## Лекция 6

# Программирование линейных и разветвляющихся алгоритмов

Преподаватель Палехова Ольга Александровна,
кафедра О7 БГТУ «Военмех»

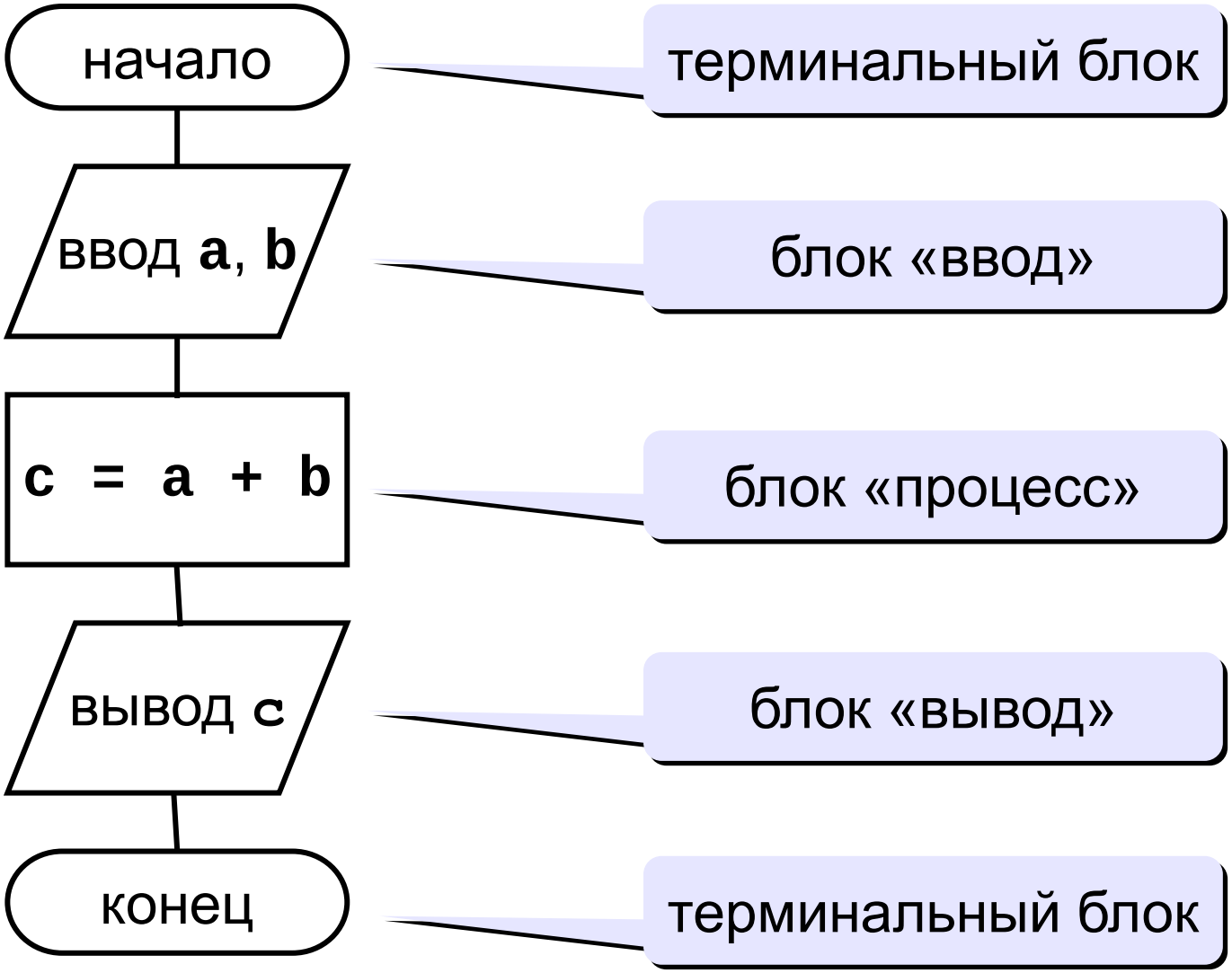
# Алгоритмы

**Алгоритм** – строго определенная последовательность действий, позволяющая получить удовлетворяющий поставленным условиям результат по исходным данным.

**Виды алгоритмов**:

- **линейный** – набор элементарных действий, выполняемых последовательно во времени друг за другом;
- **разветвляющийся** – алгоритм, содержащий две и более альтернативных последовательности действий;
- **циклический** – алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над изменяющимися данными.

# Схема линейного алгоритма



# Текст программы

```
#include <stdio.h>
int main()
{
    int a, b, c;
    printf("Введите два целых числа\n");
    scanf("%d%d", &a, &b );
    c = a + b;
    printf("c = %d", c);
    return 0;
}
```

Число блоков на схеме и команд в программе **не совпадает**

# Программирование ветвлений

1. **Универсальный** способ – инструкция ***if-else.***
2. **Множественный выбор**, организация **меню** – инструкция ***switch***.
3. **Для вычисления значений по условию** – условное выражение

   *условие* **?** *значение_если_истина* **:** *значение_если_ложь*

4. Бывает **мнимое ветвление**, для его программирования достаточно использовать одно безусловное выражение.

$$f(x) = \begin{cases} x, & \text{если } x \text{ нечетное} \\ 0, & \text{если } x \text{ четное} \end{cases}$$

```
f = x * ( x & 1 );
```

