачи на вычислительной машине

ЯП можно разделить:

- ЯП низкого уровня (машинные коды, машинно-ориентированные – ассемблеры)
- ЯП высокого уровня - алгоритмические (Pascal, Basic, СИ)

Языки программирования

Для перевода программ с языка высокого уровня в машинные коды используются специальные программы – ***трансляторы***.

Трансляторы бывают двух типов:

- ***Компилляторы*** - преобразуют всю программу целиком в исполняемый файл в машинных кодах; недостаток – сложности при отладке
- ***Интерпретаторы*** – преобразуют программу построчно и сразу выполняют; недостаток – более медленное исполнение

Объектно-ориентированные языки программирования (Visual Basic, Delphi, C++) – относятся к языкам высокого уровня, имеют свои особенности, используются для создания приложений.

Языки программирования

Описание ЯП

- Используемые символы – ***алфавит***
- Правила записи слов – ***синтаксис***
- ***Ограничения*** на использование слов

Понятие ***переменной*** – задается именем (словом), изменяет свое значение по ходу выполнения программы

Свойства переменных определяются их ***типом***:

- Числа
- Строки
- Символы
- …

С разным типом переменных можно производить различные действия

Языки программирования

Типы переменных, занимаемый в памяти объем и операции с переменными – рассмотреть самостоятельно.

Пример.

Если переменные ***A*** и ***B*** – числа, то

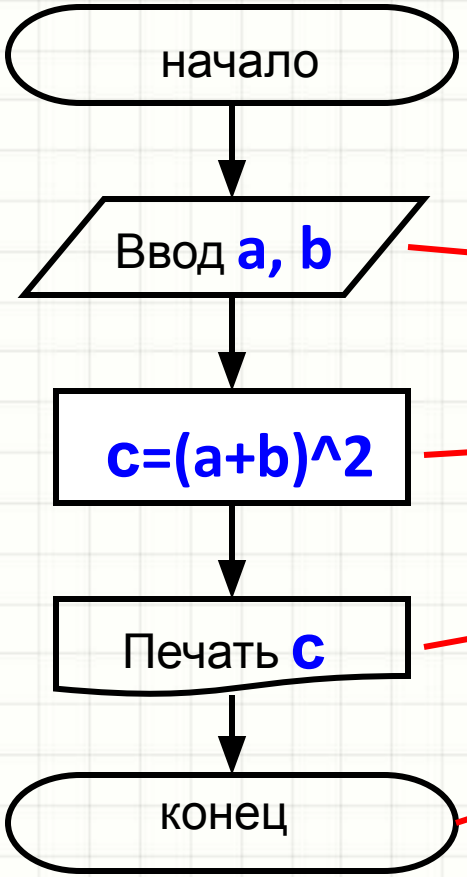
при ***A***=3 ***B***=4 ***S*** = ***A*** + ***B*** = **7**

Если переменные ***A*** и ***B*** – текстовые, то

При ***A***=3 ***B***=4 ***S*** = ***A*** + ***B*** = **34**

Пример блок-схемы и программы

Блок-схема



Программа

```
Dim a, b, c As
Integer
Input a, b
c=(a+b)^2
Print c
End
```

Операторы BASIC

Оператор объявления типа записывается в начале программы или процедуры в разделе объявлений (***Declarations***).

Синтаксис записи:

Dim Имя_переменной1 [***As тип1***][,]…

После слова ***Dim*** через запятую можно записывать несколько таких конструкций:

Dim **X** ***As Single***, **ЧислоЭлементов** ***As Integer***,
Пи ***As Double***, **S** ***As String***

После ***String*** может стоять знак * и указано число символов в строке (длина строки).

Переменные и действия с переменными

Задание переменных :

Dim **X** ***As*** **Intetger**

Dim **E** ***As*** **String**

Dim **A** ***As*** **Single**

Операции с переменными :

- математические

“ + ”, “ - ”, “ * ”, “ / ”, “ ^ ”, “ () ”

- логические

“AND”, “OR”, “NOT”

- операции отношений

“ = ”, “ > ”, “ < ”, “ >=”, “ <=”, “≠ ”

Операторы BASIC

Оператор присваивания:

ИмяПеременной = ***ЗначениеПеременной***

Совокупность данных одного типа, расположенных в памяти последовательно, может образовывать ***массив***.

Массив обозначается именем с указанием размерности и типа данных:

Dim ИмяМассива(*размерность1, размерность2, …*) **As *тип***.

Dim **B(1, 9) *As Integer***.

Обращение к элементу массива осуществляется указанием его **имени** и **индекса**. ***По умолчанию*** индекс массива начинается ***с нуля***. Для явного указания границ массива применяется служебное слово ***To***:

Dim **B(1*To* 2, 1*To* 10) *As Integer***

задан двумерный массив 2*10 (две строки, десять столбцов) с данными целого типа

Переменные и действия с переменными

Примеры задания переменных :

Dim **X** ***As*** **Intetger**

X = 24 – «*переменной X присваивается значение 24*»

A = X – «*переменной A присваивается значение переменной X*»

Массивы

Dim **A(30)** ***As*** ***Integer*** - одномерный массив

Dim **A(30, 30)** ***As*** ***Integer*** - двумерный массив

Константы

Const **Pi** ***As*** ***Single*** = 3.1428

В процессе вычислений константа изменяться не может

Операторы условия

1. Однострочная форма :

If *УсловноеВыражение* ***Then*** *Оператор1* [***Else*** *Оператор2*]

2. Многострочная форма :

