

# Программирование на алгоритмическом языке

- § 54. Алгоритм и его свойства
- § 55. Простейшие программы
- § 56. Вычисления
- § 57. Ветвления
- § 58. Циклические алгоритмы
- § 59. Процедуры
- § 60. Функции
- § 61. Рекурсия

# Программирование на алгоритмическом языке

## § 54. Алгоритм и его свойства

# Что такое алгоритм?

**Алгоритм** — это точное описание порядка действий, которые должен выполнить исполнитель для решения задачи за конечное время.

**Исполнитель** – это устройство или одушевленное существо (человек), способное понять и выполнить команды, составляющие алгоритм.

**Формальные исполнители:** не понимают (и не могут понять) смысл команд.



Мухаммед ал-Хорезми  
(ок. 783—ок. 850 гг.)

# Свойства алгоритма

**Дискретность** — алгоритм состоит из отдельных команд, каждая из которых выполняется за конечное время.

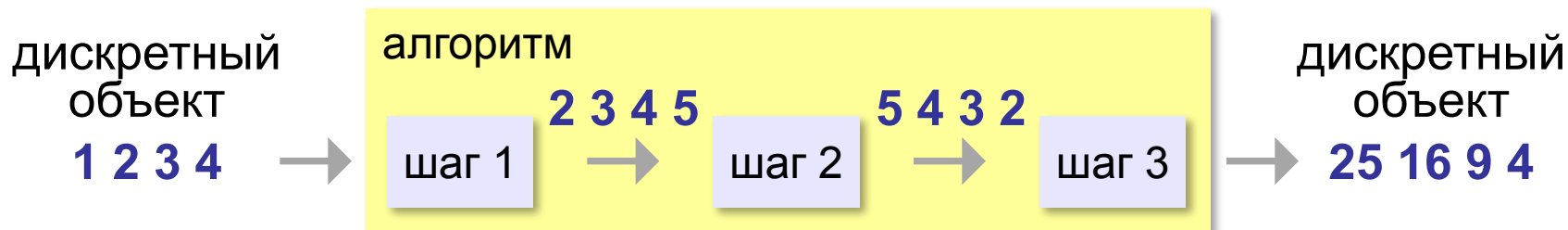
**Детерминированность** (определённость) — при каждом запуске алгоритма с одними и теми же исходными данными получается один и тот же результат.

**Понятность** — алгоритм содержит только команды, входящие в **систему команд исполнителя**.

**Конечность** (результативность) — для корректного набора данных алгоритм должен завершаться через конечное время.

**Корректность** — для допустимых исходных данных алгоритм должен приводить к правильному результату.

# Как работает алгоритм?



- получает на вход дискретный объект
- в результате строит другой дискретный объект (или выдаёт сообщение об ошибке)
- обрабатывает объект по шагам
- на каждом шаге получается новый дискретный объект

# Способы записи алгоритмов

---

- **естественный язык**

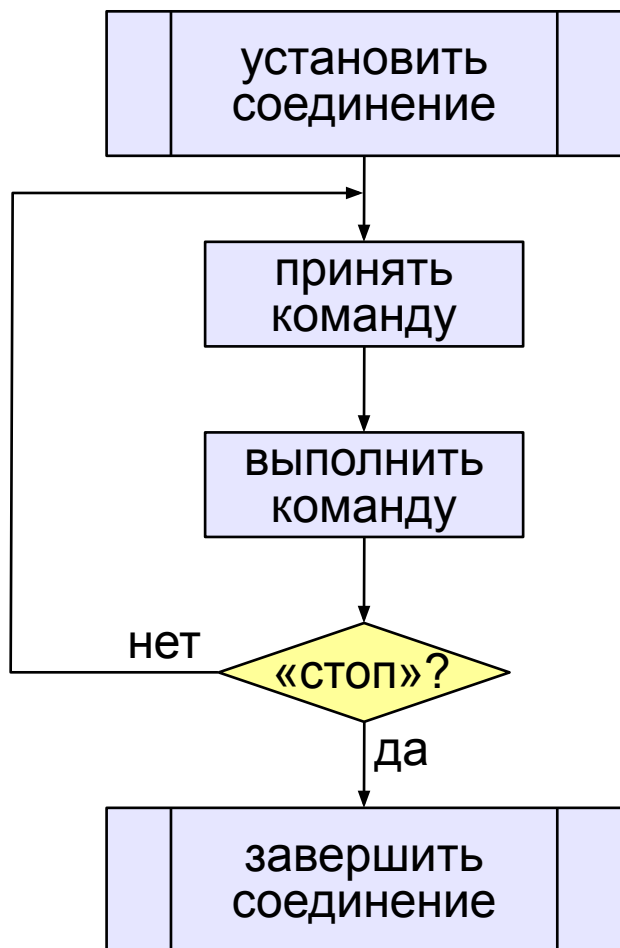
```
установить соединение  
пока не принята команда «стоп»  
    принять команду  
    выполнить команду  
завершить сеанс связи
```

- **псевдокод**

```
установить соединение  
нц  
    принять команду  
    выполнить команду  
кц_при команда = 'stop'  
завершить сеанс связи
```

# Способы записи алгоритмов

- блок-схема



- программа

```
установитьСоединение
```

```
нц
```

```
cmd := получитьКоманду
```

```
выполнитьКоманду (cmd)
```

```
кц_при cmd = 'stop'
```

```
закретьСоединение
```

# Программирование на алгоритмическом языке

## § 55. Простейшие программы



# Простейшая программа

---

название алгоритма

алг **Куку**

нач | *начало программы*

| *тело программы*

кон | *конец программы*

комментарии после |  
не обрабатываются




Что делает эта программа?

# Вывод на экран

алг Куку

нач

- ▶ вывод '2+' 
- ▶ вывод '2=?', нс
- ▶ вывод 'Ответ: 4'

кон

Протокол:

2+

Ответ: 4

# Задания

---

«В»: Вывести на экран текст «лесенкой»

Вася

пошел

гулять

«С»: Вывести на экран рисунок из букв

```
Ж
ЖЖЖ
ЖЖЖЖЖ
ЖЖЖЖЖЖЖ
НН НН
ZZZZZ
```

# Сложение чисел

**Задача.** Ввести с клавиатуры два числа и найти их сумму.

**Протокол:**

Введите два целых числа

компьютер

25 30

пользователь

25+30=55

компьютер считает сам!

