Начало программирования на языке Pascal

Учитель Информатики и ИКТ «ГБОУ Школа № 1056», Маркова А.И.

Алгоритм

- Понятное и точное предписание исполнителю выполнить конечную последовательность действий, приводящих к заданному результату.

Свойства алгоритма:

1. Дискретность (пошаговость, последовательность) – строгая последовательность отдельных действий, выполняемых один за другим.

2. Результативность (конечность) – выполнение алгоритма должно приводить к результату за конечное число шагов.

3. Точность (определённость, формальность) – каждая команда алгоритма однозначно определяет действие исполнителя.

4. Понятность – алгоритм состоит только из команд, входящих в систему допустимых команд исполнителя.

5. Массовость (не обязательное свойство) – алгоритм подходит для решения целого класса задач.

Способы записи алгоритмов

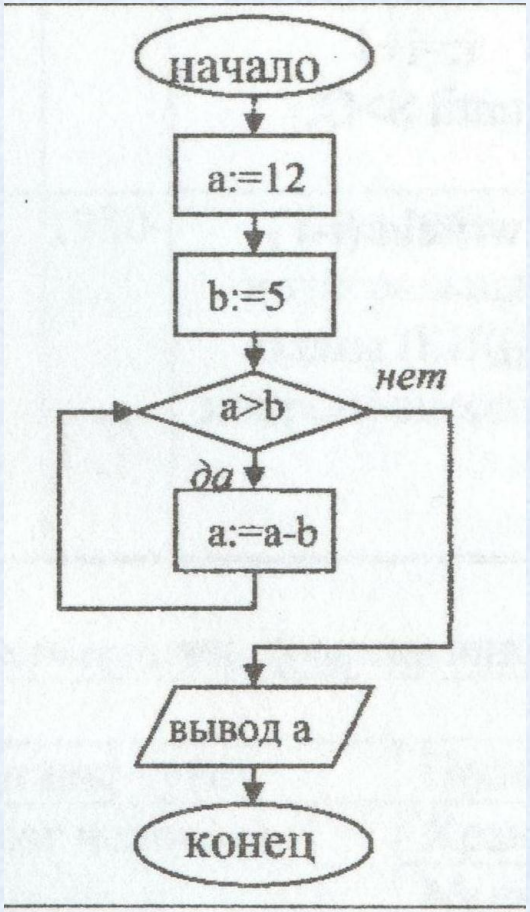
Словесный

(школьный алгоритмический язык или псевдокоды)

```
алг остаток от деления
нач цел a, b
  a:=12

  b:=5

  нц пока a>b

    a:=a-b
  кц

  вывод a

кон
```

Графический

(блок-схемы)

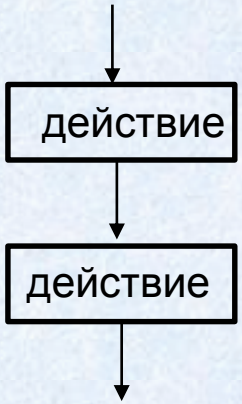


Язык программирования

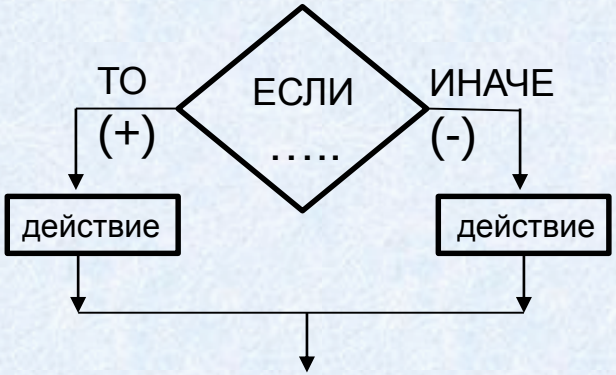
```
program ostatok;
var a, b: integer;
begin
   a:=12;
   b:=5;

   while a>b do

     a:=a-b;


   writeln (a)

end.
```