If *УсловноеВыражение* ***Then***

ПоследовательностьОператоров1

[***Else***

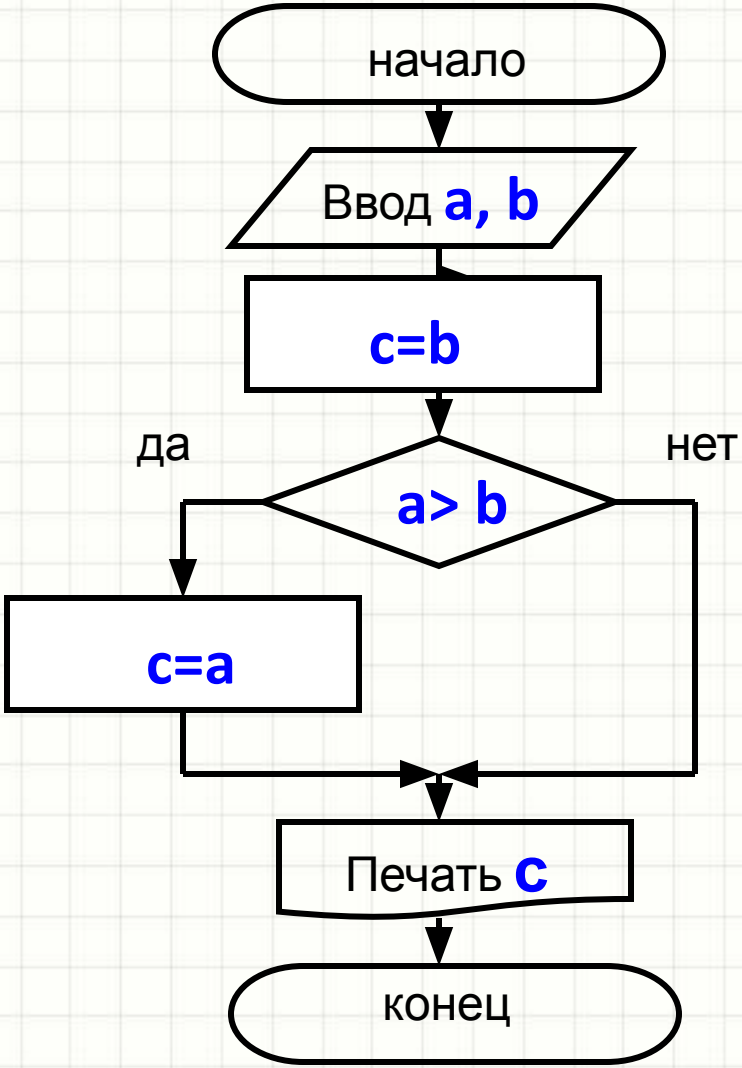
ПоследовательностьОператоров2]

End If

Примечание: операторы, заключенные в квадратные скобки, могут отсутствовать (не обязательны).

Операторы условия

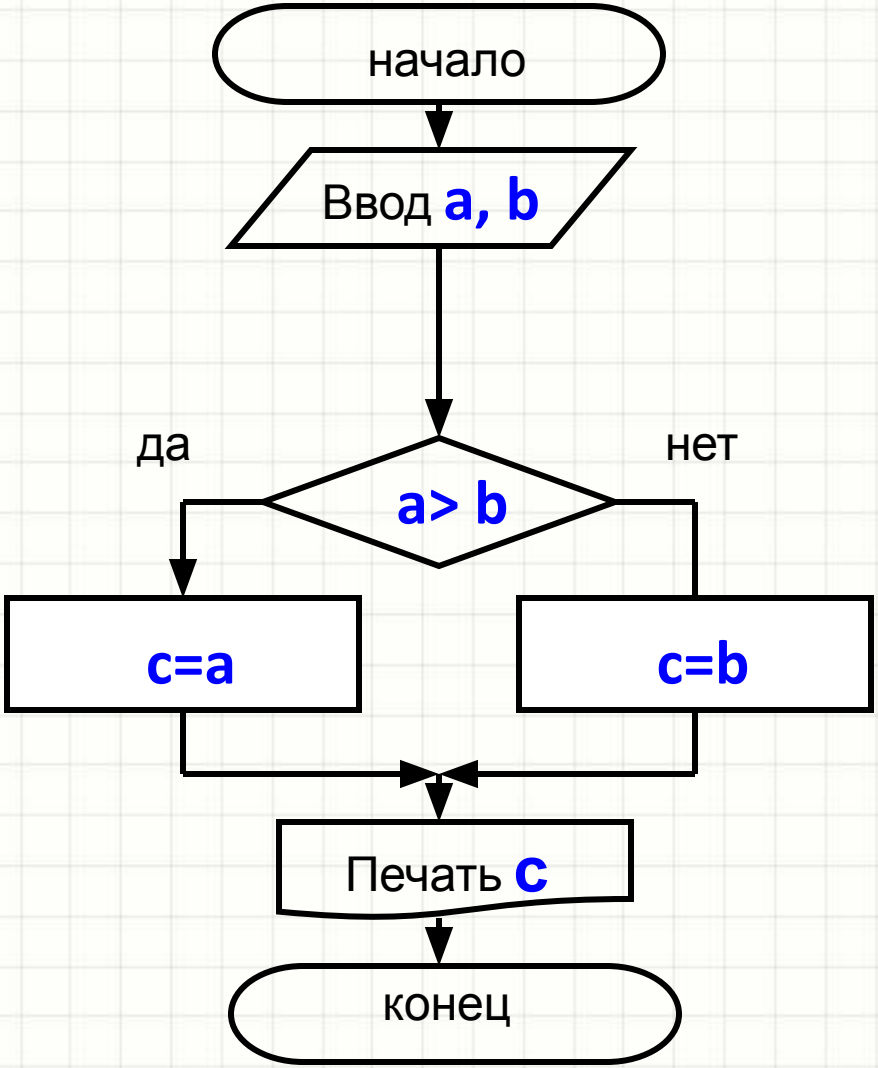
Использование ***однострочного*** оператора **If**




```
Dim  a, b, c As
     Integer
Input  a, b
c=b
If  a>b Then  c=a
Print  c
End
```

Операторы условия (примеры)

Использование ***многострочного*** оператора **If**



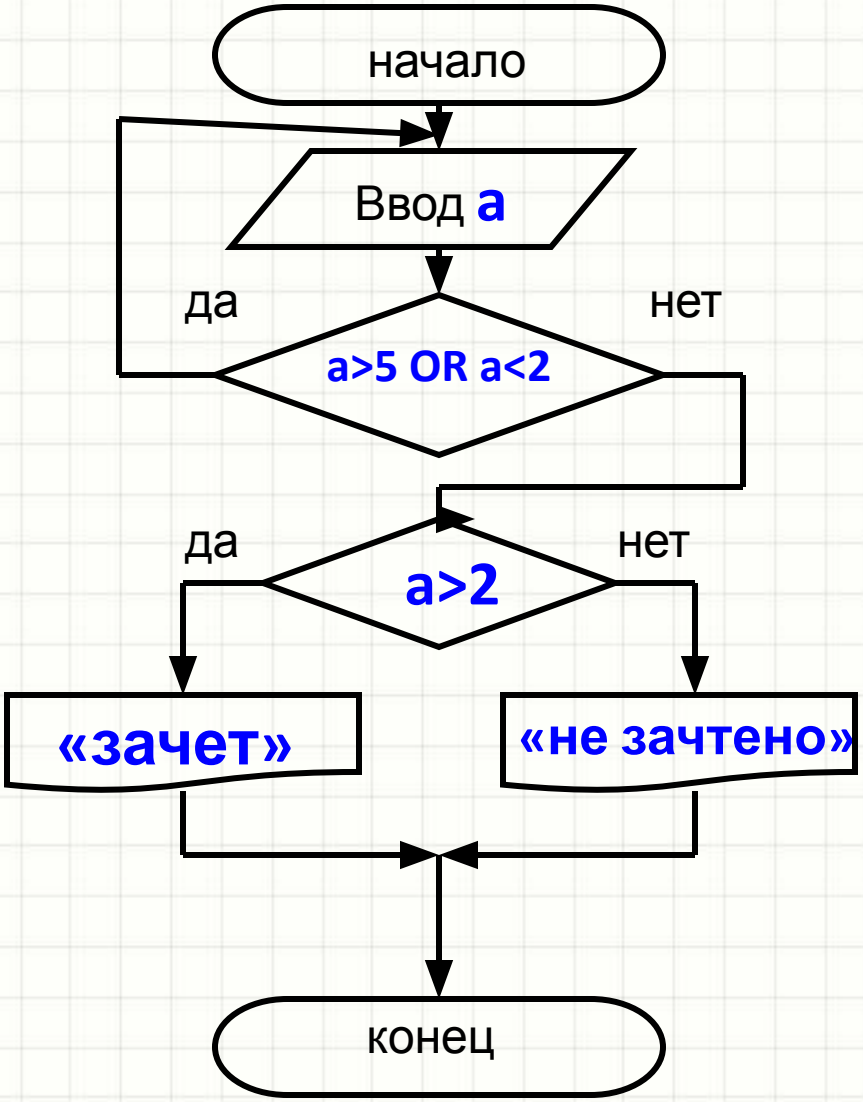

```
Dim  a, b, c As  Integer
Input  a, b
If  a>b Then
   c=a
Else
   c=b
End If
Print  c
End
```