# Разветвляющийся алгоритм

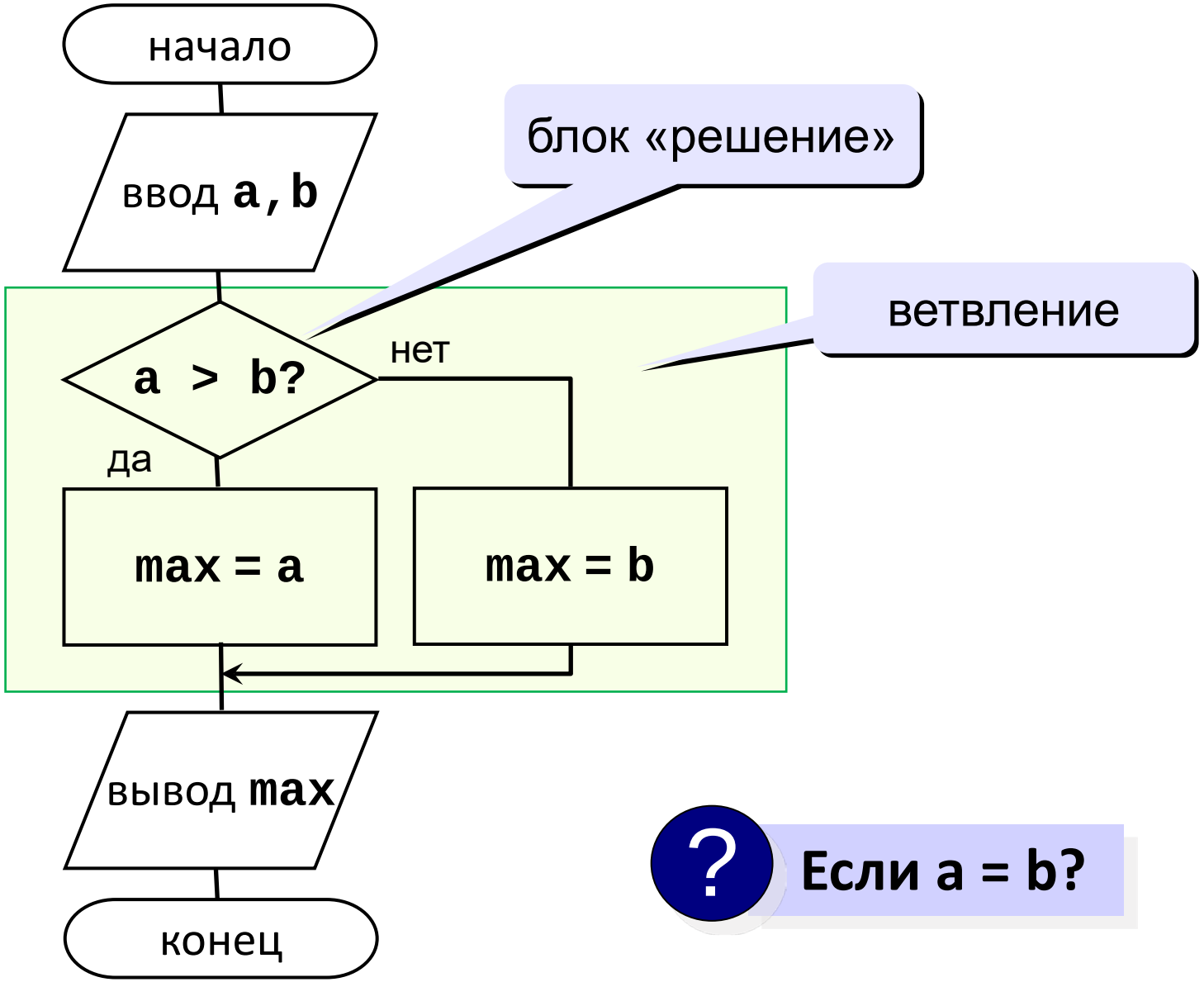
**Задача.** Ввести два целых числа и вывести на экран наибольшее из них.

**Идея решения:** надо вывести на экран первое число, если оно больше второго, или второе, если оно больше первого.

**Особенность:** есть два варианта действий исполнителя, но исполняться должен только один. Выбор варианта будет зависеть от некоторых условий, определяемых значениями данных (***если … иначе …***).


Исполнитель отработает **только одну** последовательность из всех возможных вариантов

# Схема алгоритма. Вариант 1




? Если a = b?

# Текст программы

```
#include <stdio.h>
int main()
{
    int a, b, max;
    printf("Введите два целых числа\n");
    scanf("%d%d", &a, &b );
    if (a > b)
        max = a;
    else
        max = b;
    printf("Наибольшее число %d", max);
    return 0;
}
```