Основные алгоритмические конструкции

Следование

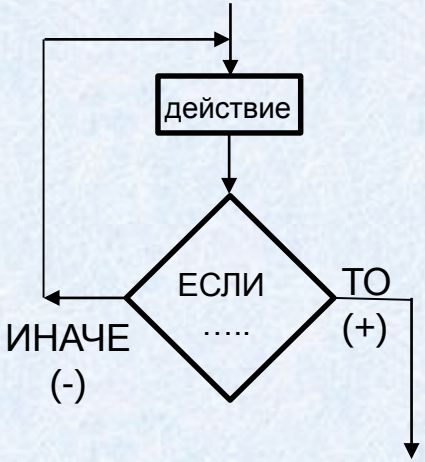


Ветвление

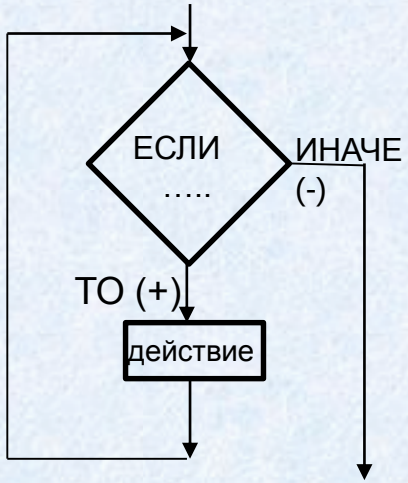


Цикл

Постусловие



Предусловие



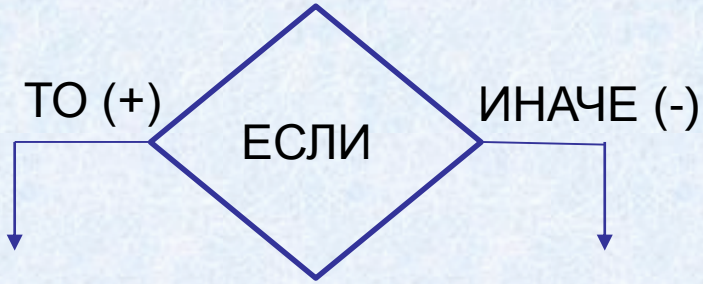
Основные элементы блок-схемы

начало программы
конец программы

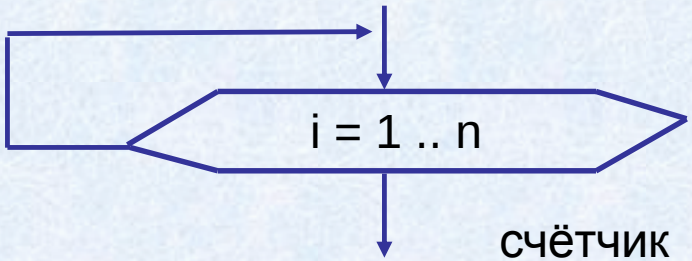
Связывающий элемент

действие

ввод данных
вывод данных

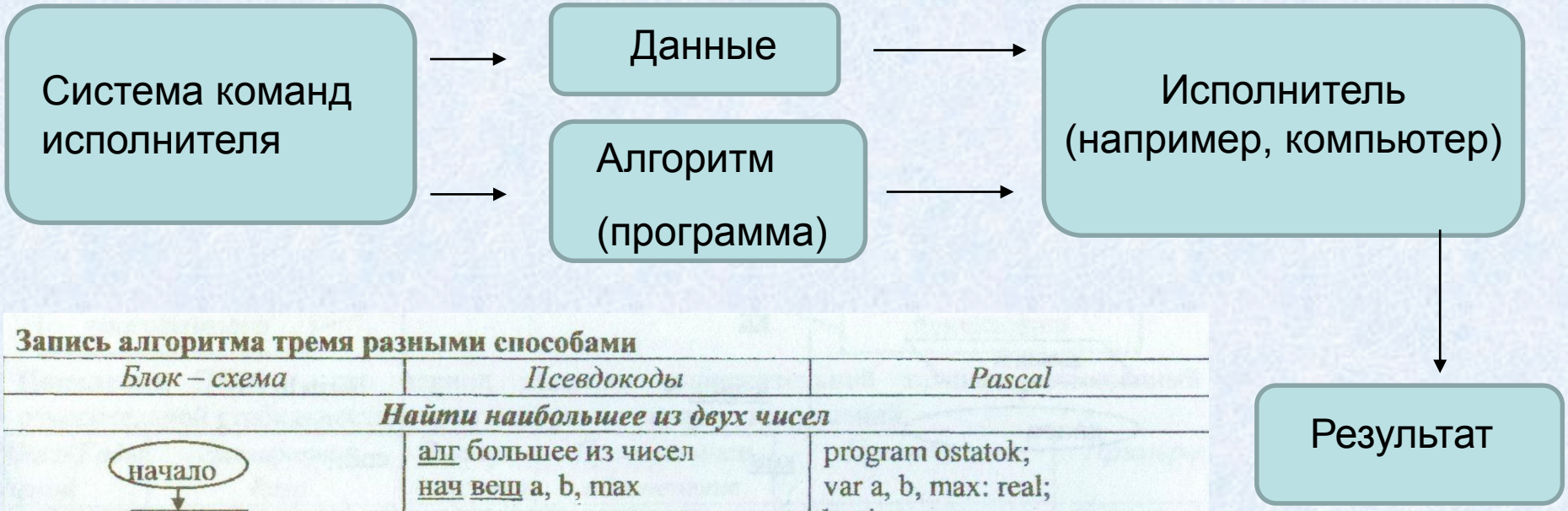


Конструкция условие



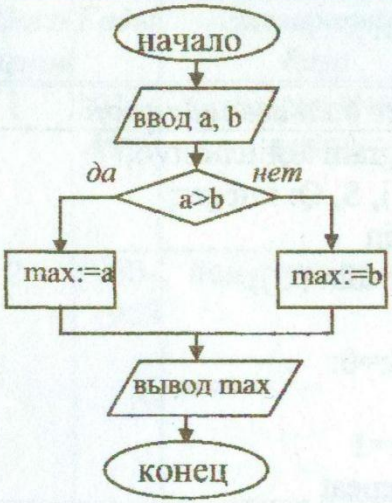
счётчик (например. Считает кол-во повторений)

Программа - последовательность команд, которые следует выполнить компьютеру для реализации заданного алгоритма.



Запись алгоритма тремя разными способами

Блок – схема	*Псевдокоды*	*Pascal*
Найти наибольшее из двух чисел		

	алг большее из чисел
	нач вещ a, b, max
	
	ввод a, b
	если a>b
	то max:=a
	иначе max:=b
	все
	
	вывод max
	
	кон
		program ostatok;
		var a, b, max: real;
		begin
		readln (a, b);
		if a>b
		then max:=a
		else max:=b;
		
		writeln (max)
		
		end.

Структура программы

на языке программирования Pascal

Название программы пишется
либо латиницей
либо на английском языке

```
Program_название программы;
```

Раздел
описания переменных

```
var
переменные : тип чисел;
```

Тело программы

```
begin
    ……;
    ……;
    ……;
```

Конец программы

```
end.
```

Специальные символы языка Pascal

символ	значение
:=	Присваивание переменной, стоящей слева от символа, значения выражения стоящего справа
;	Разделитель операторов в программе
()	Скобки для арифметических и логических выражений
:	Разделитель в описаниях переменных и формате операторов вывода
..	Многоточие для списков
+ -	Бинарные операции (не только арифметические)
* /	Бинарные операции (не только арифметические)
=	Логическое равенство, элемент описание констант
<>	Логическое неравенство
<	Отношение «меньше чем»

символ	значение
>	Отношение «больше чем»
<=	Отношение «меньше или равно»
>=	Отношение «больше или равно»
.	Конец программы, модуля, а так же десятичная точка в буквальных константах вещественного типа
{ } (* *)	Пары скобок для комментариев