Операторы условия

3. С вложенными операторами :

```
If УсловноеВыражение1 Then
Else
    ПоследовательностьОператоров1
    If УсловноеВыражение2 Then
        ПоследовательностьОператоров2
    End If
End If
```

Операторы условия (примеры)




```
Dim  a As  Integer
 5 Input  a
If  a>5 OR a<2 Then
   GOTO 5
Else
   If  a>2 Then
      Print  «зачет»
   Else
      Print  «не зачтено»
   End If
End If
End
```

GOTO 5 – оператор безусловного перехода (к метке 5)

Операторы условия

4. Использование оператора ***ElseIf*** позволяет использовать другую запись подобной конструкции в программе, объединяя два операторных слова ***Else*** и ***If*** в одно и обойтись одним оператором ***End If*** вместо двух:

If *УсловноеВыражение1* ***Then***

ПоследовательностьОператоров1

ElseIf *УсловноеВыражение2* ***Then***

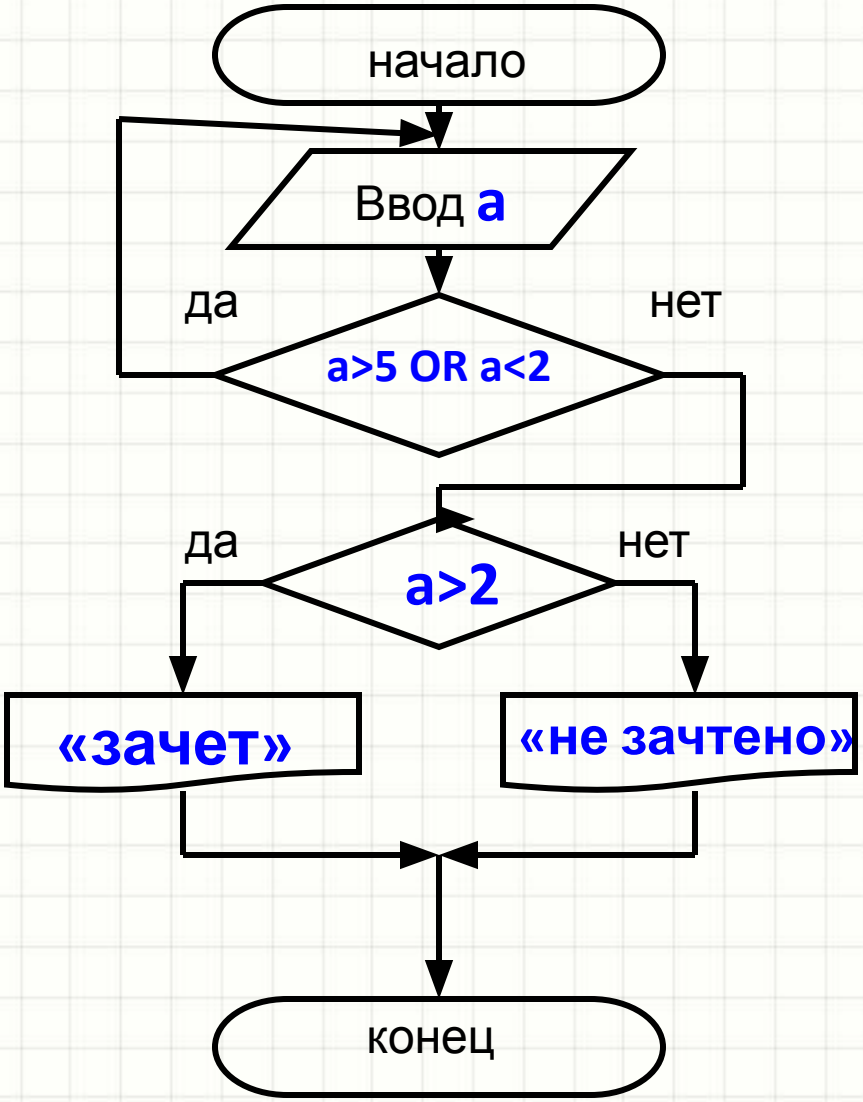
ПоследовательностьОператоров2

Else

ПоследовательностьОператоров3

End If

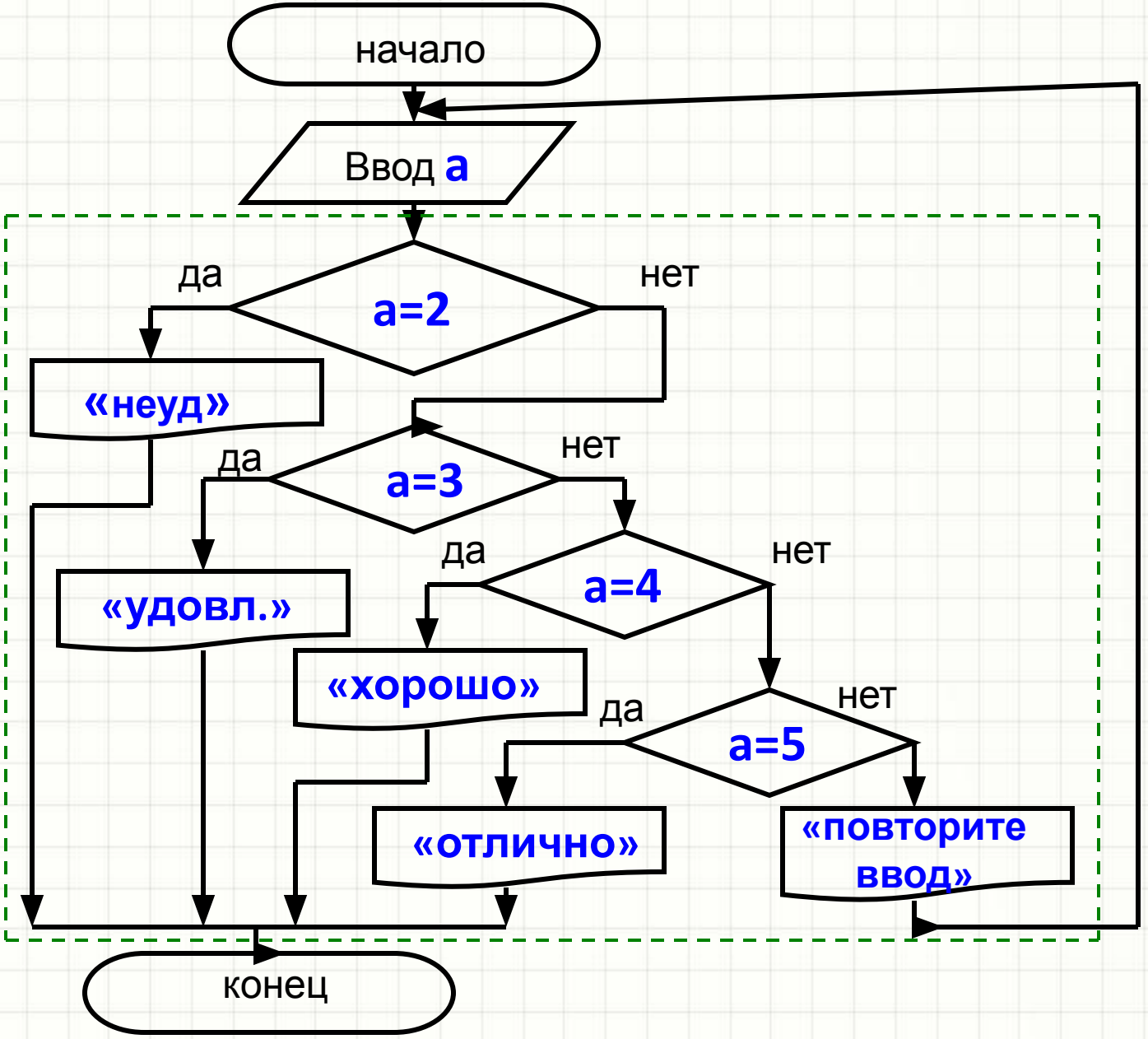
Операторы условия (примеры)




