

Программирование на языке Паскаль

- | | |
|--|--|
| 1. <u>Введение</u> | 7. <u>Графика</u> |
| 2. <u>Ветвления</u> | 8. <u>Графики функций</u> |
| 3. <u>Сложные условия</u> | 9. <u>Процедуры</u> |
| 4. <u>Циклы</u> | 10. <u>Рекурсия</u> |
| 5. <u>Циклы с условием</u> | 11. <u>Анимация</u> |
| 6. <u>Оператор выбора</u> | 12. <u>Случайные числа</u> |
| | 13. <u>Функции</u> |

Программирование на языке Паскаль

Тема 1. Введение

Алгоритм

Алгоритм — это четко определенный план действий для исполнителя.

Свойства алгоритма

- **дискретность**: состоит из отдельных шагов (команд)
- **понятность**: должен включать только команды, известные исполнителю (входящие в СКИ)
- **определенность**: при одинаковых исходных данных всегда выдает один и тот же результат
- **конечность**: заканчивается за конечное число шагов
- **массовость**: может применяться многократно при различных исходных данных
- **корректность**: дает верное решение при любых допустимых исходных данных

Программа

Программа — это

- алгоритм, записанный на каком-либо языке программирования
- набор команд для компьютера

Команда — это описание действий, которые должен выполнить компьютер.

- откуда взять исходные данные?
- что нужно с ними сделать?