[ ]	Скобки, для ссылки на элемент массива или указания диапазона значений индекса
^	Символ разыменования указателя
@	Операция получения указателя
@@	Операция получения адреса
_	Символ подчёркивания. Иногда заменяет пробел.

Зарезервированные слова, наиболее часто используемые в программах, и их краткий смысл

Зарезервированные слова	Смысл
Program unit	Заголовки, т.е. первые операторы программ и библиотечных модулей
var const procedure function	Для описания переменных, констант и составных частей программ (подпрограммы-функции, подпрограммы-процедуры)
type array record .. end string file of	Операторы описания типов переменных, задаваемых пользователем
begin end	Слова, используемые для программирования составных операторов, а так же начинающие и оканчивающие последовательность исполняемых операторов программы
implementation interface	В библиотечных модулях используются зарезервированные слова

Зарезервированные слова	Смысл

If … then … else
for … to… do
repeat … until
case … of… end
for … downto … do
while … do
	Операторы, управляющие ходом выполнения программы – управляющие операторы

Div Mod Shl Shr Sqrt Sqr And Or Not	Зарезервированные слова для обозначения арифметических операций и логических операций
Object Constructor Destructor Public virtual	В программах, написанных с использованием методов объектно-ориентированных языков программирования

Работая над программой, нам необходимо указать к какому типу данных относиться переменная, используемая в программе, а так необходимо определиться к какому типу записи будет принадлежать сама переменная.

Вся эта информация указывается в разделе описания переменных:

Раздел описания переменных:

```
var
переменные : тип чисел;
тип данных : данные;
```

Рассмотрим каждый тип данных более подробно.

Целые числа

Целые числа — расширение множества натуральных чисел N, получаемое добавлением к N нуля и отрицательных чисел вида (- n).
Множество целых чисел обозначается Z.
Необходимость рассмотрения целых чисел продиктована невозможностью, в общем случае, вычесть из одного натурального числа другое — можно вычитать только меньшее число из большего.