```
Dim  a As  Integer
 5 Input  a
If  a>5 OR a<2Then
   GOTO 5
   ElseIf  a>2 Then
   Print  «зачет»
   Else
   Print  «не зачтено»
End If
End
```

GOTO 5 – оператор безусловного перехода (к метке 5)

Операторы условия (примеры)

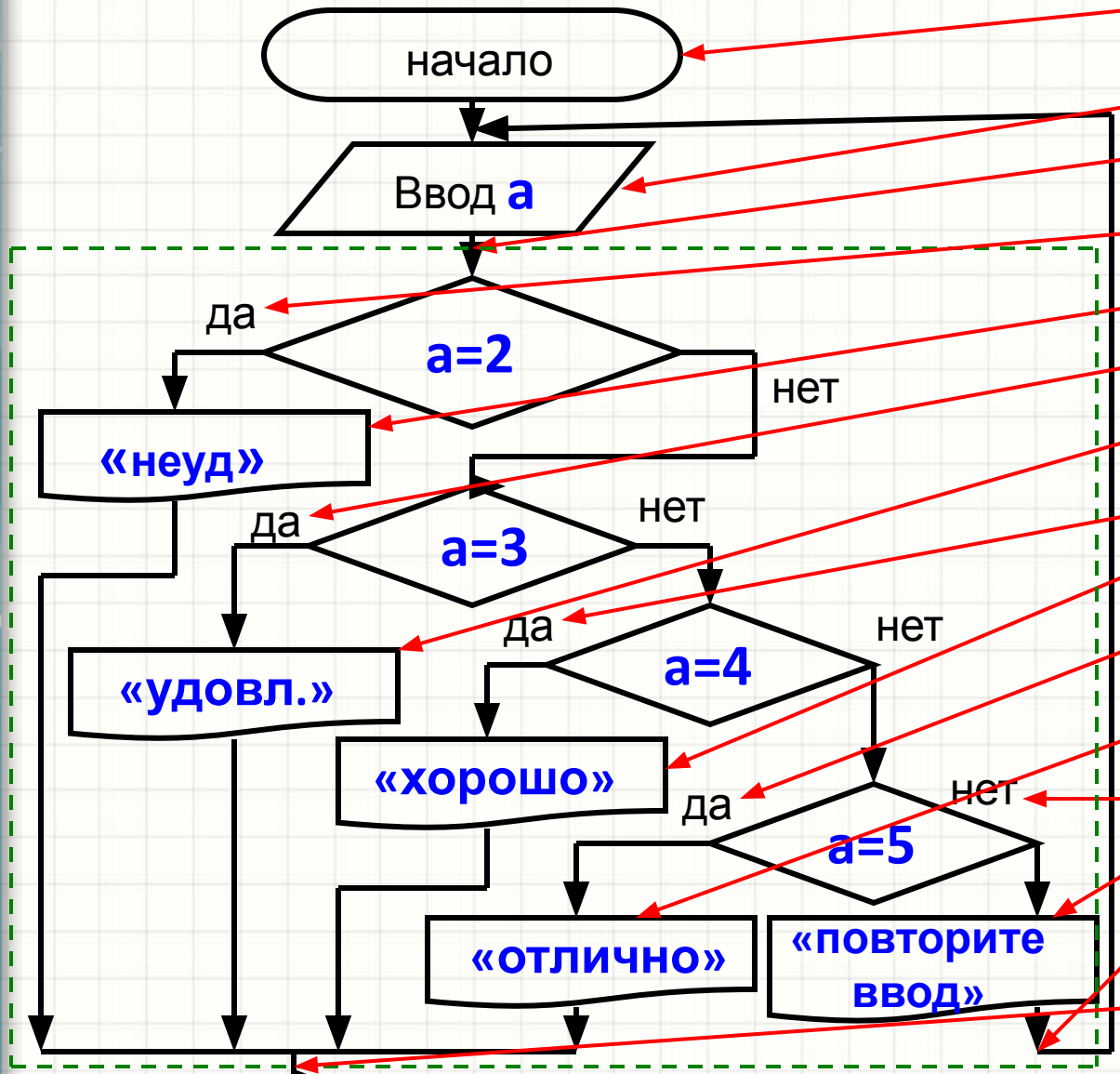


Операторы условия

5. Оператор множественного выбора ***Select Case*** удобнее применять, если требуется проверка нескольких условий:

Select Case *Переменная*
Case *Значение1*
Последовательность операторов1
...
Case *Значение(N–1)*
Последовательность операторов(N–1)
[***Case Else***
Последовательность операторовN]
End Select

Операторы условия (примеры)




```
Dim a As Integer
5 Input a
Select Case a
Case 2
Print «неуд»
Case 3
Print «удовл.»
Case 4
Print «хорошо»
Case 5
Print «отлично»
Case Else
Print «повторите ввод»
GOTO 5
End Select
End
```

Операторы условия

5. Оператор множественного выбора ***Select Case***:

Select Case *Переменная*

Case *Значение1*

Последовательность операторов1

...