?

1. Как ввести числа в память?
2. Где хранить введенные числа?
3. Как вычислить?
4. Как вывести результат?

# Сумма: псевдокод

алг **Сумма**

нач

| ввести два числа

| вычислить их сумму

| вывести сумму на экран

кон

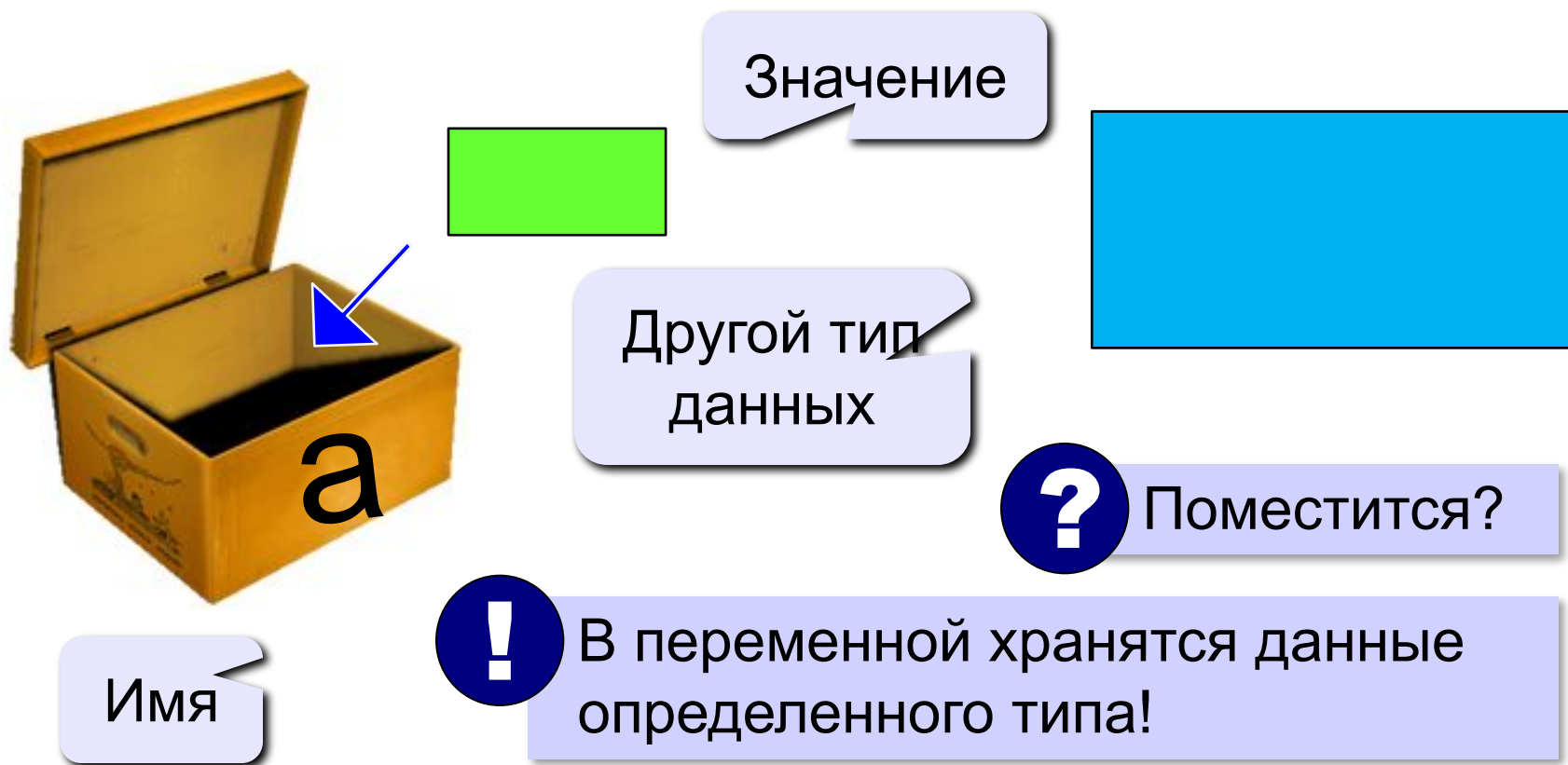
**Псевдокод** – алгоритм на русском языке с элементами языка программирования.



Компьютер не может исполнить псевдокод!

# Переменные

**Переменная** – это величина, имеющая имя, тип и значение. Значение переменной можно изменять во время работы программы.



# Имена переменных

**МОЖНО** использовать

- латинские буквы (A-Z), русские буквы (А-Я)

заглавные и строчные буквы **различаются**

- цифры

имя не может начинаться с цифры

- знак подчеркивания \_

**НЕЛЬЗЯ** использовать

~~• скобки~~

~~• знаки +, =, !, ? и др.~~

Какие имена правильные?

**AXby R&B 4Wheel Вася “PesBarbos”**

**TU154 [QuQu] \_ABBA A+B**

# Объявление переменных

---

## Типы переменных:

- цел | целая
- вещ | вещественная
- и другие...

## Объявление переменных:

тип – целые

список имен  
переменных

выделение  
места в памяти

**цел** a, b, c



# Тип переменной

---

- область допустимых значений
- допустимые операции
- объём памяти
- формат хранения данных
- для предотвращения случайных ошибок

## Начальные значения:

**цел**  $a$ ,  $b = 1$ ,  $c = 55$



Что в переменной  $a$ ?

# Как записать значение в переменную?

оператор  
присваивания

`a := 5`



При записи нового значения старое стирается!

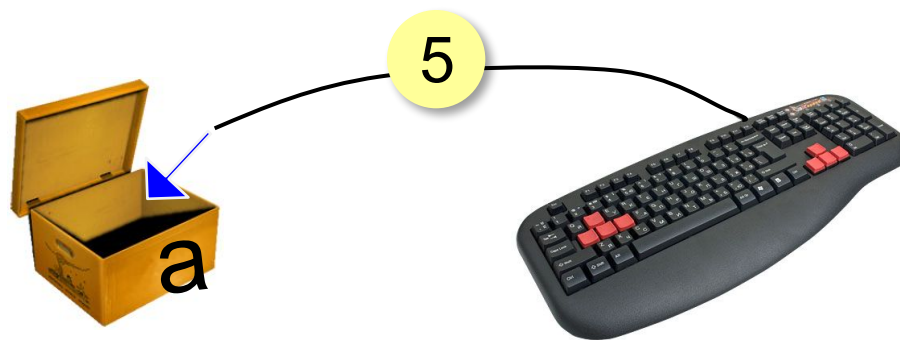
**Оператор** – это команда языка программирования (инструкция).