Название числового типа данных	Длина, байт числового типа данных	Диапазон значений числового типа данных
Byte	1	0..255
ShortInt	1	-128..+127
Word	2	0..65535
Integer	2	-32768..+32767
LongInt	4	-2 147 483 648..+2 147 483 647

Целые числа –**integer** – все целые числа

(1, 2, 99, 125 и т.д.)

С целыми числовыми типами данных

можно выполнять следующие операции:

Арифметические:

- сложение(+); x + x
- вычитание(-); x - x
- умножение(*); x * 5
- остаток от деления (mod); 2,6
- целая часть от деления нацело (div); 2,6
- возведение в степень; sqr x
- вычисление квадратного корня; sqrt x
- унарный плюс (+);
- унарный минус (-).

С **целыми** **числовыми типами данных**

можно выполнять следующие операции:

Операции отношения:

- отношение равенства (=); x = x
- отношение неравенства (<>); x <> x
- отношение меньше (<); x < x
- отношение больше (>); x >x
- отношение не меньше (>=); x >= x
- отношение не больше (<=) x <= x

Вещественные числа

Вещéственное, или **действи́тельное число** — математический объект, возникший из потребности измерения геометрических и физических величин окружающего мира, а также проведения таких вычислительных операций, как извлечение корня, вычисление логарифмов, решение алгебраических уравнений, исследование поведения функций.

Длина числового типа данных, байт	Название числового типа данных	Количество значащих цифр числового типа данных	Диапазон десятичного порядка числового типа данных
4	Single	7..8	-45..+38
6	Real	11..12	-39..+38
8	Double	15..16	-324..+308
10	Extended	19..20	-4951..+4932
8	Comp	19 . .20	$-2*10^{63}+1..+2*10^{63}-1$

Вещественные числа – **real** – целые + дробные
(0, 0,25, 105, 201,5 и т.д.)

С **действительными числовыми типами данных** можно выполнять следующие операции:

Арифметические:

- сложение(+); x + x
- вычитание(-); x - x
- умножение(*); x * 5
- возведение в степень; sqr x
- вычисление квадратного корня; sqrt x
- унарный плюс (+);
- унарный минус (-).

С **действительными** числовыми типами данных можно выполнять следующие операции:

Операции отношения:

- отношение равенства (=); x = x
- отношение неравенства (<>); x <> x
- отношение меньше (<); x < x
- отношение больше (>); x >x
- отношение не меньше (>=); x >= x
- отношение не больше (<=) x <= x

Типы данных

Символьный – **char** – используется для описания символьных переменных (f, k, b, r, tip, some, и т.д.).

Строковый тип – **string** – допустимыми значениями переменных или констант строкового типа являются строки символов.

Логический – **boolean** - принимает 2 значение ложь (false) и истина (true)

Скалярный тип – перечисляемые типы отрезков и отрезки типов: type

Prism = 0..6; ‘описание отрезка типа

Типы данных

Массивы:

- одномерные - array [n..m] of <type>;
- Двумерные – twodim : array [k..n, k..m] of <type>;
  twodim : array [k..n] of array [k..m] of <type>

Множество – **set** - задаёт совокупность значений, которая является множеством всех подмножество базового типа.

Запись – **record** - структурный тип данных, который содержит определенное число элементов (полей), причем эти записи могут иметь разный тип.

Файловый тип:

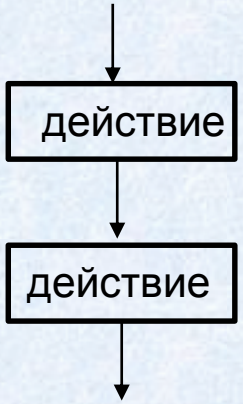
- текстовый тип - **text**;
- типизированный – **file of type_ID**;
- не типизированный – **file**;

Алгоритмическая конструкция

СЛЕДОВАНИЕ

Следование – алгоритмическая конструкция, в которой каждое следующее действие идет за предыдущим, без каких-либо отклонений и вариантов.

Конструкция:



Оператор := это оператор присвоения, т.е. какой-то переменной мы задаём конкретное значение.

Например:

x := 5; - переменной x задаём значение 5

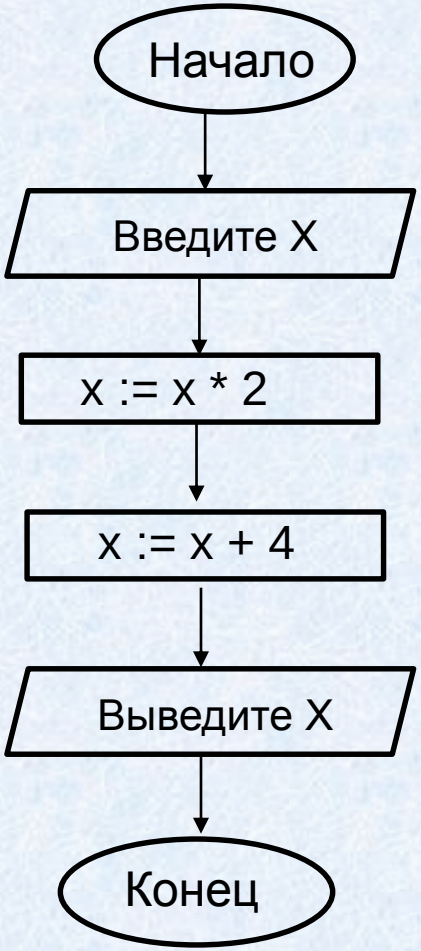
z := x + 5; - значением переменной z является сумма переменной x и числа 5.