Последовательность операторовN]

End Select

В качестве «*Значение*» может быть указано:

- число;
- переменная;
- выражение;
- интервал: 1 ***To*** 10

«Последовательность операторовN» - действия, которые следует выполнить, когда не верно ***ни одно из предыдущих условий***

Операторы циклов

Алгоритмические структуры циклов применяются в случае, если какие-либо операции требуется применять определенное количество раз, или пока не выполнится некоторое условие.

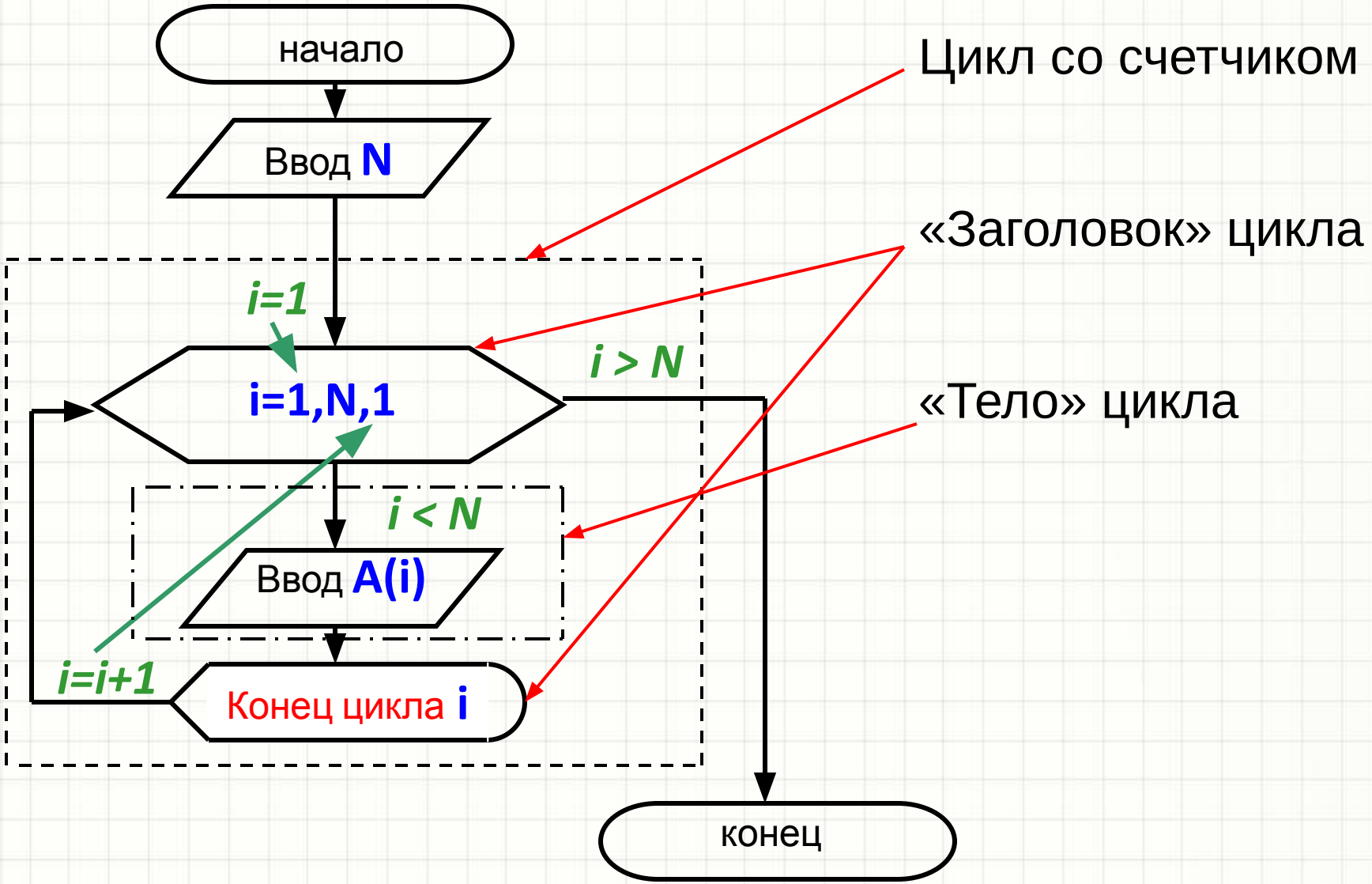
Циклы бывают:

- со счетчиком (типа ***For***) –действия в цикле повторяются известное заранее количество раз;
- с условием (типа ***Do***) – выход по условию– выполняются до выполнения заранее заданного условия.

Повторяющиеся в цикле операции называются ***телом цикла***.