**Оператор присваивания** – это команда для записи нового значения в переменную.

# Ввод значения с клавиатуры

оператор  
ввода

**ВВОД а**

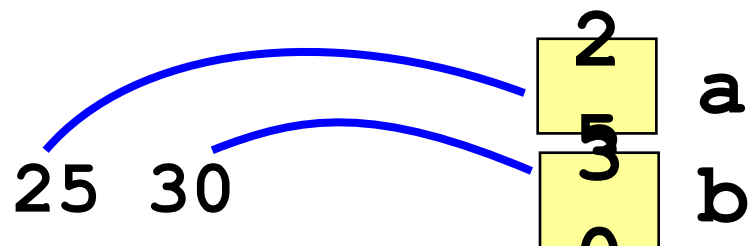


1. Программа ждет, пока пользователь введет значение и нажмет *Enter*.
2. Введенное значение записывается в переменную **а**.

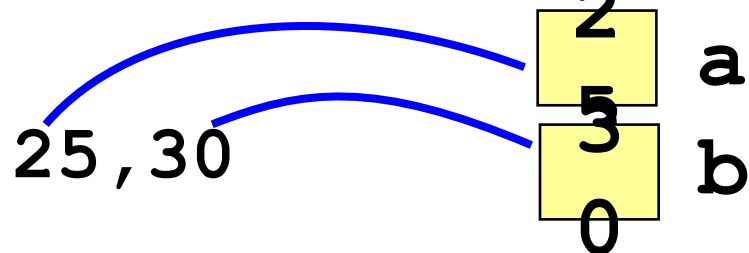
# Ввод значений переменных

ввод a, b

через пробел:



через запятую:



# Изменение значений переменной

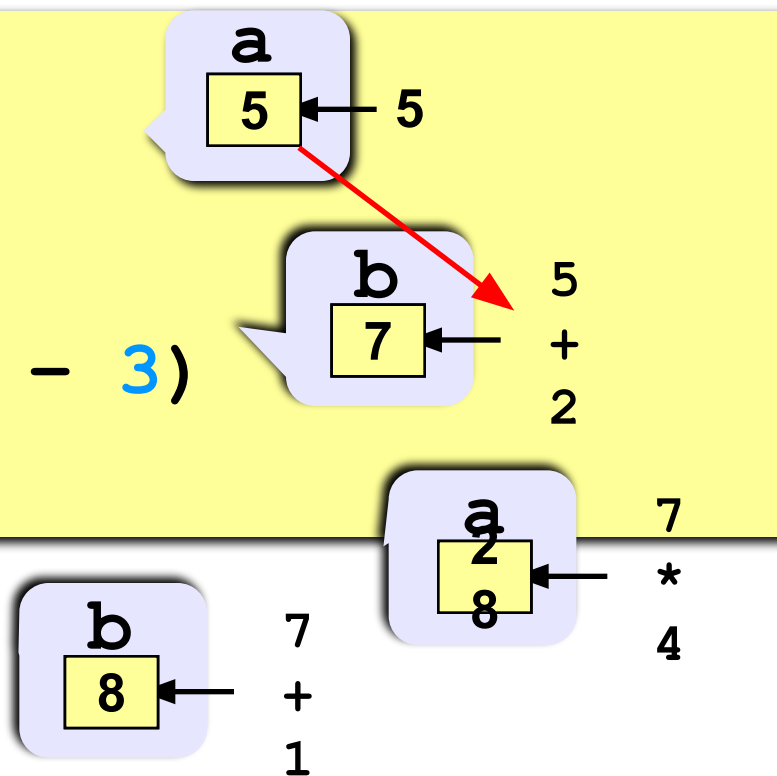
**цел** a, b

a := 5

b := a + 2

a := (a + 2) \* (b - 3)

b := b + 1



# Вывод данных

---

вывод **a**

| вывод значения  
| переменной **a**

вывод **a, nс**

| вывод значения  
| переменной **a** и **переход**  
| **на новую строку**

вывод **'Привет!'**

| вывод текста

вывод **'Ответ: ', c**

| вывод текста и значения переменной **c**

вывод **a, '+', b, '=', c**

# Сложение чисел: простое решение

---

алг **Сумма**

нач

**цел** a, b, c

ввод a, b

c := a + b

вывод c

кон



Что плохо?

# Сложение чисел: полное решение

алг **Сумма**

нач

**цел** a, b, c

подсказка

вывод 'Введите два целых числа'

ввод a, b

c := a + b

вывод a, '+', b, '=', c

кон

**Протокол:**

КОМПЬЮТЕР

**Введите два целых числа**

**25 30**

ПОЛЬЗОВАТЕЛЬ

**25+30=55**



# Снова про оператор вывода

## Вычисление выражений:

вывод a, '+', b, '=', a+b

## Форматный вывод (КуМир 2.0+):

a := 123

вывод a: 5

123  
5 знаков

# Программирование на алгоритмическом языке

## § 56. Вычисления

# Типы данных

---

- **цел** | **целое**
- **вещ** | **вещественное**
- **лог** | **логические значения**
- **сим** | **символ**
- **лит** | **литерная переменная**

# Арифметическое выражения

3    2    1    4    5    6

**a := (c + b \* 5 \*\* 3 - 1) / 2 \* d**

**Приоритет** (*старшинство*):

- 1) скобки
- 2) возведение в степень (\*\*)
- 3) умножение и деление
- 4) сложение и вычитание