Пример: Выполните последовательное умножение на два и прибавьте к получившемуся числу четыре. Число вводится с клавиатуры пользователем.
Ответ выведите в виде одного числа.

Словесный алгоритм:

1. Введите число x;
2. Умножьте полученное число на два;
3. Прибавьте к полученному числу четыре;
4. Выведите результат

Блок-схема



Программа на языке Pascal:

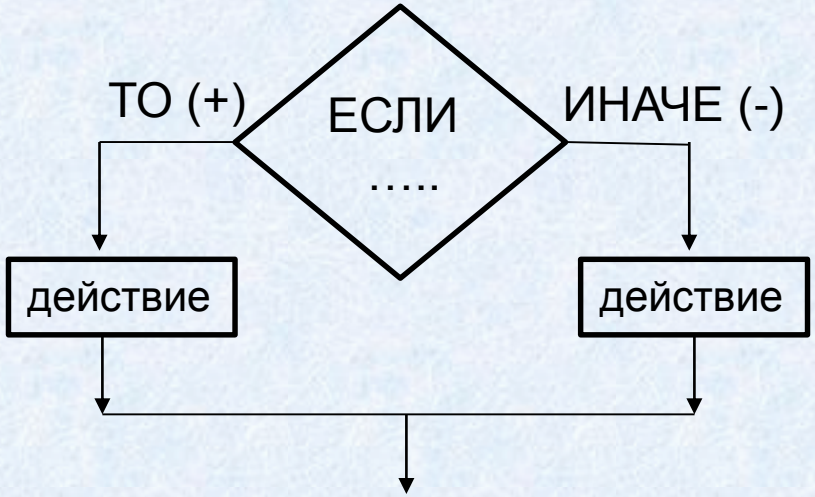
```
Program 1;
var
   x : integer;
begin
   readln (x);
   x := x * 2;
   x := x + 4;
   writeln (x);
end.
```

Алгоритмическая конструкция

ВЕТВЛЕНИЕ

Ветвление – алгоритмическая конструкция, в которой задаётся некоторое условие(вопрос), при ответе на который, выполнятся действие, в зависимости от ответа «Да» или «Нет»

Конструкция:



Служебные слова конструкции:

If … then … else;

если … то … иначе

Пример:

Определите, чему равно значение z после сравнения 2-х чисел, при условии, что если x > y то к z прибавляется x, иначе если x < y, то к z прибавляется y, иначе из z вычитается x. Случай, когда x=y в данной задаче не рассматривается. Числа вводятся с клавиатуры. Начальное значение z = 0.