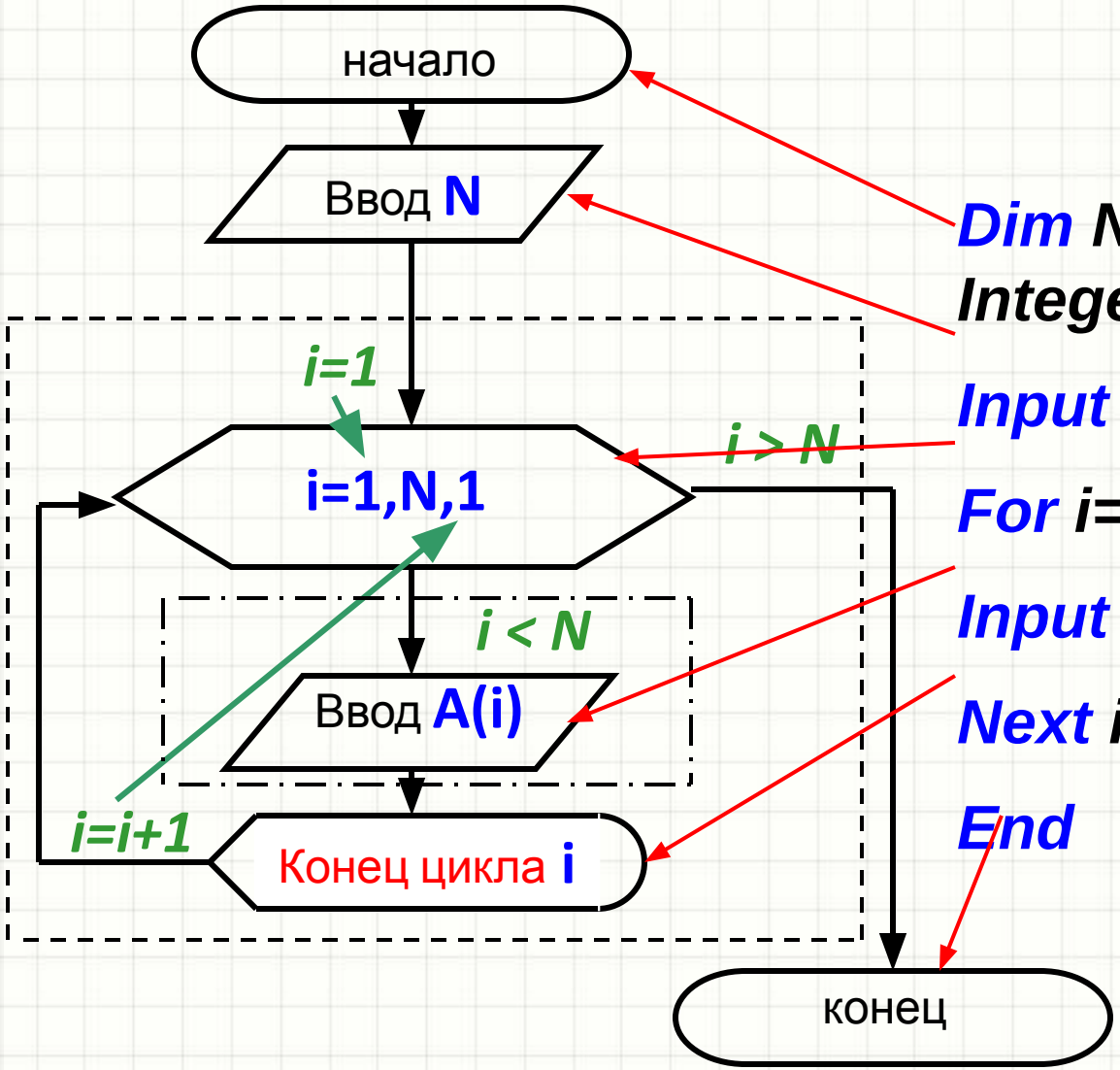
Цикл со счетчиком

Ввод значений одномерного массива



Цикл со счетчиком

Ввод значений одномерного массива




```
Dim N, A(1000) As Integer
Input N
For i=1 To N [Step 1]
Input A(i)
Next i
End
```

Цикл со счетчиком

Синтаксис:

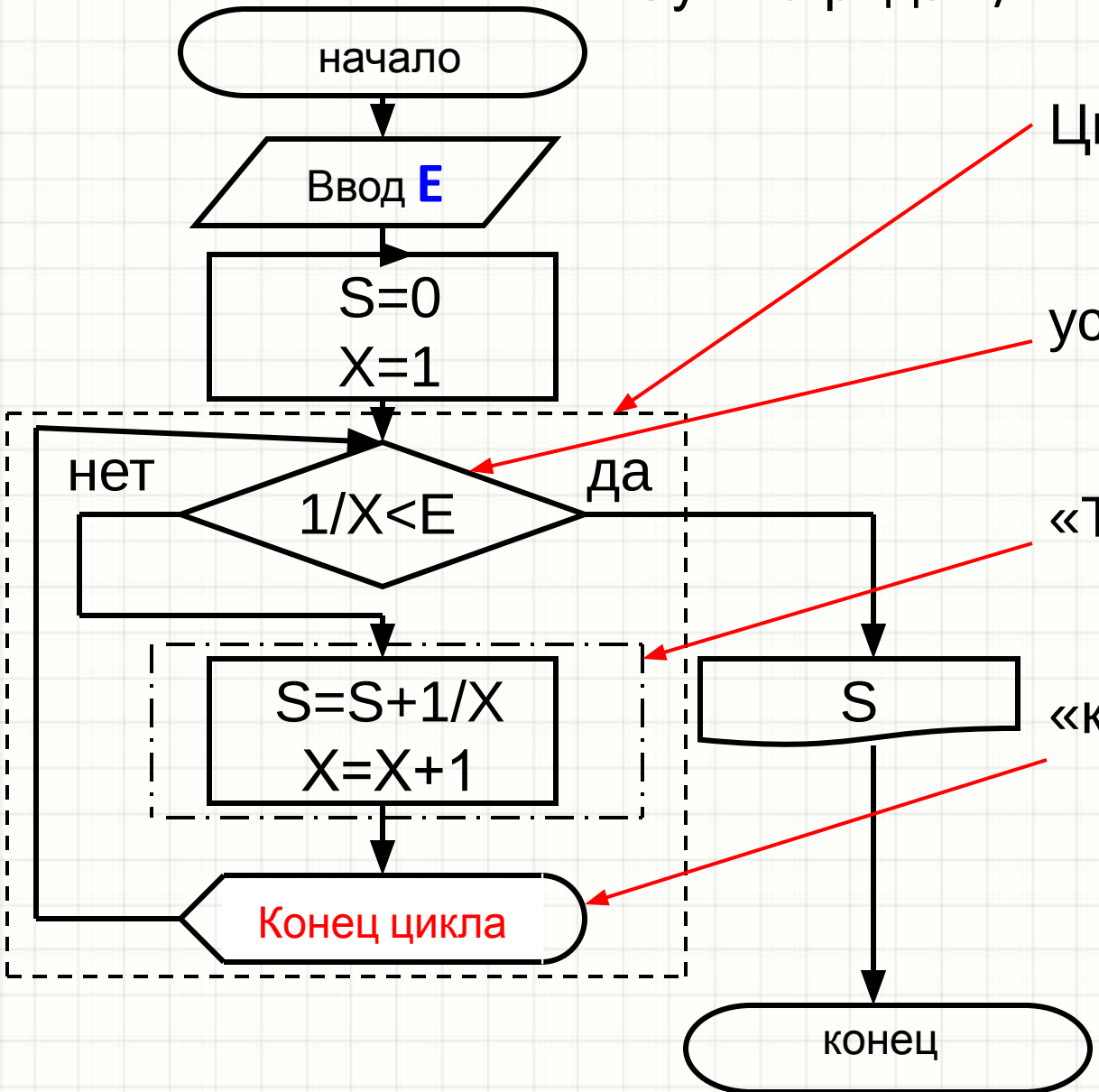
For Имя=значение1 To значение2 [Step значение3]
Операторы тела цикла
Next Имя

Имя – имя переменной (счетчика)
значение1 – начальное значение счетчика (при первом входе в цикл)
значение2 – **предельное** значение счетчика
значение3 – значение **шага** изменения счетчика

В качестве ***«Значение1 (2,3)»*** может быть указано:
- число;
- переменная;
- выражение