полная форма инструкции *if*

условная операция ?:

```
max = a > b ? a : b;
```

значение выражения при истинности условия

значение выражения, если условие ложно

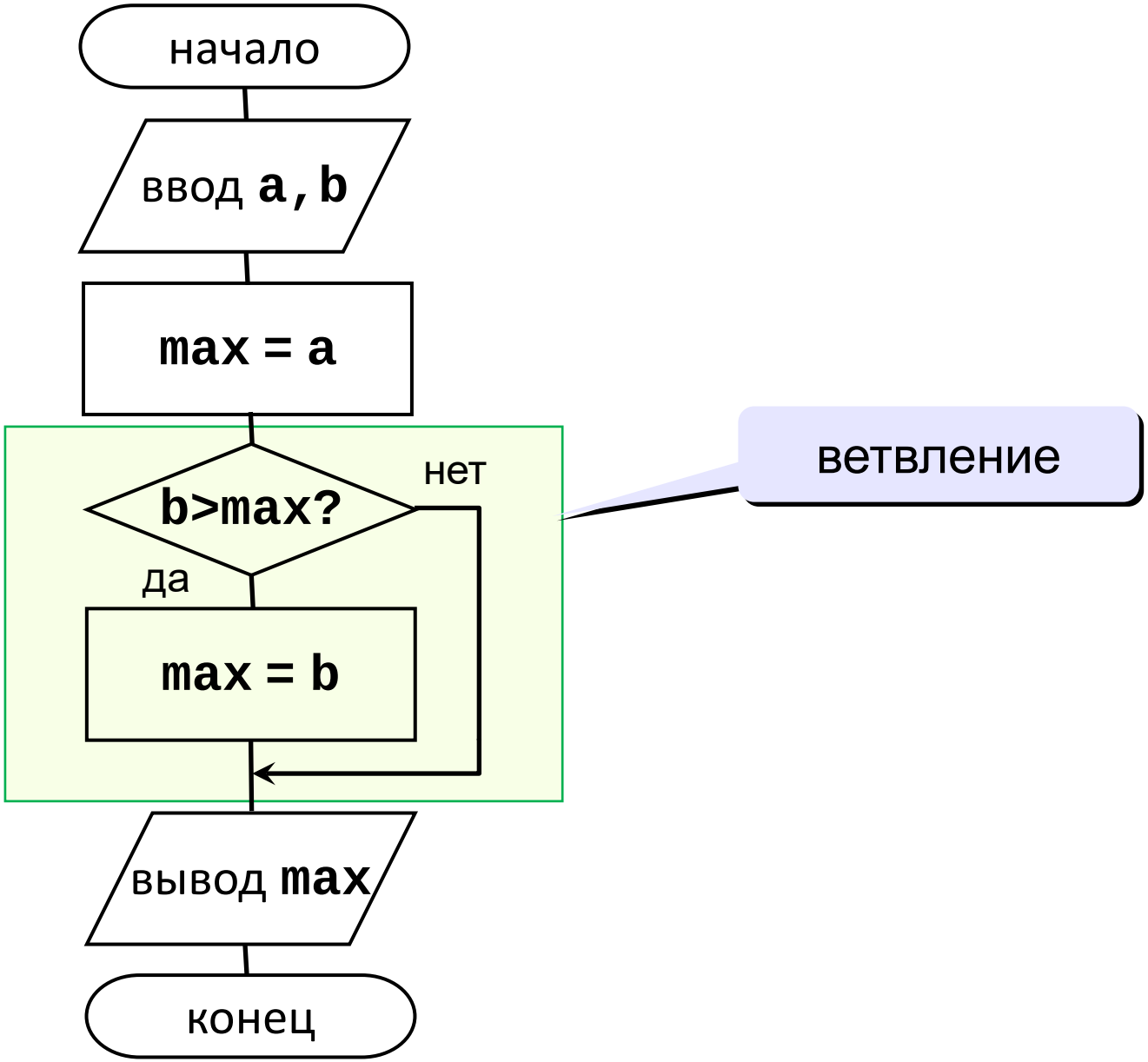
# Инструкция *if-else*

```
if ( выражение, определяющее условие )
{
        /* что делать, если значение выражения
           не равно нулю (условие верно) */
}
else
{
        /* что делать, если значение выражения
           равно нулю (условие неверно) */
}
```

**Особенности:**

- *выражение* может быть **любым**
- вторая часть (***else*** …) может отсутствовать (неполная форма)
- если в блоке один оператор, { } не нужны

# Схема алгоритма. Вариант 2



# Текст программы

```
#include <stdio.h>
int main()
{
    int a, b, max;
    printf ("Введите два целых числа\n");
    scanf ("%d%d", &a, &b);
    max = a;
    if ( b > max )
        max = b;
    printf ("Наибольшее число %d", max);
    return 0;
}
```

неполная форма инструкции *if*

```
max = b > max ? b : max;
```

! Операция *?:* является тернарной, условное выражение всегда содержит **три операнда**