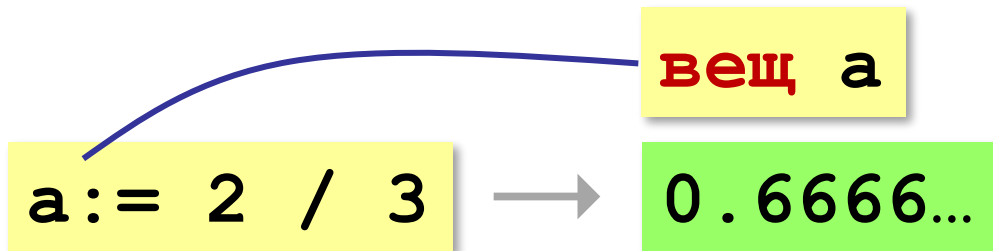
$$a = \frac{c + b \cdot 5^3 - 1}{2} \cdot d$$

**Вывод** 5 \*\* 3

## Деление, `div`, `mod`

---

Результат деления «/» – вещественное число:



`div` – деление нацело (остаток отбрасывается)

`mod` – остаток от деления

```
цел a, b, d
d := 85
b := div(d, 10)
a := mod(d, 10)
```

## div и mod для отрицательных чисел

Вывод `div(-7, 2)`, нс  
 Вывод `mod(-7, 2)`

$$\begin{array}{l} -4 \\ 1 \end{array} \quad -7 = (-4) * 2 + 1$$

остаток  $\geq 0$



В других языках не так!

$$7 = 3 * 2 + 1$$

$$-7 = (-3) * 2 + (-1)$$

# Вещественные числа



Целая и дробная части числа разделяются точкой!

```
вещ x
x := 123.456
```

Форматный вывод (КуМир 2.0+):

```
a := 1
вывод a/3
вывод a/3:7:3
```

```
0.33333333
  0.333
```

всего знаков

в дробной части

# Вещественные числа

## Экспоненциальный формат:

**цел** a = 1

**вывод** a/30000, нс

**вещ** b = 12345678

**вывод** b

$3,333333 \cdot 10^{-5}$

3.333333e-05

1.234568e+07

$1,234568 \cdot 10^7$



# Стандартные функции

---

- abs** (x) — модуль
- sqrt** (x) — квадратный корень
- sin** (x) — синус угла, заданного **в радианах**
- cos** (x) — косинус угла, заданного **в радианах**
- exp** (x) — экспонента  $e^x$
- ln** (x) — натуральный логарифм
- int** (x) — целая часть числа

# Случайные числа

## Случайно...

- встретить друга на улице
- разбить тарелку
- найти 10 рублей
- выиграть в лотерею

## Случайный выбор:

- жеребьевка на соревнованиях
- выигравшие номера в лотерее

## Как получить случайность?



# Случайные числа на компьютере

## Электронный генератор



- нужно специальное устройство
- нельзя воспроизвести результаты

**Псевдослучайные числа** – обладают свойствами случайных чисел, но каждое следующее число вычисляется по заданной формуле.

## Метод середины квадрата (Дж. фон Нейман)

зерно

564321

в квадрате

- малый период  
(последовательность повторяется через  $10^6$  чисел)

318458191041

209938992481

# Генератор случайных чисел

## Вещественные числа в интервале [0,10):

**вещ** X, Y

X := `rand(0, 10)` | интервал от 0 до 10 (<10)

Y := `rand(0, 10)` | это уже другое число!

англ. *random* – случайный

## Целые числа в интервале [0,10]:

**цел** K, L

K := `irand(0, 10)` | интервал от 0 до 10 (<=10)

L := `irand(0, 10)` | это уже другое число!

англ. *Integer* – целый

# Задачи

---

«А»: Ввести с клавиатуры три целых числа, найти их сумму, произведение и среднее арифметическое.

**Пример:**

Введите три целых числа:

5 7 8

$$5+7+8=20$$

$$5*7*8=280$$

$$(5+7+8)/3=6.667$$

«В»: Ввести с клавиатуры координаты двух точек (А и В) на плоскости (вещественные числа). Вычислить длину отрезка АВ.

**Пример:**

Введите координаты точки А:

5.5 3.5

Введите координаты точки В:

1.5 2

$$\text{Длина отрезка АВ} = 4.272$$

# Задачи

---

**«С»:** Получить случайное трехзначное число и вывести через запятую его отдельные цифры.

**Пример:**

Получено число 123.

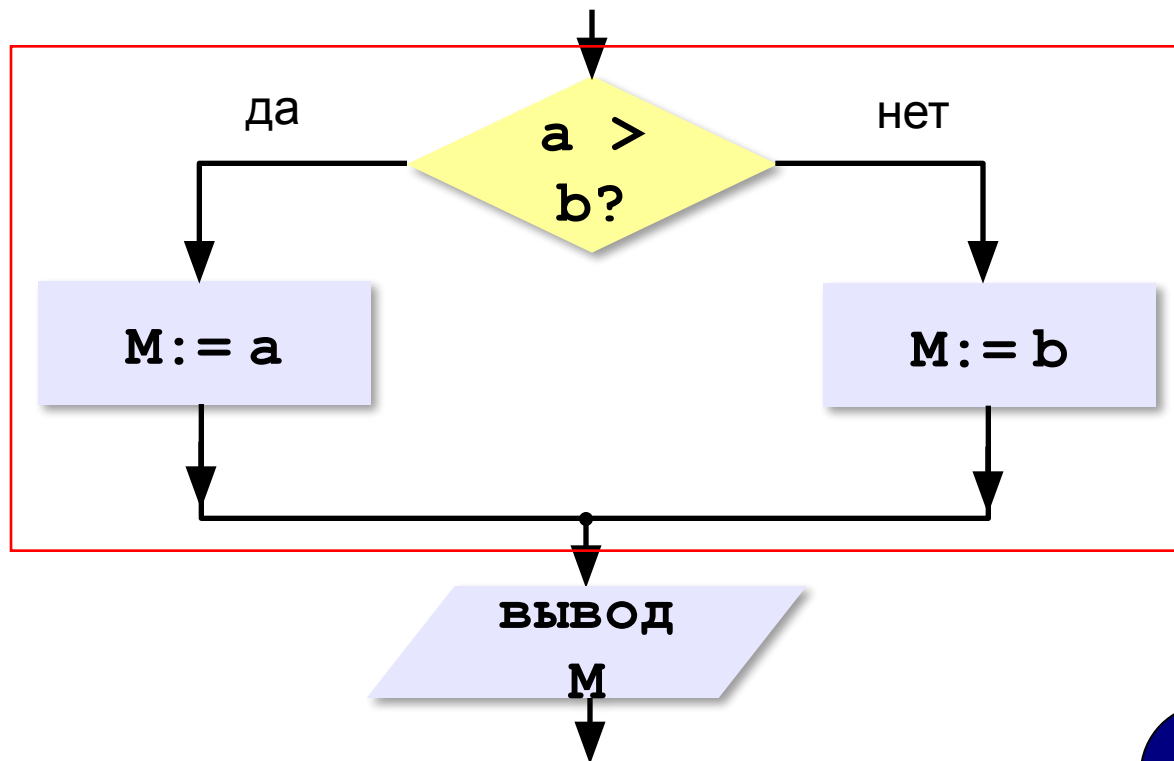
Его цифры 1, 2, 3.