Цикл с условием

Сумма ряда 1/x



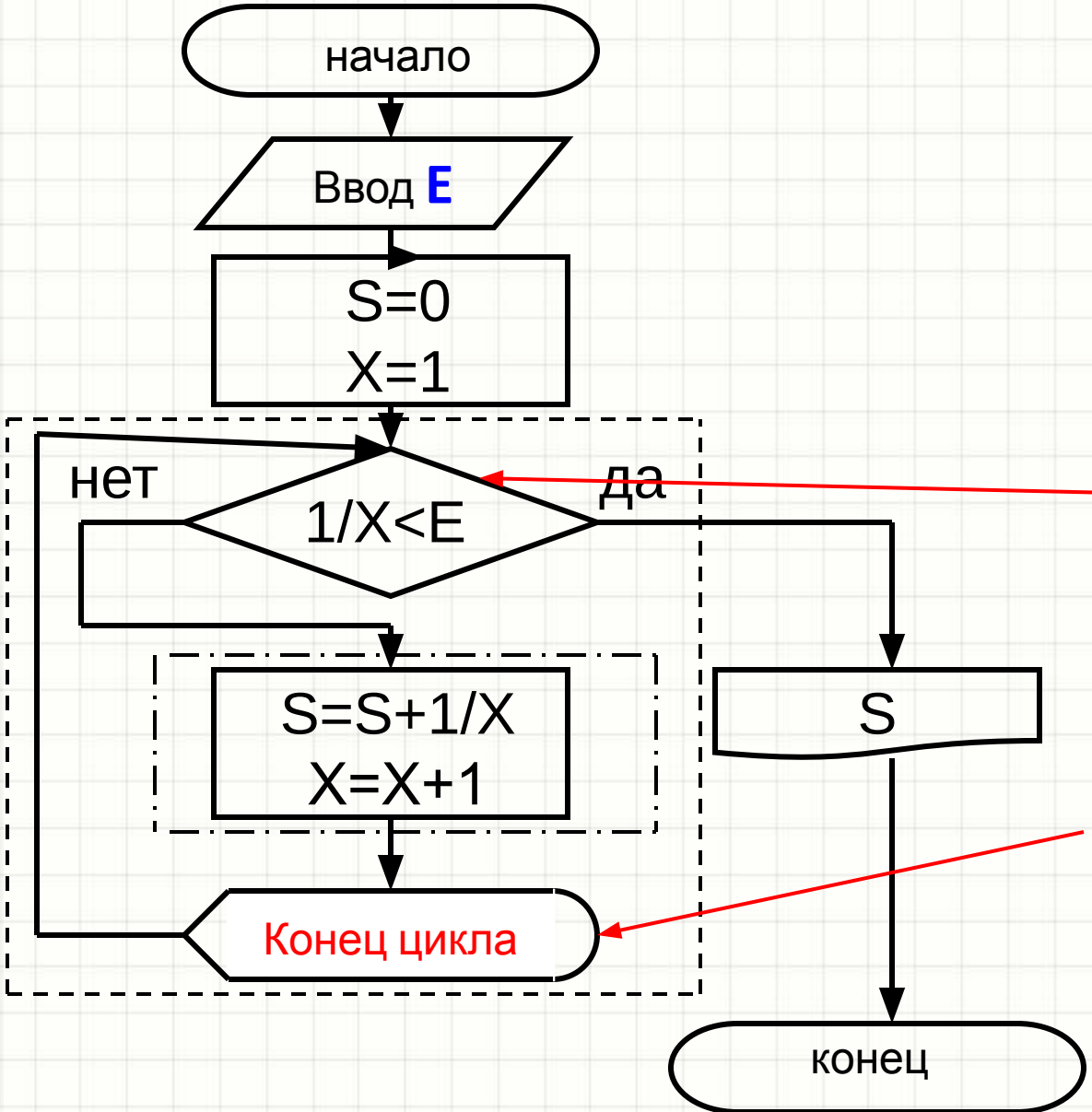
Цикл с условием

условие

«Тело» цикла

«конец» цикла

Цикл с условием




```
Dim X As Integer
Dim E,S As Double
Input E
S=0
X=1
Do Until 1/x<E
   S=S+1/X
   X=X+1
Loop
Print S
End
```

Цикл с условием

Если число повторений не известно заранее, то организуются ***циклы с условием***.

Синтаксис цикла имеет две формы в зависимости от местоположения условий:

Форма 1:

Do Условие
ОператорыТелаЦикла
Loop

ОператорыТелаЦикла могут быть не выполнены **ни разу**

Форма 2:

Do
ОператорыТелаЦикла
Loop Условие

ОператорыТелаЦикла всегда будут выполнены **хотя бы один раз**

Цикл с условием

Условие тоже бывает двух типов:

– С ключевым словом ***While*** (условие продолжения цикла).

В этом случае ***ОператорыТелаЦикла*** выполняются, если значение ***УсловногоВыражения*** есть **Истина** (**True**), иначе цикл завершается.

Форма 1:

Do While УсловноеВыражение
ОператорыТелаЦикла
Loop

Форма 2:

Do
ОператорыТелаЦикла
Loop While УсловноеВыражение

Цикл с условием

– С ключевым словом ***Until*** (условие завершения цикла). В этом случае ***ОператорыТелаЦикла*** выполняются, если значение ***УсловногоВыражения*** есть **Ложь** (**False**), иначе цикл завершается.