# Множественное ветвление

**Задача.** Ввести два целых числа, сравнить их между собой и вывести на экран сообщение об их отношении.

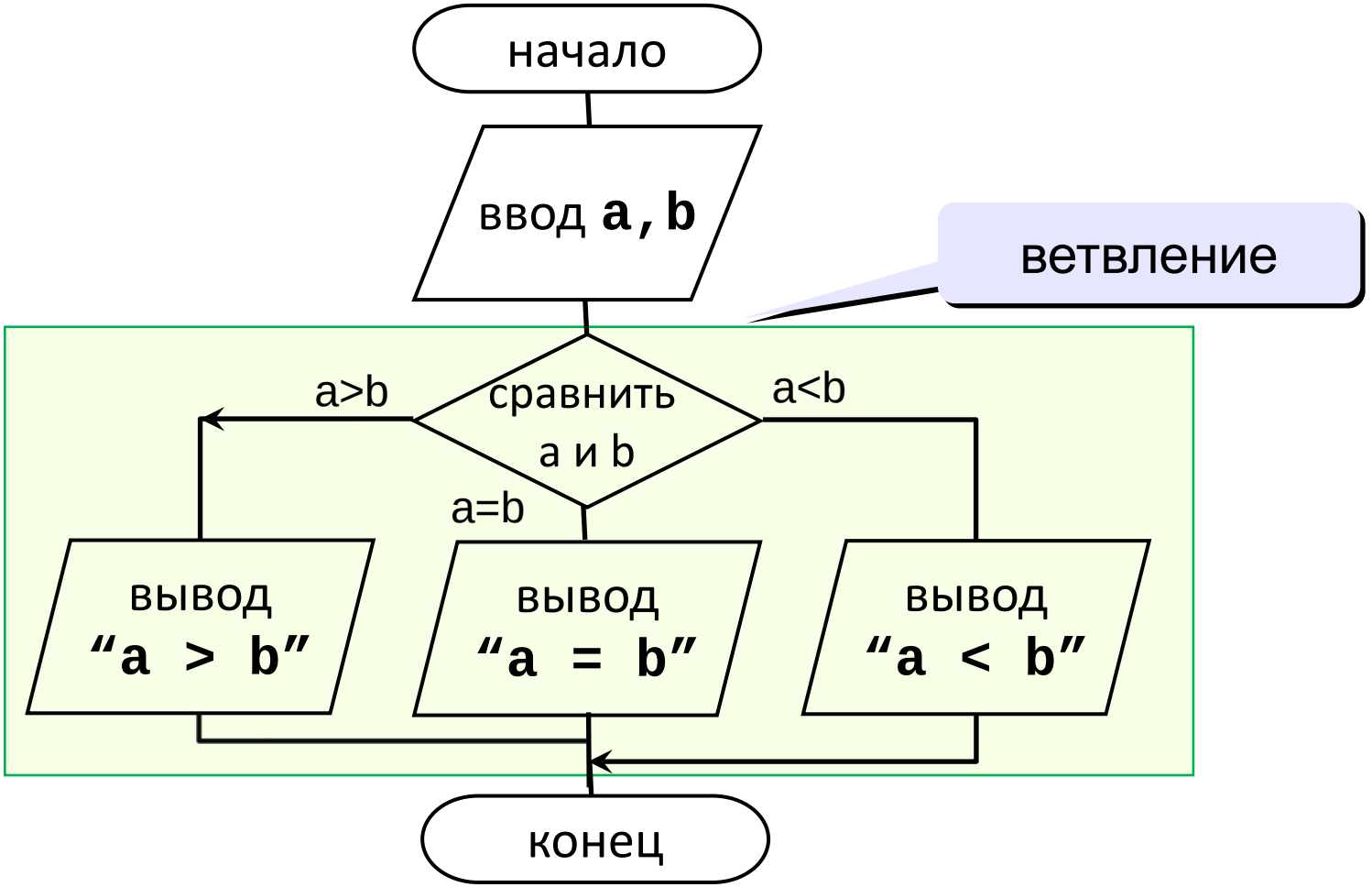
**Проблема.** Есть три варианта:

- первое число больше второго,
- первое число меньше второго,
- числа равны.

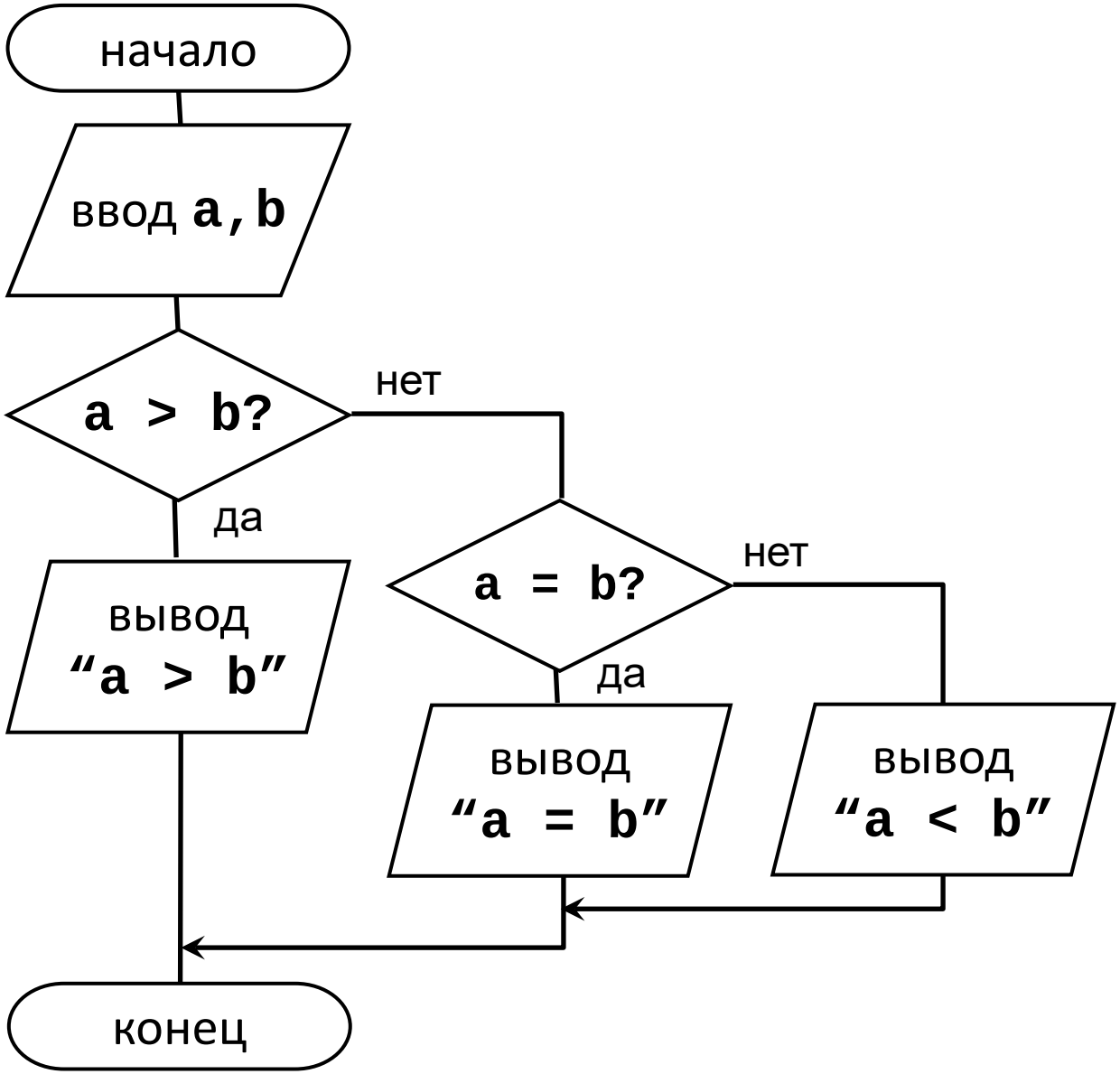
Инструкция ***if-else*** позволяет запрограммировать только две ветви алгоритма.