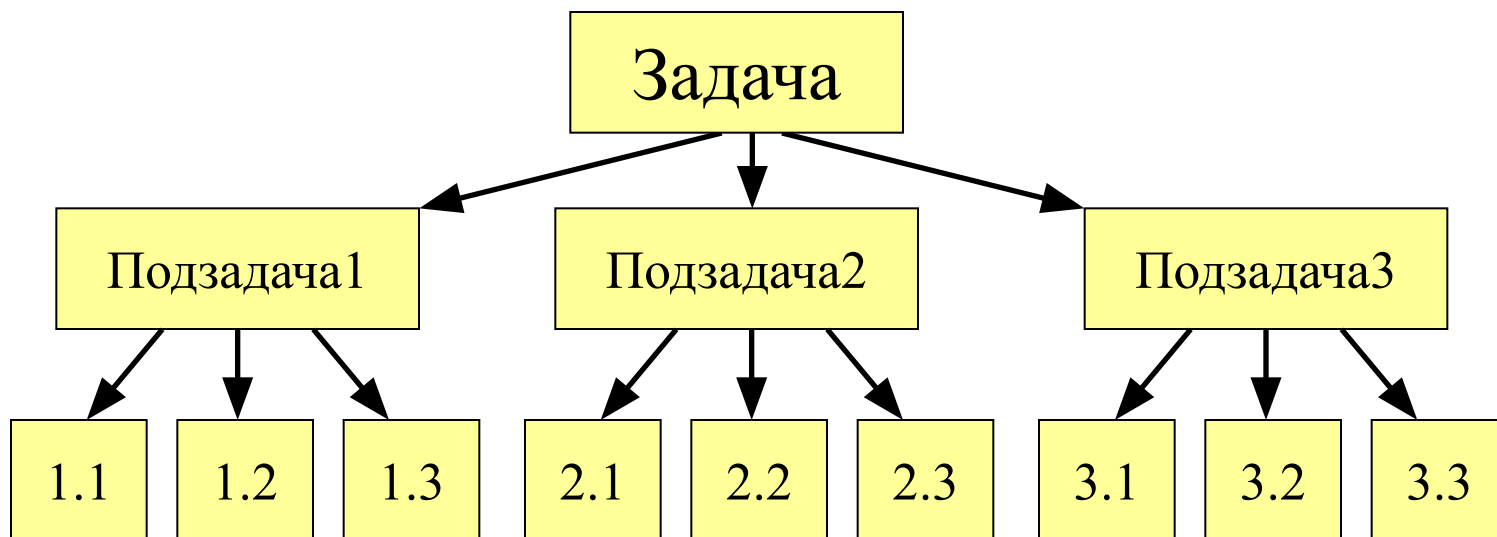
Языки программирования

- **Машинно-ориентированные (низкого уровня)** - каждая команда соответствует одной команде процессора (ассемблер)
- **Языки высокого уровня** – приближены к естественному (английскому) языку, легче воспринимаются человеком, **не зависят от конкретного компьютера**
 - *для обучения*: Бейсик, ЛОГО, Паскаль
 - *профессиональные*: Си, Фортран, Паскаль
 - *для задач искусственного интеллекта*: Пролог, ЛИСП
 - *для Интернета*: JavaScript, Java, Perl, PHP, ASP

Язык Паскаль

1970 – Никлаус Вирт (Швейцария)

- язык для обучения студентов
- разработка программ «сверху вниз»



- разнообразные структуры данных (массивы, структуры, множества)

Из чего состоит программа?

```
program <имя программы>;  
const ...; {константы}  
var ...; {переменные}  
  
{ процедуры и функции }  
begin  
    ... {основная программа}  
end.
```

комментарии в фигурных скобках не
обрабатываются

Из чего состоит программа?

Константа — постоянная величина, имеющая имя.

Переменная — изменяющаяся величина, имеющая имя (ячейка памяти).

Процедура — вспомогательный алгоритм, описывающий некоторые действия (рисование окружности).

Функция — вспомогательный алгоритм для выполнения вычислений (вычисление квадратного корня, **sin**).