Словесный алгоритм

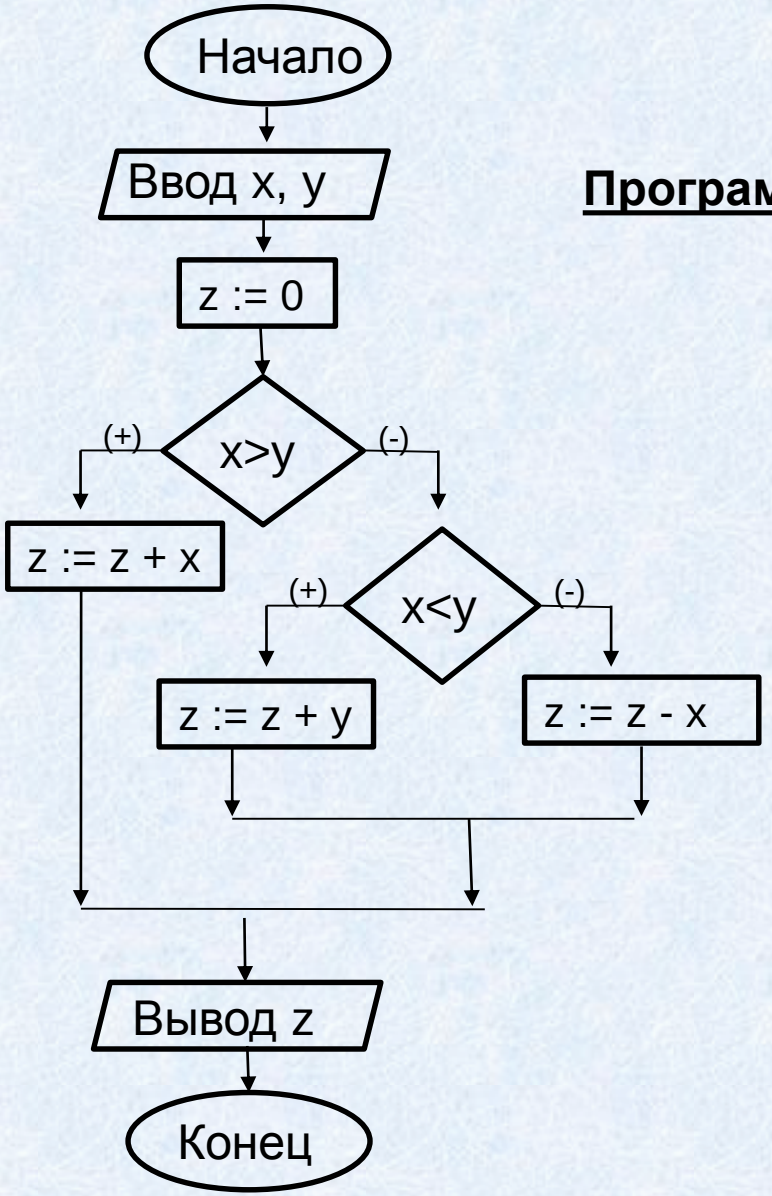
1. Введите число x
2. Введите число y;
3. z := 0
4. Если x > y, то z := z + x

   иначе если x < y,

   то z := z + y

   иначе z := z - x;
5. Выведите результат z.

Блок-схема



Программа на языке Pascal:

```
Program 2;
var
   x, y, z : integer;
begin
   z := 0;
   readln (x);
   readln (y);
   if x>y then z := z+x
          else
          if x<y then
                 z:=z+y
          else z:=z-x;
   writeln (z);
end.
```

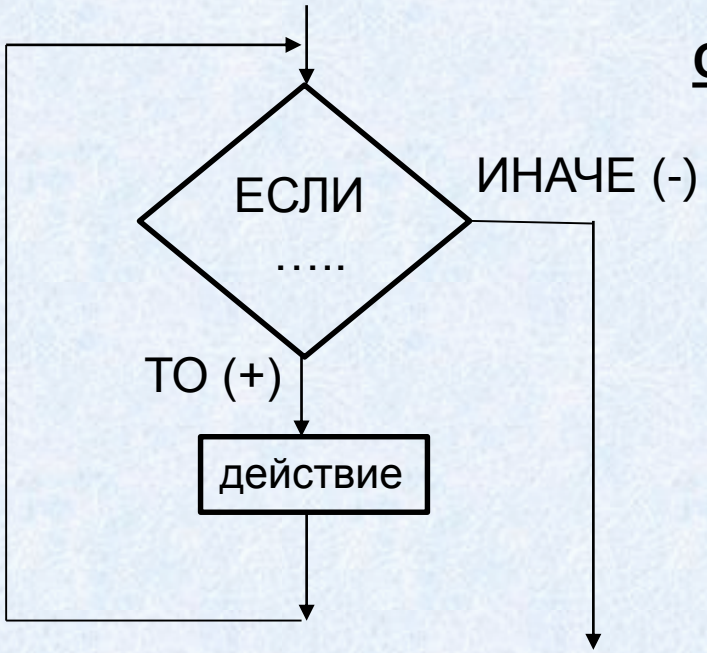
Алгоритмическая конструкция

Цикл с предусловием

Цикл – алгоритмическая конструкция, в которой некоторые действия повторяются обусловленное количество раз, указанное в условии цикла.

Цикл с предусловием – цикл, в котором в зависимости от условия, повторяется одно и тоже действие, то количество раз, пока условие не перестанет соблюдаться.