# Программирование на алгоритмическом языке

## § 57. Ветвления

# Условный оператор

Задача: **изменить порядок действий** в зависимости от выполнения некоторого условия.



полная  
форма  
ветвления

? Если  $a = b$ ?



# Условный оператор: полная форма

---

## Полная форма:

**если  $a > b$  то**

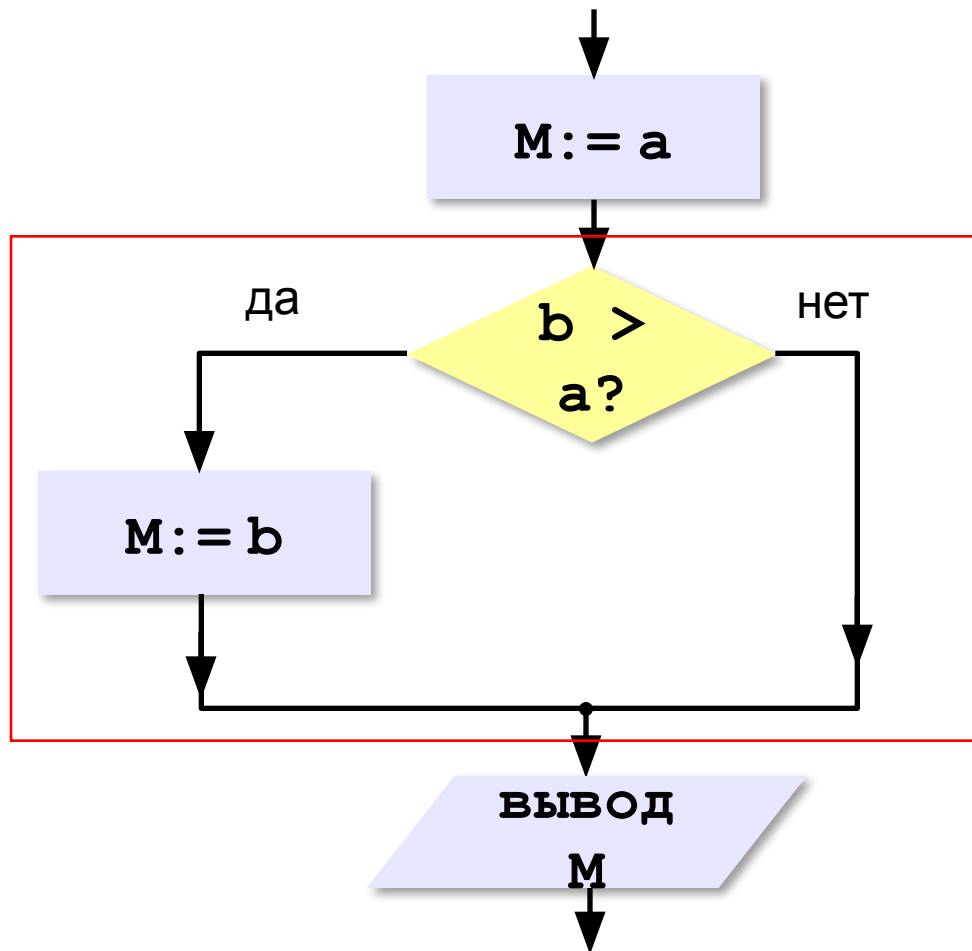
**$M := a$**

**иначе**

**$M := b$**

**все**

# Условный оператор: неполная форма



**M := a**  
**если  $b > a$  то**  
**M := b**  
**все**

неполная  
форма  
ветвления

# Условный оператор

```
если  $a > b$  то
```

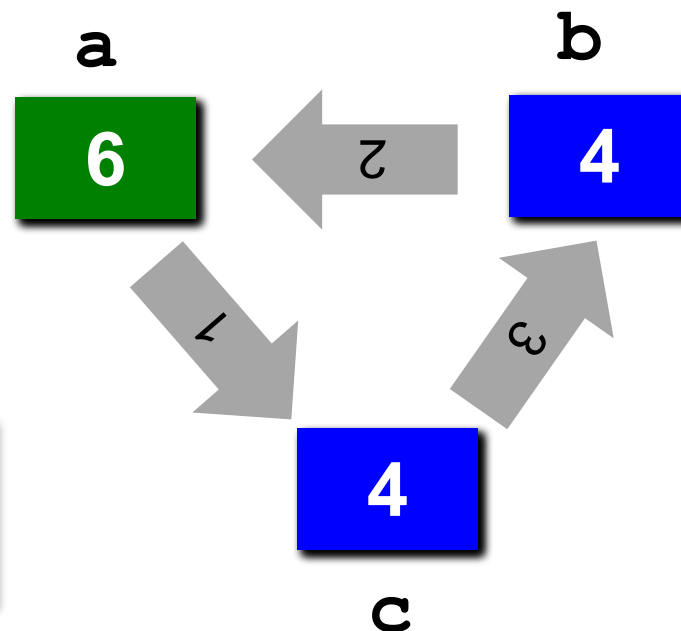
```
   $c := a$ 
```

```
   $a := b$ 
```

```
   $b := c$ 
```

```
все
```

**?** Что делает?



**?** Можно ли обойтись без переменной  $c$ ?

# Знаки отношений

---

**>** **<** больше, меньше

**>=** больше или равно

**<=** меньше или равно

**=** равно

**<>** не равно

# Вложенные условные операторы

Задача: в переменных **a** и **b** записаны возрасты Андрея и Бориса. Кто из них старше?