**Решение:** проверять условия последовательно.

# Схема алгоритма



# Схема программы



# Текст программы

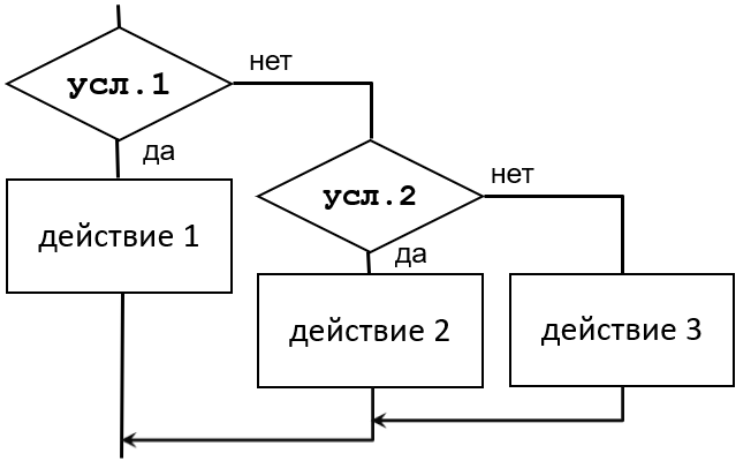
```
#include <stdio.h>
int main()
{
    int a, b;
    printf("Введите два целых числа\n");
    scanf("%d%d", &a, &b );
    if (a > b)
        printf("Первое число больше второго\n");
    else
        if (a == b)
            printf("Числа равны\n");
        else
            printf("Первое число меньше второго\n");
    return 0;
}
```

Внешняя инструкция ***if***

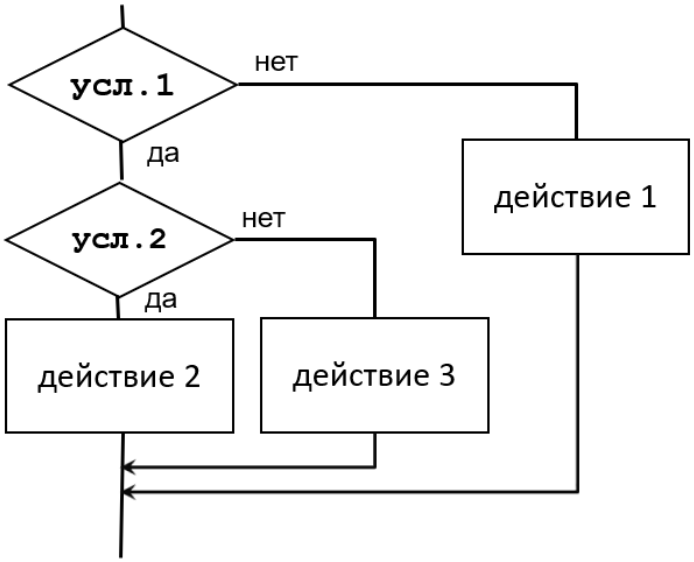
Вложенная инструкция ***if***

Отступы для читаемости

# Полные вложенные инструкции *if-else*

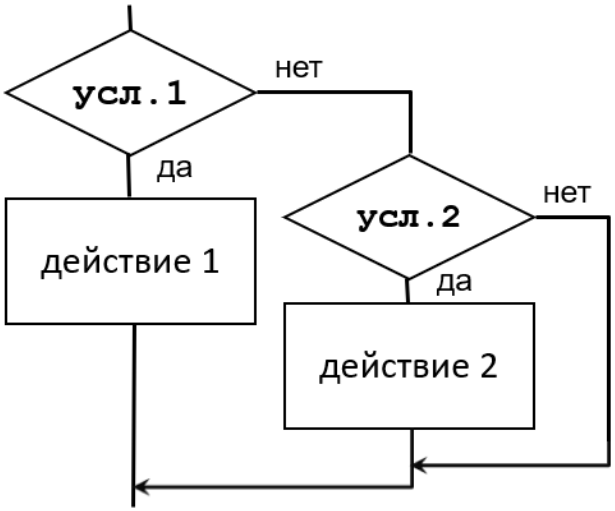



```
if ( усл.1 )
    /* действие 1*/;
else
    if ( усл.2 )
        /* действие 2*/;
    else
        /* действие 3*/;
```