Имена программы, констант, переменных

Имена могут включать

- латинские буквы (A-Z)

заглавные и строчные буквы не различаются

- цифры

имя не может начинаться с цифры

- знак подчеркивания _

Имена **НЕ** могут включать

- русские буквы
- пробелы
- скобки, знаки +, =, !, ? и др.

Какие имена правильные??

AXby R&B 4Wheel Вася “PesBarbos” TU154
[QuQu] _ABBA A+B

Константы

const

i2 = 45; { целое число }

pi = 3.14; { вещественное число }

целая и дробная часть отделяются точкой

qq = 'Вася'; { строка символов }

можно использовать русские буквы!

L = True; { логическая величина }

может принимать два значения:

- True (истина, «да»)
- False (ложь, «нет»)

Переменные

Переменная — это величина, имеющая имя, тип и значение.

Значение переменной можно изменять во время работы программы.

Типы переменных:

- integer { целая }
- real { вещественная }
- char { один символ }
- string { символьная строка }
- boolean { логическая }

Объявление переменных (выделение памяти):

```
var a, b: integer;  
    Q: real;  
    s1, s2: string;
```

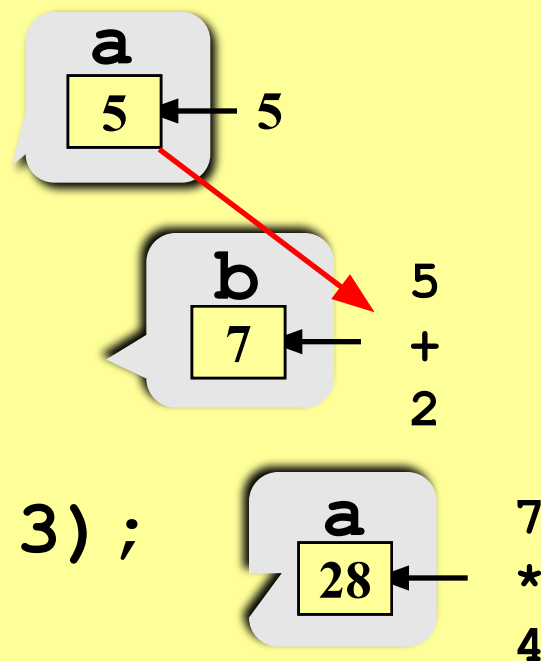
Как изменить значение переменной?

Оператор — это команда языка программирования высокого уровня.

Оператор присваивания служит для изменения значения переменной.