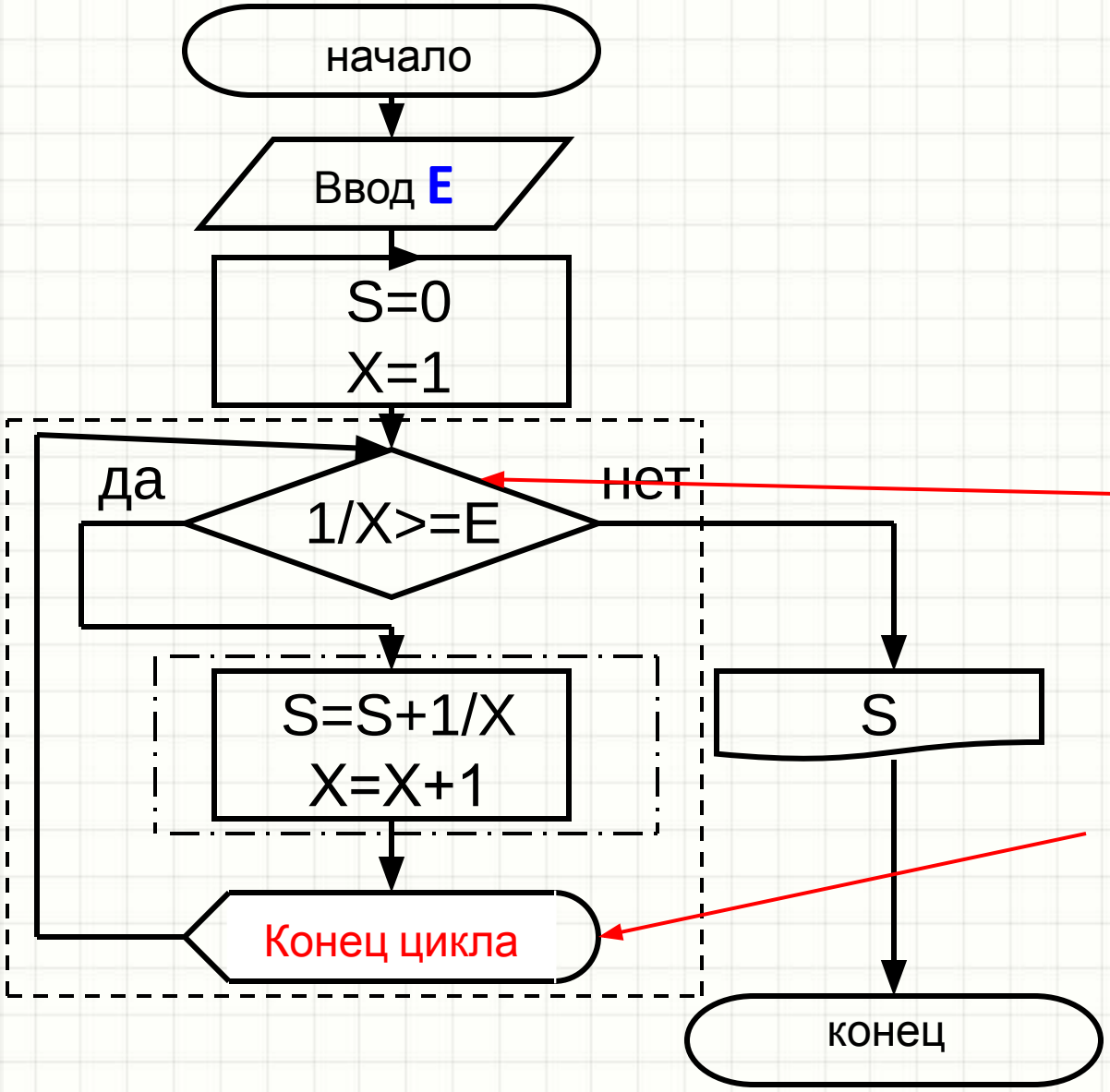
Форма 1:

Do Until УсловноеВыражение
ОператорыТелаЦикла
Loop

Форма 2:

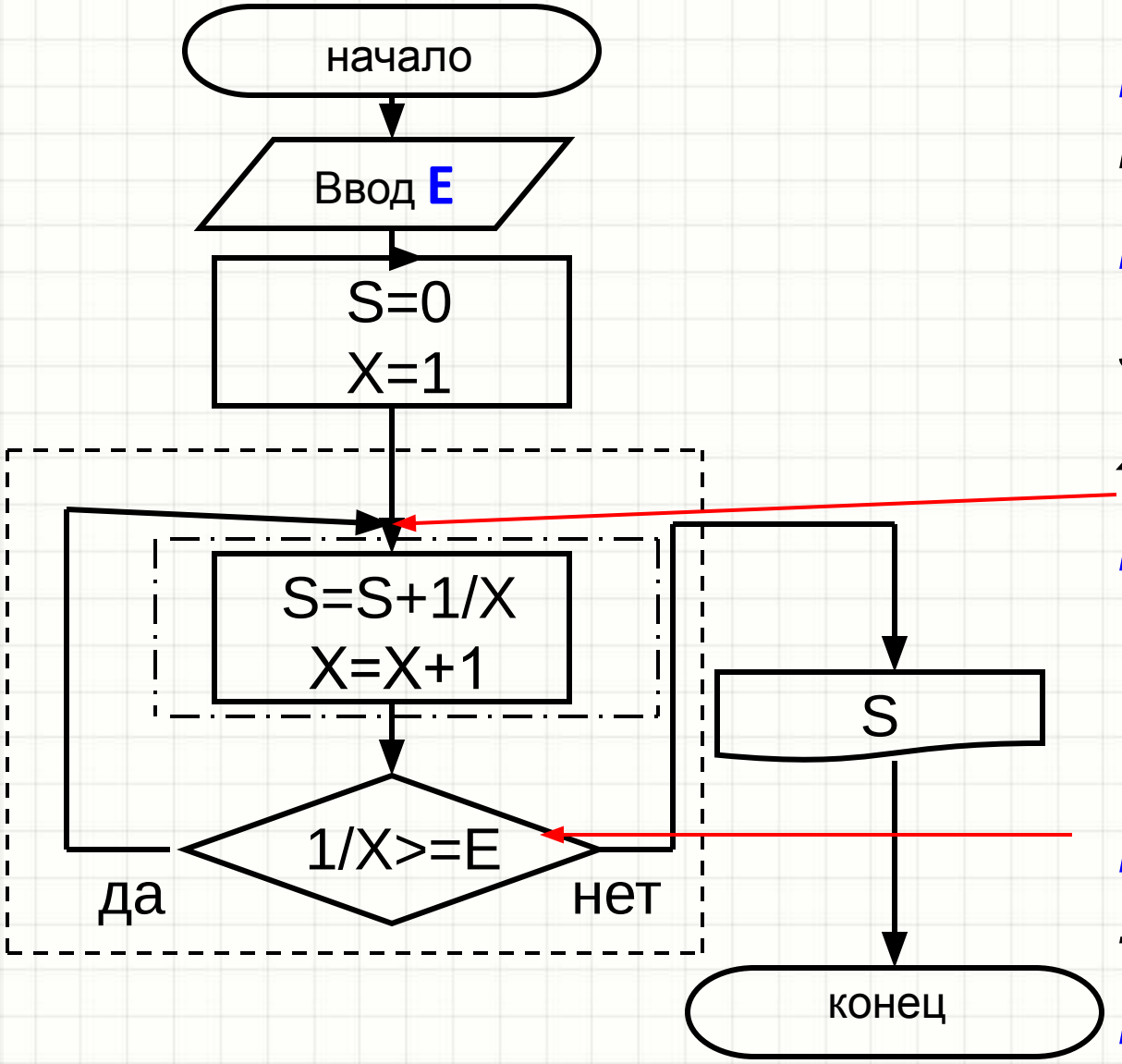
Do
ОператорыТелаЦикла
Loop Until УсловноеВыражение

Цикл с условием




```
Dim X As Integer
Dim E,S As Double
Input E
S=0
X=1
Do While 1/x>=E
   S=S+1/X
   X=X+1
Loop
Print S
End
```

Цикл с условием




```
Dim X As Integer
Dim E,S As Double
Input E
S=0
X=1
Do
   S=S+1/X
   X=X+1
Loop While 1/x>=E
Print S
End
```