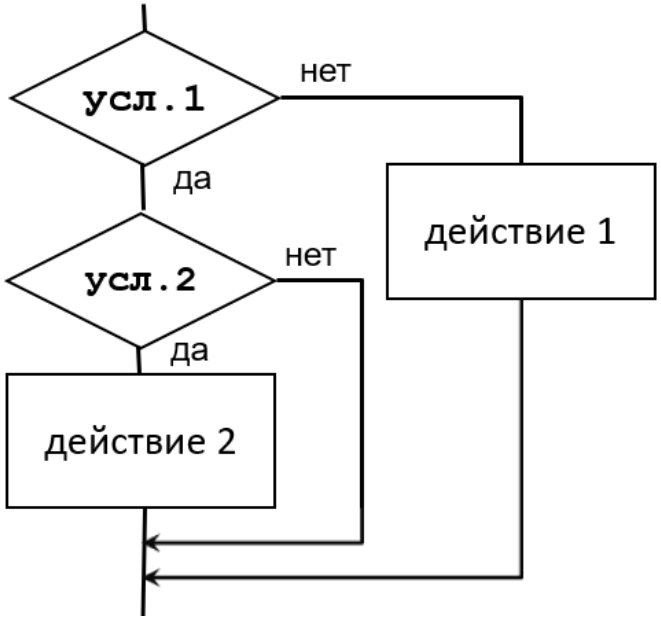



```
if ( усл.1 )
    if ( усл.2 )
        /* действие 2*/;
    else
        /* действие 3*/;
else
    /* действие 1*/;
```

# Неполные вложенные инструкции *if*




```
if ( усл.1 )
    /* действие 1*/;
else
    if ( усл.2 )
        /* действие 2*/ ;
```




```
if ( усл.1 )
{
    if ( усл.2 )
        /* действие 2*/ ;
}
else
    /* действие 1*/;
```

# Множественное ветвление

**Задача.** Ввести выражение вида

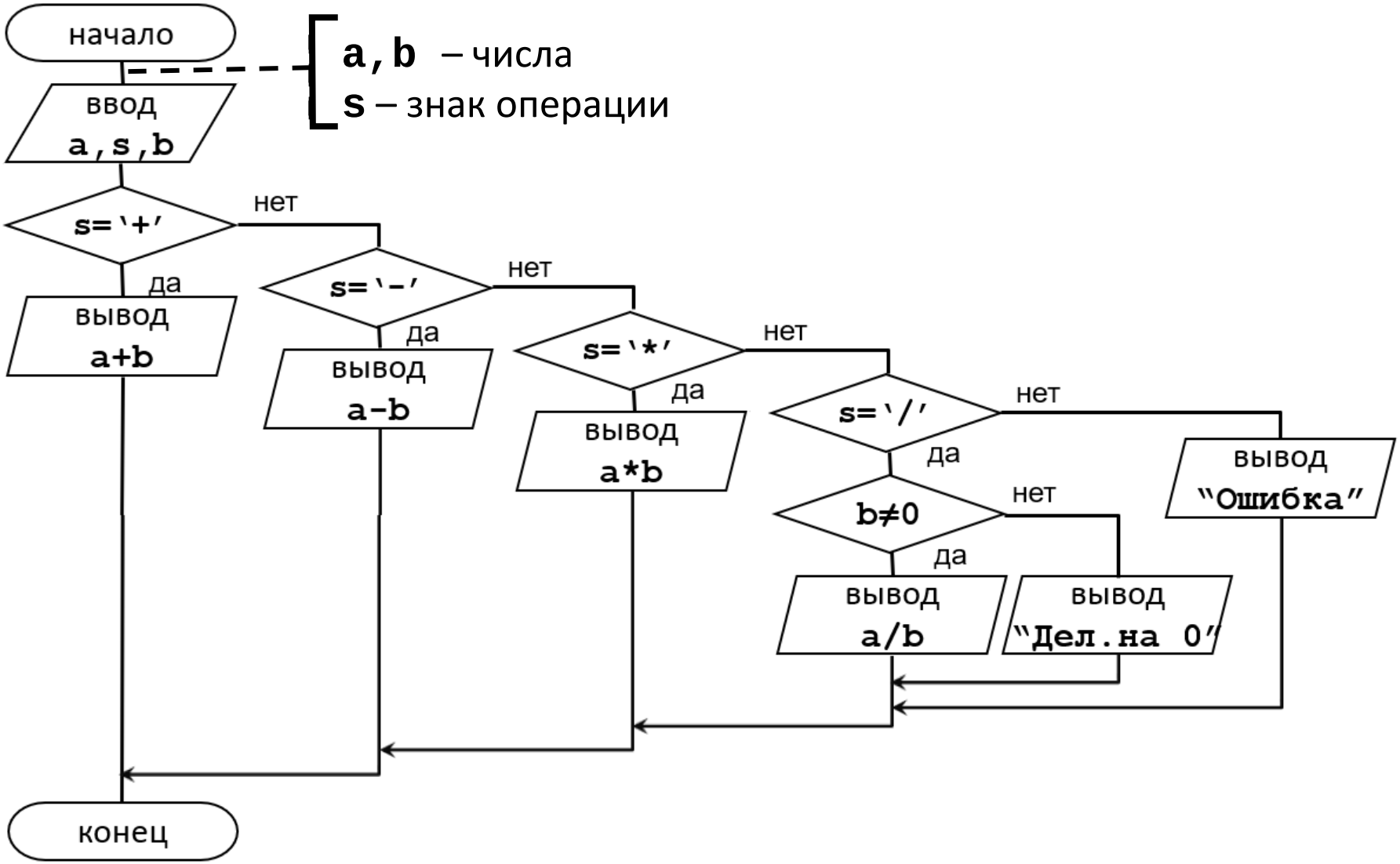
*«число знак_операции число»*

и вывести на экран его значение, если это выражение является арифметическим, в противном случае вывести сообщение об этом.