Пример

```
:program qq;  
var a, b: integer;  
begin  
    a := 5;  
    b := a + 2;  
    a := (a + 2) * (b - 3);  
end.
```



Оператор присваивания

Общая структура:

<имя переменной> := <выражение>;

Арифметическое выражение может включать

- КОНСТАНТЫ
- имена переменных
- знаки арифметических операций:

+ **-**

/

div

mod

умножение

деление

деление
нацело

остаток от
деления

- ВЫЗОВЫ функций
- круглые скобки ()

Какие операторы неправильные?

```
program qq;  
var a, b: integer;  
    x, y: real;  
begin  
    a := 5;  
    10 := x;  
    y := 7,8;  
    b := 2.5;  
    x := 2*(a + y) ;  
    a := b + x;  
end.
```

имя переменной должно быть
слева от знака :=

целая и дробная часть
отделяются **точкой**

нельзя записывать
вещественное значение в целую
переменную

Ручная прокрутка программы

```
program qq;  
var a, b: integer;  
begin  
    a := 5;  
    b := a + 2;  
    a := (a + 2) * (b - 3);  
    b := a div 5;  
    a := a mod b;  
    a := a + 1;  
    b := (a + 14) mod 7;  
end.
```

[illegible]

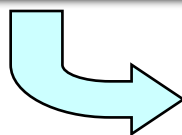
Порядок выполнения операций

- вычисление выражений в скобках
- умножение, деление, **div**, **mod** слева направо
- сложение и вычитание слева направо

2 3 5 4 1 7 8 6

z := (5*a*c+3*(c-d)) / a*(b-c) / b;

$$x = \frac{a^2 + 5c^2 - d(a+b)}{(c+d)(d-2a)}$$



$$z = \frac{5ac + 3(c-d)}{ab} (b-c)$$

2 6 3 4 7 5 1 12 8 11 10 9

x := (a*a+5*c*c-d*(a+b)) / ((c+d)*(d-2*a)) ;

Сложение двух чисел

Задача. Ввести два целых числа и вывести на экран их сумму.

Простейшее решение:

```
program qq;  
var a, b, c: integer;  
begin  
    read ( a, b );  
    c := a + b;  
    writeln ( c );  
end.
```

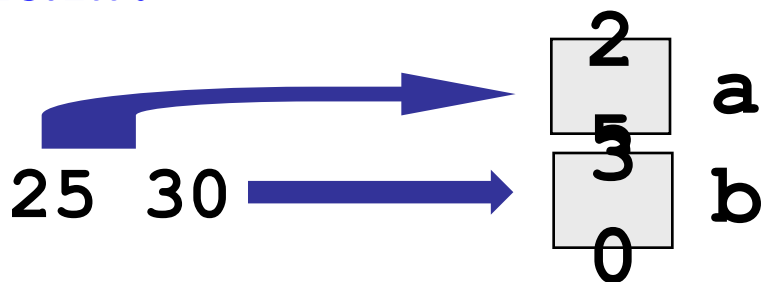
Оператор ввода

`read (a) ;` { ввод значения
 переменной a }

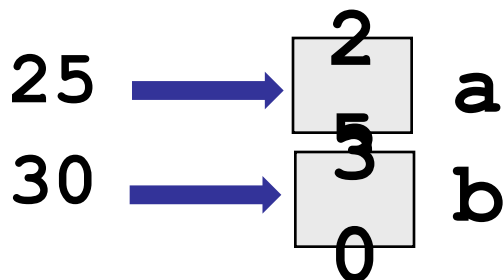
`read (a , b) ;` { ввод значений
 переменных a и b }

Как вводить два числа?