Сколько вариантов?

если  $a > b$  то

    вывод 'Андрей старше'

иначе

    если  $a = b$  то

        вывод 'Одного возраста'

    иначе

        вывод 'Борис старше'

все

все



Зачем нужен?

вложенный  
условный оператор

# Задачи

---

**«А»:** Ввести три целых числа, найти максимальное из них.

**Пример:**

Введите три целых числа:

1 5 4

Максимальное число 5

**«В»:** Ввести пять целых чисел, найти максимальное из них.

**Пример:**

Введите пять целых чисел:

1 5 4 3 2

Максимальное число 5

# Задачи

---

**«С»:** Ввести последовательно возраст Антона, Бориса и Виктора. Определить, кто из них старше.

**Пример:**

Возраст Антона: 15

Возраст Бориса: 17

Возраст Виктора: 16

Ответ: Борис старше всех.

**Пример:**

Возраст Антона: 17

Возраст Бориса: 17

Возраст Виктора: 16

Ответ: Антон и Борис старше Виктора.

# Сложные условия

Задача: набор сотрудников в возрасте **25-40 лет**  
(включительно).

сложное условие

```
если  $v \geq 25$  и  $v \leq 40$  то  
    вывод 'подходит'  
иначе  
    вывод 'не подходит'  
все
```

и

или

не

## Приоритет :

- 1) отношения ( $<$ ,  $>$ ,  $\leq$ ,  $\geq$ ,  $=$ ,  $\langle \rangle$ )
- 2) не
- 3) и
- 4) или



# Задачи

---

**«А»:** Напишите программу, которая получает три числа и выводит количество одинаковых чисел в этой цепочке.

**Пример:**

Введите три числа:

5 5 5

Все числа одинаковые.

**Пример:**

Введите три числа:

5 7 5

Два числа одинаковые.

**Пример:**

Введите три числа:

5 7 8

Нет одинаковых чисел.

# Задачи

---

**«В»:** Напишите программу, которая получает номер месяца и выводит соответствующее ему время года или сообщение об ошибке.

**Пример:**

**Введите номер месяца :**

**5**

**Весна .**

**Пример:**

**Введите номер месяца :**

**15**

**Неверный номер месяца .**

# Задачи

---

**«С»:** Напишите программу, которая получает возраст человека (целое число, не превышающее 120) и выводит этот возраст со словом «год», «года» или «лет». Например, «21 год», «22 года», «25 лет».

**Пример:**

Введите возраст: **18**

Вам 18 лет.

**Пример:**

Введите возраст: **21**

Вам 21 год.

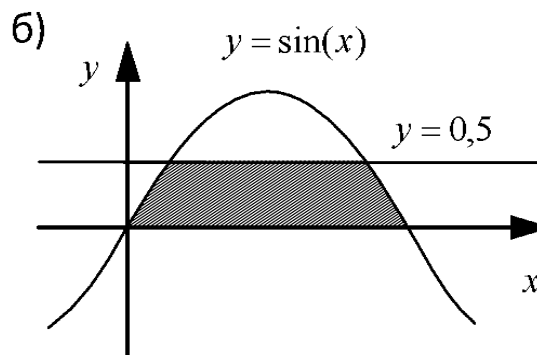
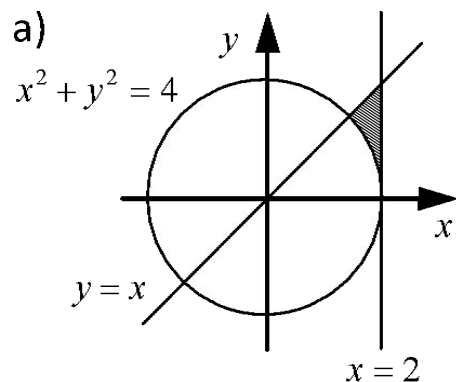
**Пример:**

Введите возраст: **22**

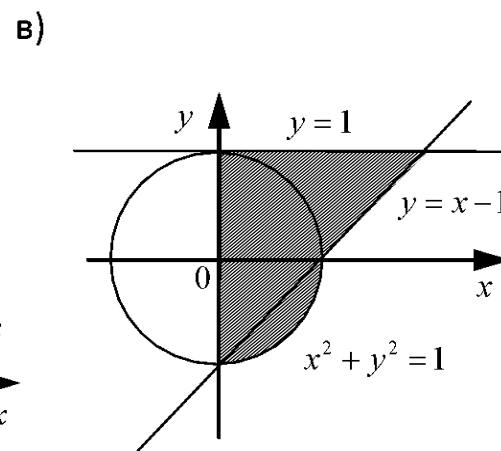
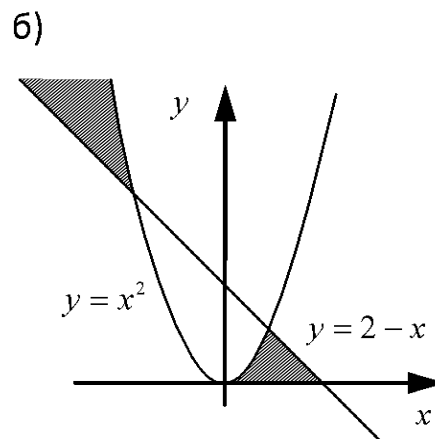
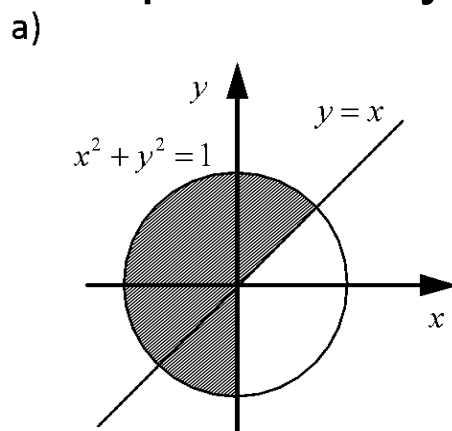
Вам 22 года.

# Задачи

«А»: Напишите условие, которое определяет заштрихованную область.