**Особенность:** нужно сравнить *знак_операции* на равенство с четырьмя знаками арифметических действий.


**Можно решить известными методами?**

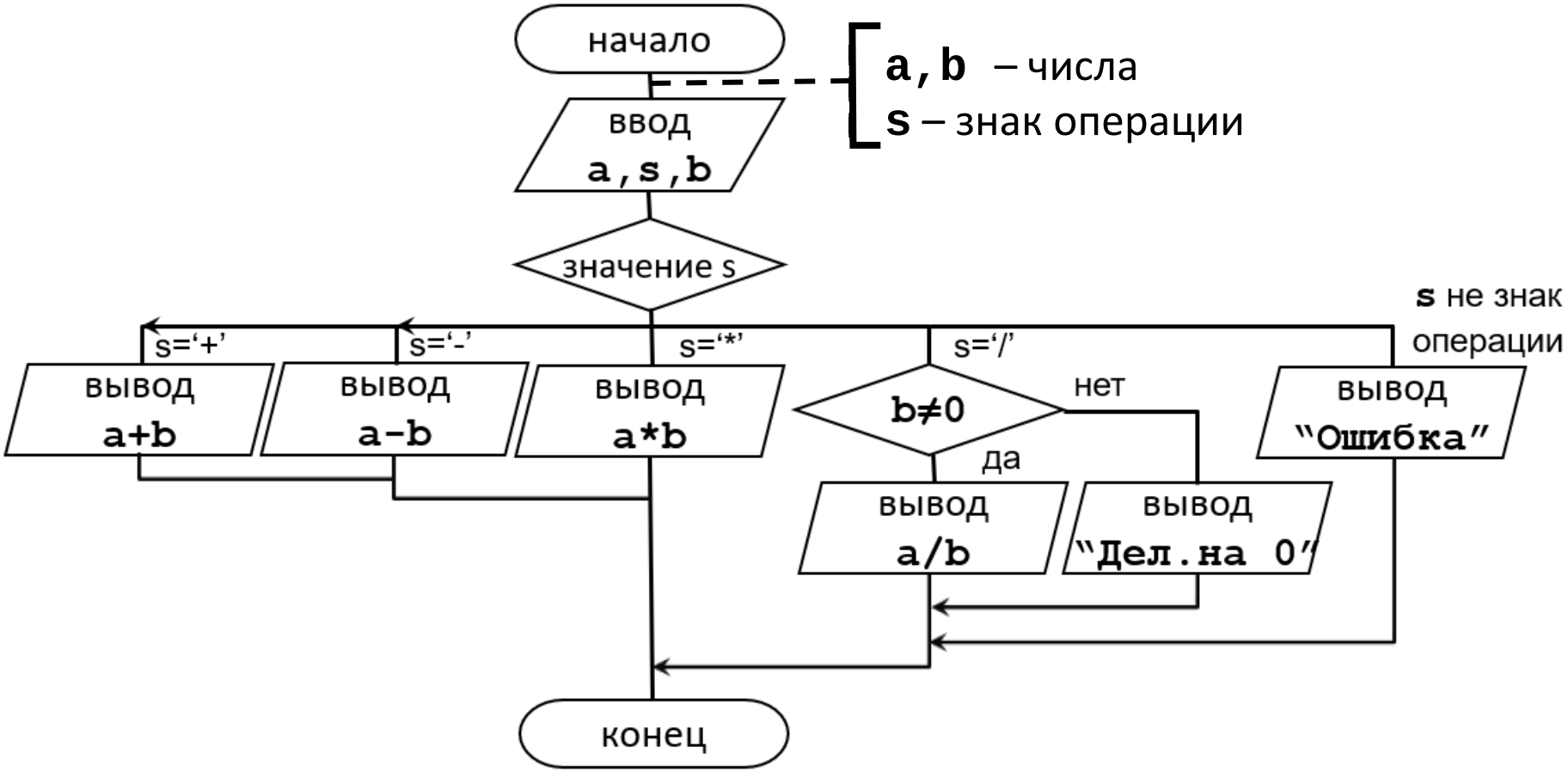
# Схема программы



# Текст программы. Инструкция *if-else*

```
#include <stdio.h>
int main()
{
    double a, b;
    char s;
    printf("Введите арифметическое выражение\n");
    scanf("%lf%c%lf", &a, &s, &b );
    if (s == '+')
        printf("%f\n", a+b);
    else
        if (s == '-')
            printf("%f\n", a-b);
        else
            if (s == '*')
                printf("%f\n", a*b);
            else
                if (s == '/')
                    if ( b )
                        printf("%f\n", a/b);
                    else
                        printf("Деление на 0\n");
                else
                    printf("Ошибка в выражении\n");
    return 0;
}
```

# Схема алгоритма



# Текст программы. Инструкция *switch*

```
#include <stdio.h>
int main()
{
    double a, b;
    char s;
    printf("Введите арифметическое выражение\n");
    scanf("%lf%c%lf", &a, &s, &b );
    switch (s)
    {
        case '+' : printf("%f\n", a+b); break;
        case '-' : printf("%f\n", a-b); break;
        case '*' : printf("%f\n", a*b); break;
        case '/' :
            if ( b )
                printf("%f\n", a/b);
            else
                printf("Деление на 0\n");
            break;
        default : printf("Ошибка в выражении\n");
    }
    return 0;
}
```