через пробел:



через *Enter*:



Оператор вывода

`write (a);` { вывод значения
 переменной `a` }

`writeln (a);` { вывод значения
 переменной `a` и переход
 на новую строку }

`writeln ('Привет!');` { вывод текста }

`writeln ('Ответ: ', c);` { вывод
 текста и значения переменной `c` }

`writeln (a, '+', b, '=', c);`

Форматы вывода

```

program qq;
var i: integer;
    x: real;
begin
  i := 15;
  writeln ( '>', i, ' ');
  writeln ( '>', i:5, '<' );
  x := 12.345678;
  writeln ( '>', x, '<' );
  writeln ( '>', x:10, '<' );
  writeln ( '>', x:7:2, '<' );
end.

```

ВСЕГО
СИМВОЛОВ

```

>15<
>   15<

>1.234568E+001<
> 1.23E+001<
>  12.35<

```

ВСЕГО
СИМВОЛОВ

В дробной
части

Полное решение

```
program qq;  
var a, b, c: integer;  
begin  
    writeln('Введите два целых числа');  
    read ( a, b );  
    c := a + b;  
    writeln ( a, '+', b, '=', c );  
end.
```

Протокол:

ЭТО ВЫВОДИТ КОМПЬЮТЕР

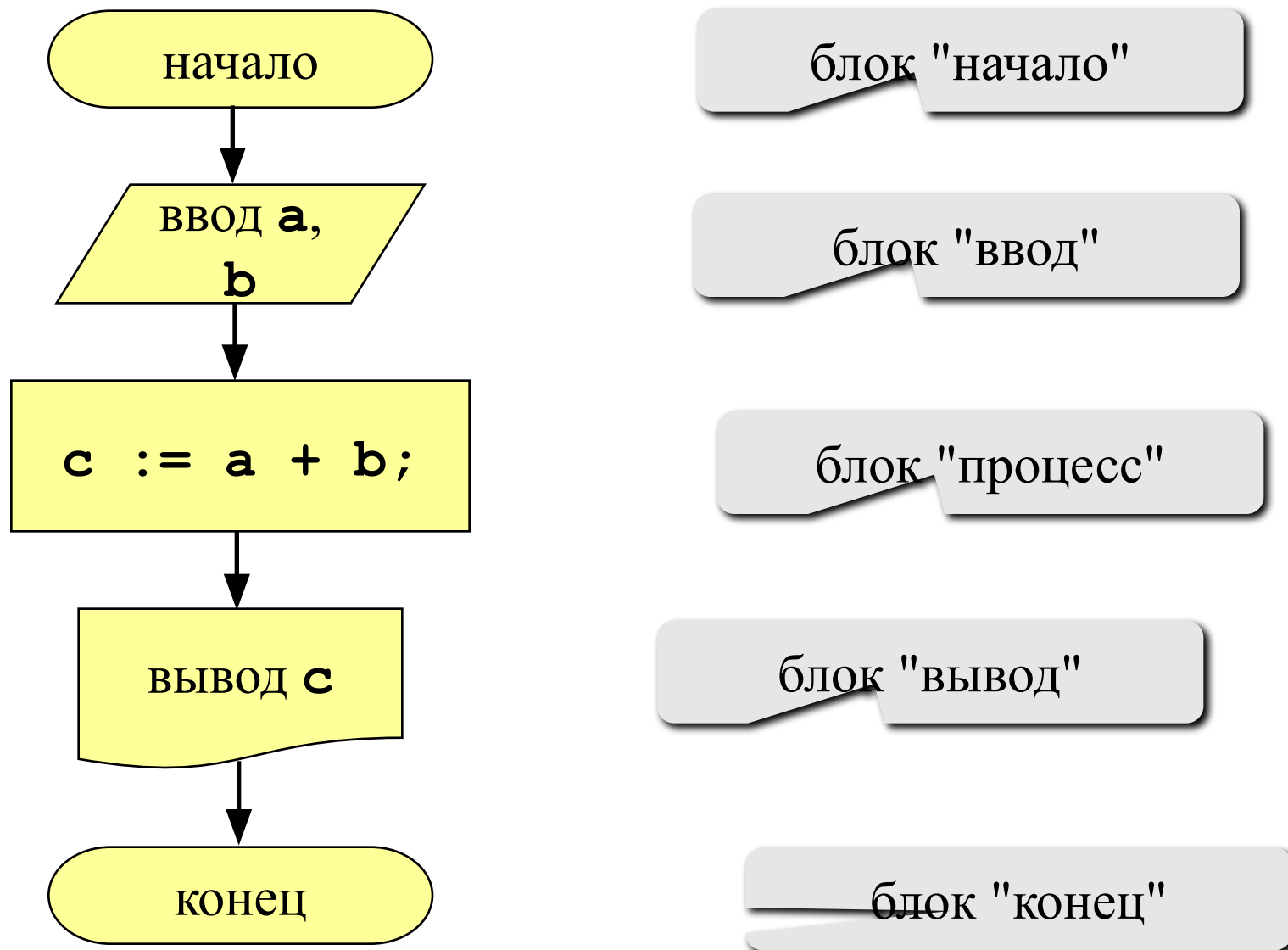
Введите два целых числа

25 30

ЭТО ВВОДИТ ПОЛЬЗОВАТЕЛЬ

25+30=55

Блок-схема линейного алгоритма



Задания

"4": Ввести три числа, найти их сумму и произведение.

Пример:

Введите три числа:

4 5 7

$$4+5+7=16$$

$$4*5*7=140$$

"5": Ввести три числа, найти их сумму, произведение и среднее арифметическое.

Пример:

Введите три числа:

4 5 7

$$4+5+7=16$$

$$4*5*7=140$$

$$(4+5+7) / 3 = 5.33$$

Программирование на языке Паскаль

Тема 2. Ветвления

Разветвляющиеся алгоритмы

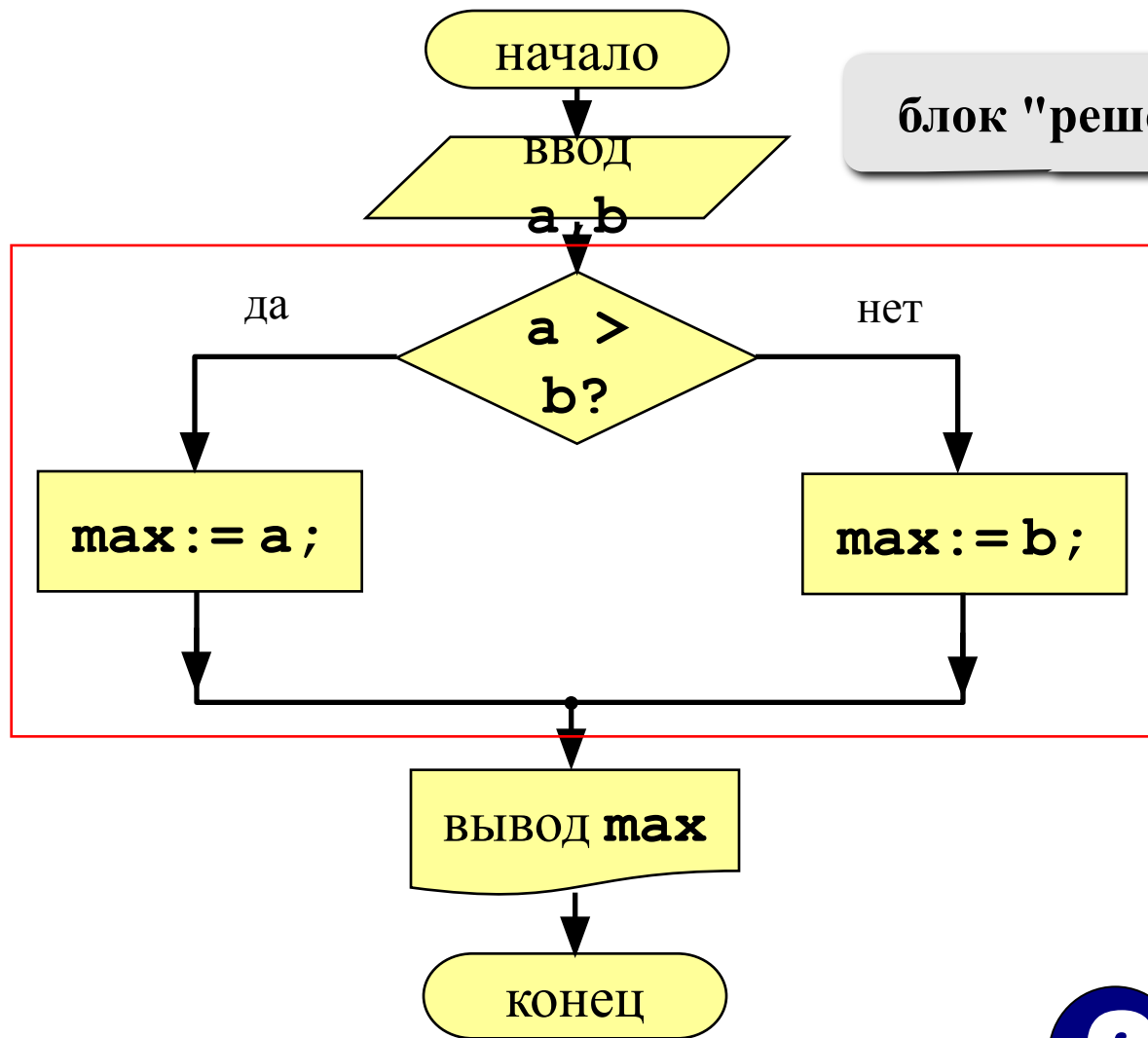
Задача. Ввести два целых числа и вывести на экран наибольшее из них.