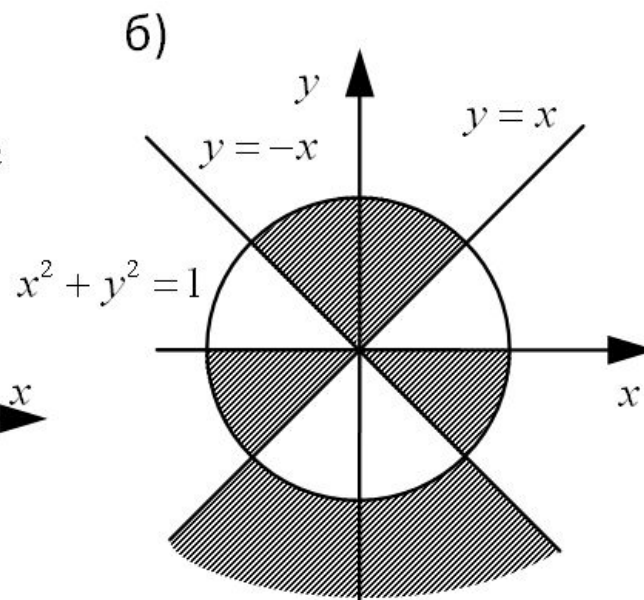
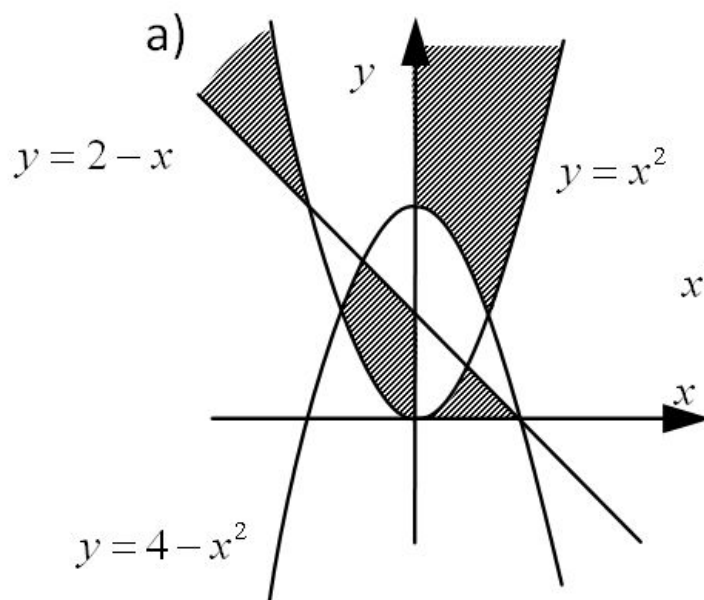


«В»: Напишите условие, которое определяет заштрихованную область.



# Задачи

«С»: Напишите условие, которое определяет заштрихованную область.



# Множественный выбор

```
если m = 1 то  
    вывод 'январь' все  
если m = 2 то  
    вывод 'февраль' все  
...  
если m = 12 то  
    вывод 'декабрь' все
```

```
выбор  
    при m = 1:   вывод 'январь'  
    при m = 2:   вывод 'февраль'  
    ...  
    при m = 12:  вывод 'декабрь'  
иначе           вывод 'ошибка'  
все
```

# Множественный выбор

---

## Знак числа:

выбор

при  $x < 0$ :  $\text{sgn} := -1$

при  $x = 0$ :  $\text{sgn} := 0$

при  $x > 0$ :  $\text{sgn} := 1$

все

любое условие

# Множественный выбор

**СИМ** с

...

**выбор**

при с = 'а':

**ВЫВОД** 'антилопа', нс

**ВЫВОД** 'Анапа'

...

при с = 'я':

**ВЫВОД** 'ягуар', нс

**ВЫВОД** 'Якутск'

**иначе**

**ВЫВОД** 'ошибка'

**все**

несколько  
операторов в  
блоке



# Программирование на алгоритмическом языке

## **§ 58. Циклические алгоритмы**

# Что такое цикл?

**Цикл** – это многократное выполнение одинаковых действий.

## Два вида циклов:

- цикл с **известным** числом шагов (сделать 10 раз)
- цикл с **неизвестным** числом шагов (делать, пока не надоест)

*Задача.* Вывести на экран 10 раз слово «Привет».



Можно ли решить известными методами?

# Повторения в программе

---

вывод 'Привет' , нс

вывод 'Привет' , нс

вывод 'Привет' , нс

...

вывод 'Привет' , нс



Что плохо?