Конструкция:



Служебные слова конструкции:

while … do … else … ;

пока … делаем … иначе …;

for … to… do …;

для … (диапазон значений) … делаем …;

Пример: Определите, чему будет равна сумма чисел от 1 до n (n – вводиться с клавиатуры)

Решение: т.к. для решения данной задачи необходимо будет несколько переменных, то сначала определимся с ними:

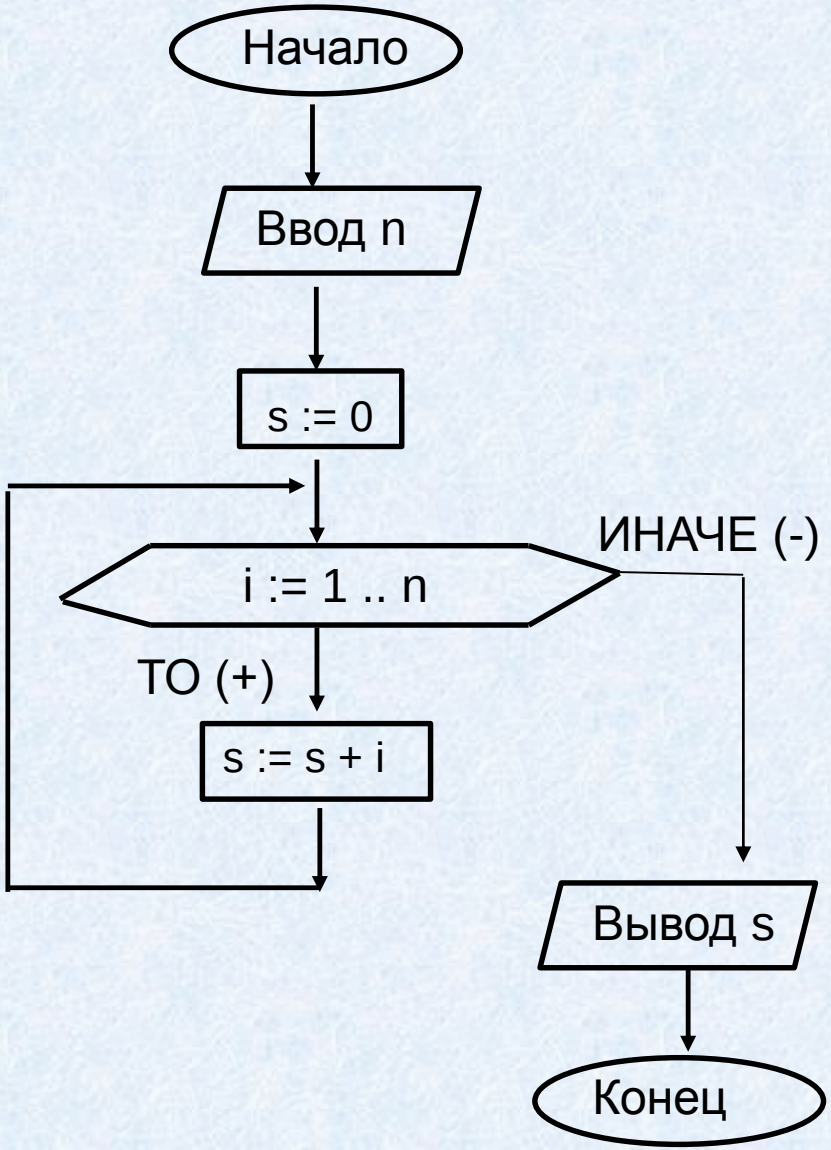
1. Подсчёт суммы – s, изначально сумма равно 0.
2. Конечное значение – n.
3. Счётчик пройденных элементов (включая n) – i.

Словесный алгоритм

1. Введите число n;
2. s := 0;
3. Для i от 1 до n делаем s := s + I
4. Вывод s.

Теперь составим решение в виде блок-схемы и напишем программу.

Блок-схема



Программа на языке Pascal

```
Program summa_3;
var
   i, n, s : integer;
begin
   readln (n);
   s := 0;
   for I := 1 to n do
   s := s + I;
   writeln (s);
end.
```

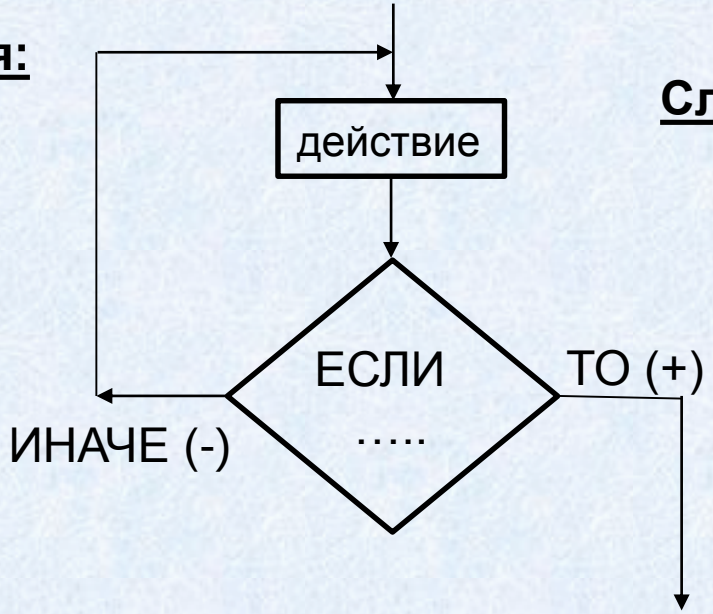
Алгоритмическая конструкция

Цикл с постусловием

Цикл – алгоритмическая конструкция, в которой некоторые действия повторяются обусловленное количество раз, указанное в условии цикла.