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

Особенность: действия исполнителя зависят от некоторых условий (*если ... иначе ...*).

Алгоритмы, в которых последовательность шагов зависит от выполнения некоторых условий, называются
разветвляющимися.

Вариант 1. Блок-схема



блок "решение"

полная форма
ветвления



Если $a = b$?

Вариант 1. Программа

```
program qq;  
var a, b, max: integer;  
begin  
    writeln('Введите два целых числа');  
    read ( a, b );  
    if a > b then begin  
        max := a;  
    end  
    else begin  
        max := b;  
    end;  
    writeln ('Наибольшее число ', max);  
end.
```

полная форма
условного
оператора

Условный оператор

```
if <условие> then begin
    {что делать, если условие верно}
end
else begin
    {что делать, если условие неверно}
end;
```

Особенности:

- перед **else** НЕ ставится точка с запятой
- вторая часть (**else** ...) может отсутствовать (неполная форма)
- если в блоке один оператор, можно убрать слова **begin** и **end**

Что неправильно?

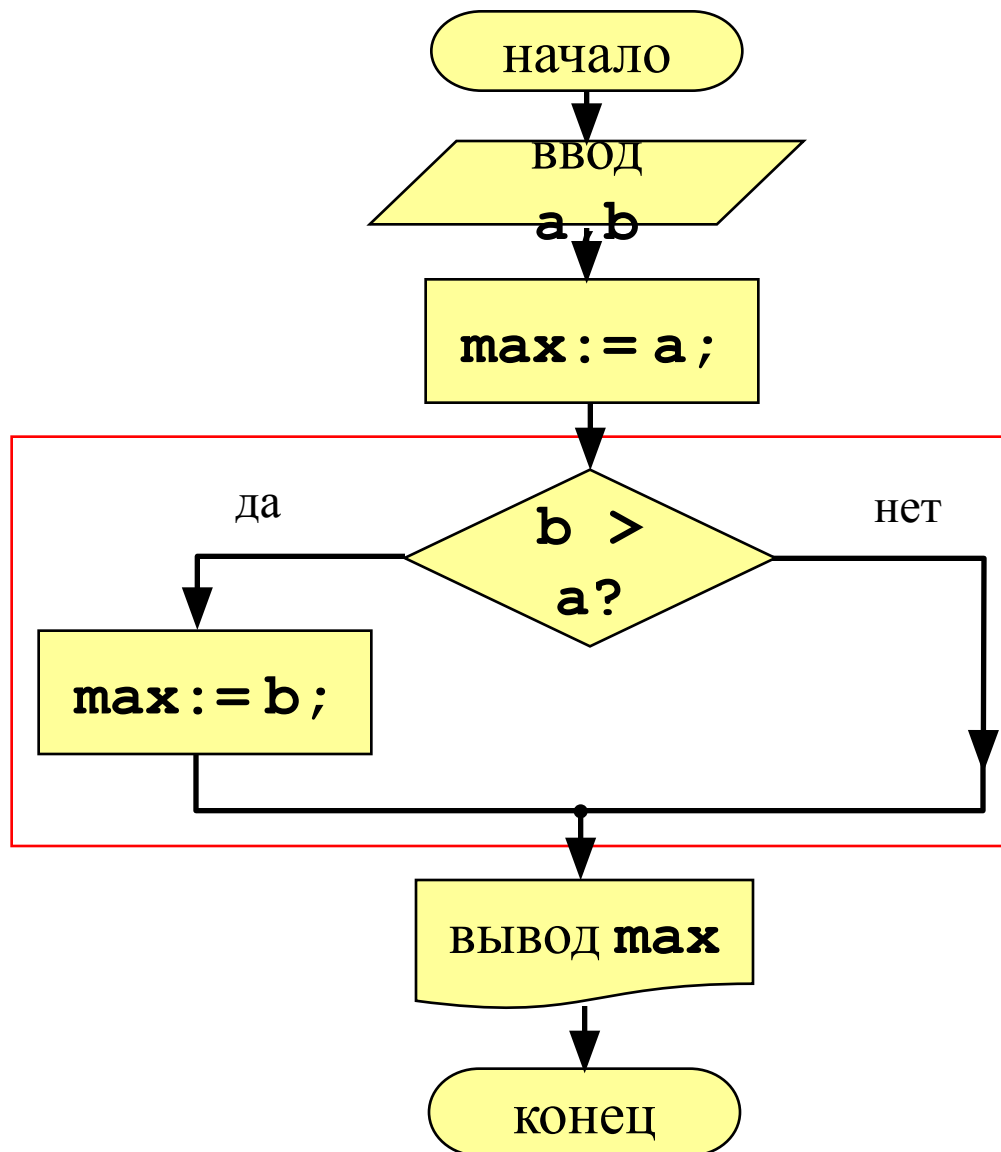
```
if a > b then begin
  a := b;
end
else begin
  b := a;
end;
```

```
if a > b then begin
  a := b; end
else begin
  b := a;
end;
```

```
if a > b then begin
  a := b;
end
else begin
  b := a;
end;
```

```
if a > b then begin
  a := b;
end
else begin
  b := a;
end;
```

Вариант 2. Блок-схема



неполная форма
ветвления

Вариант 2. Программа

```
program qq;  
var a, b, max: integer;  
begin  
    writeln('Введите два целых числа');  
    read ( a, b );  
    max := a;  
    if b > a then  
        max := b;  
    writeln ('Наибольшее число ', max);  
end.
```

неполная форма
условного
оператора

Вариант 2Б. Программа

```
program qq;  
var a, b, max: integer;  
begin  
    writeln('Введите два целых числа');  
    read ( a, b );  
    max := b;  
    if a > b then  
        max := a;  
    writeln ('Наибольшее число ', max);  
end.
```


Что неправильно?

```
if a > b then  
    a := b  
else b := a;
```

```
if a > b then begin  
    a := b;  
end  
else b := a;
```

```
if a > b then  
    a := b  
else b := a;
```

```
if b >= a then  
    b := a;
```

Задания

"4": Ввести три числа и найти наибольшее из них.

Пример:

Введите три числа:

4 15 9

Наибольшее число 15

"5": Ввести пять чисел и найти наибольшее из них.

Пример:

Введите пять чисел:

4 15 9 56 4

Наибольшее число 56