Константы для сравнения

Выйти из switch

Если нет совпадений

# Инструкция *switch*

**Особенности:**

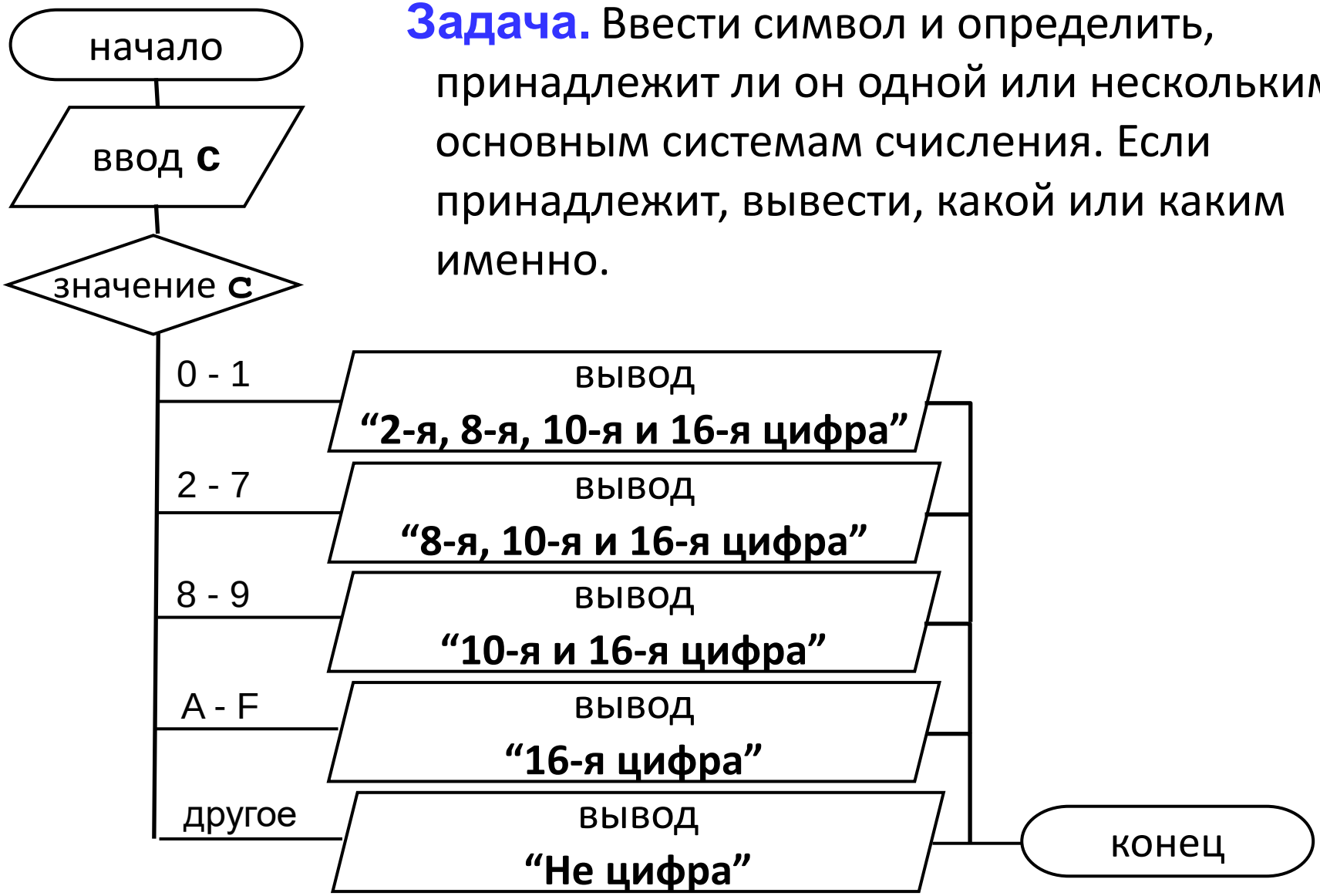
- после **`switch`** может быть имя переменной или арифметическое выражение целого типа (**`char, int`** и производные от них);

```
switch ( i+3 )
{
  case 1: a = b; break;
  case 2: a = c;
}
```

```
switch ( x )
{
  case 1: a = b; break;
  case 1: a = c;
}
```

- нельзя ставить два одинаковых значения;
- **основное** назначение **`switch`** – пропуск нескольких команд.

# Множественный выбор




**Задача.** Ввести символ и определить, принадлежит ли он одной или нескольким основным системам счисления. Если принадлежит, вывести, какой или каким именно.

# Текст программы. Инструкция *switch*

```
#include <stdio.h>
int main()
{
    char symbol;
    printf ("Введите символ\n");
    scanf ("%c", &symbol);
    switch (symbol)
    {
        case '0' : case '1' : printf ("Это двоичная цифра\n");
        case '2' : case '3' : case '4' : case '5' :
        case '6' : case '7' : printf ("Это восьмеричная цифра\n");
        case '8' : case '9' : printf ("Это десятичная цифра\n");
        case 'a' : case 'A' : case 'b' : case 'B' :
        case 'c' : case 'C' : case 'd' : case 'D' :
        case 'e' : case 'E' : case 'f' :
        case 'F' : printf ("Это шестнадцатеричная цифра\n"); break;
        default : printf ("Это не цифра\n");
    }
    return 0;
}
```

```
Введите символ
5
Это восьмеричная цифра
Это десятичная цифра
Это шестнадцатеричная цифра
```

Выйти из switch