Цикл с постусловием – цикл, в котором после выполнения действия, происходит проверка условия, и при не выполнении условия действие выполняется до тех пор, пока на поставленное условие не будет дан положительный ответ.

Конструкция:



Служебные слова конструкции:

for … downto… do …;

repeat … until … ;

case … of … end;

Пример: Определите, произведение цифр вводимого числа. Число вводится с клавиатуры.

Решение: т.к. для решения данной задачи необходимо будет несколько переменных, то сначала определимся с ними:

1. Вводимое число – n (по условию вводиться с клавиатуры);
2. Произведение цифр в числе – p;

Так же для решения данной задачи, нам потребуются две операции:

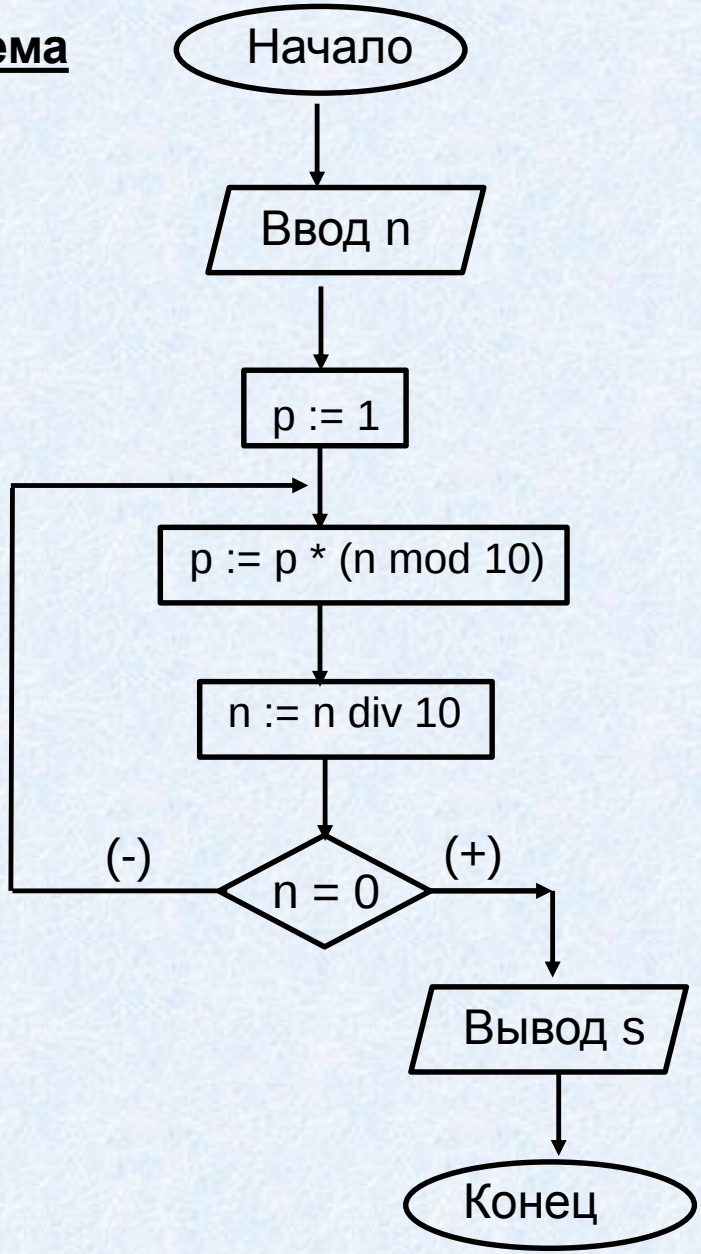
1. Деление числа нацело - div (n, 10)
2. Выделение остатка числа при делении – mod (n, 10)

Словесный алгоритм

1. Введите число n;
2. p := 1; - т.к. если значение произведения будет равно 0, то такое решение не имеет смысла и математически не верно.
3. Начало цикла
   p := p * (n mod 10);
   n := n div 10;
   До n = 0.
4. Вывод p.

Теперь составим решение
в виде блок-схемы и напишем программу.

Блок-схема



Программа на языке Pascal

```
Program 4_proizvedenie;
var
   n, p : integer;
begin
    readln (n);
    p := 1;
    repeat
       p := p * (n mod 10)
       n := n div 10
     until n = 0;
   writeln ('p = ', p);
end.
```