# Повторения в программе

начало цикла

```
алг Привет
```

```
нач
```

тело цикла

```
нц 10 раз
```

```
вывод 'Привет!', нс
```

конец цикла

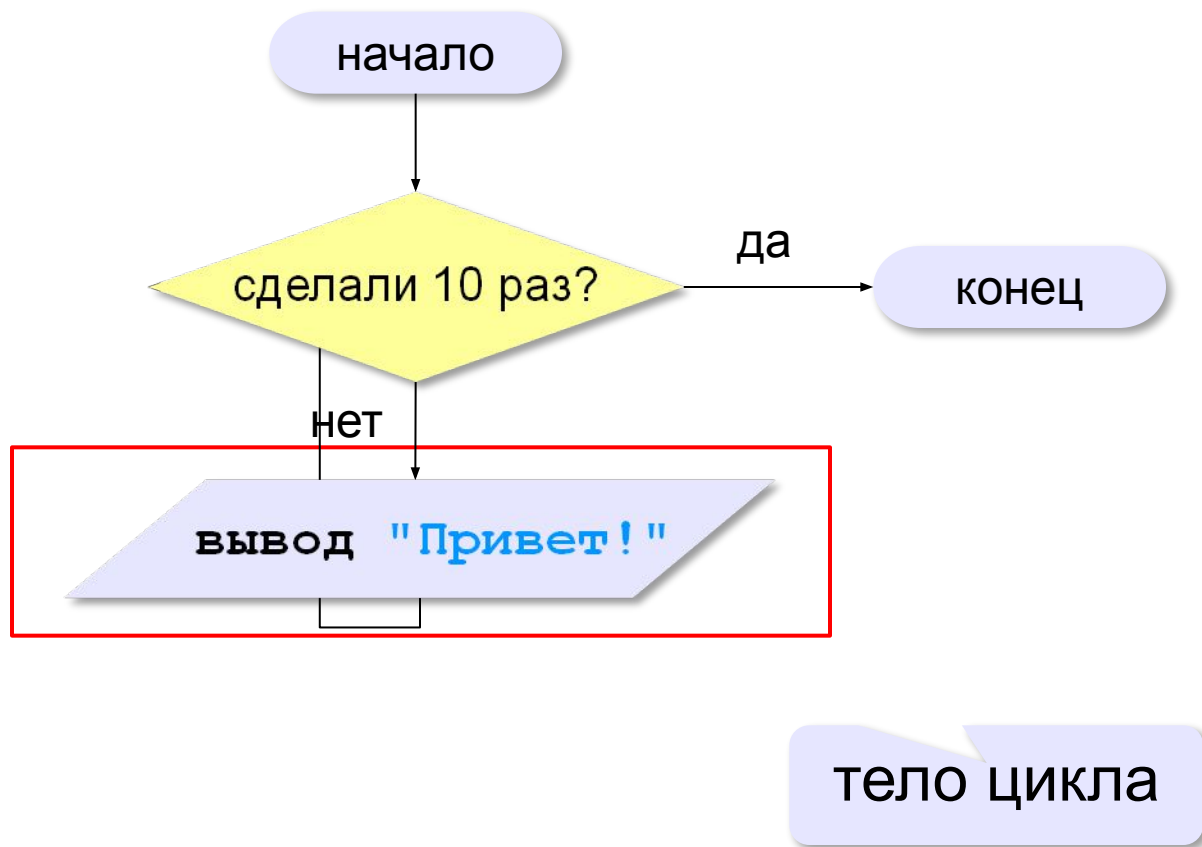
```
кц
```

```
кон
```



Как выглядит блок-схема?

# Блок-схема цикла



# Как организовать цикл?

```
счётчик := 0
пока счётчик < 10
  вывод 'привет', нс
  увеличить счётчик на 1
```

результат операции  
автоматически  
сравнивается с нулём!

```
счётчик := 10
пока счётчик > 0
  вывод 'привет', нс
  уменьшить счётчик на 1
```



Какой способ удобнее для процессора?

# Число шагов – переменная

---

Ввести количество повторений с клавиатуры:

```
цел N  
вывод 'Сколько раз? '  
ввод N  
нц N раз  
    вывод 'Привет! ', нс  
кц
```



Можно ли сделать без цикла?

## Цикл с условием

**Задача.** Определить **количество цифр** в десятичной записи целого положительного числа, записанного в переменную  $n$ .

```
счётчик := 0
пока n > 0
    отсечь последнюю цифру n
    увеличить счётчик на 1
```

$n$	счётчик
1234	0

**?** Как отсечь последнюю цифру?

```
n := div(n, 10)
```

**?** Как увеличить счётчик на 1?

```
счётчик := счётчик + 1
```



# Цикл с условием

начальное значение  
счётчика

условие  
продолжения

заголовок  
цикла

```
count := 0
нц пока n > 0
    n := div(n, 10)
    count := count + 1
кц
```

конец  
цикла

тело цикла



Цикл с предусловием – проверка на входе в цикл!

# Цикл с условием

---

При известном количестве шагов:

```
k := 0
нц пока k < 10
    вывод 'привет' , нс
    k := k + 1
кц
```

Зацикливание:

```
k := 0
нц пока k < 10
    вывод 'привет' , нс
кц
```

# Сколько раз выполняется цикл?

$a := 4; b := 6$

нц пока  $a < b$ ;  $a := a + 1$  кц

2 раза

$a = 6$

$a := 4; b := 6$

нц пока  $a < b$ ;  $a := a + b$  кц

1 раз

$a = 10$

$a := 4; b := 6$

нц пока  $a > b$ ;  $a := a + 1$  кц

0 раз

$a = 4$

$a := 4; b := 6$

нц пока  $a < b$ ;  $b := a - b$  кц

1 раз

$b = -2$

$a := 4; b := 6$

нц пока  $a < b$ ;  $a := a - 1$  кц

**защелкивание**

# Цикл с постусловием

заголовок  
цикла

нц

```
вывод 'Введите n > 0: '  
ввод n
```

кц при  $n > 0$  ;

тело цикла

условие  
окончания

- при входе в цикл условие **не проверяется**
- цикл всегда выполняется **хотя бы один раз**
- в последней строке указывают **условие окончания** цикла, а не условие его продолжения

# Задачи

---

**«А»:** Напишите программу, которая получает два целых числа  $A$  и  $B$  ( $0 < A < B$ ) и выводит квадраты всех натуральных чисел в интервале от  $A$  до  $B$ .

**Пример:**

**Введите два целых числа :**

**10 12**

$10 * 10 = 100$

$11 * 11 = 121$

$12 * 12 = 144$

**«В»:** Напишите программу, которая получает два целых числа и находит их произведение, не используя операцию умножения. Учтите, что числа могут быть отрицательными.

**Пример:**

**Введите два числа :**

**10 -15**

$10 * (-15) = -150$

# Задачи

---

**«С»:** Ввести натуральное число  $N$  и вычислить сумму всех чисел Фибоначчи, меньших  $N$ . Предусмотрите защиту от ввода отрицательного числа  $N$ .

**Пример:**

Введите число  $N$ :

**10000**

Сумма 17709

## Задачи-2

---

**«А»:** Ввести натуральное число и найти сумму его цифр.

**Пример:**

Введите натуральное число:

**12345**

Сумма цифр 15.

**«В»:** Ввести натуральное число и определить, верно ли, что в его записи есть две одинаковые цифры, стоящие рядом.

**Пример:**

Введите натуральное число:

**12342**

Нет.

**Пример:**

Введите натуральное число:

**12245**

Да.

## Задачи-2

---

**«С»:** Ввести натуральное число и определить, верно ли, что в его записи есть две одинаковые цифры (не обязательно стоящие рядом).

**Пример:**

**Введите натуральное число:**

**12342**

**Да .**

**Пример:**

**Введите натуральное число:**

**12345**

**Нет .**



# Цикл с переменной

Задача. Вывести все степени двойки от  $2^1$  до  $2^{10}$ .



Можно ли сделать с циклом «пока»?

```
k := 1
```

```
n := 2
```

```
нц пока k <= 10
```

```
    вывод n, нс
```

```
    n := n * 2
```

```
    k := k + 1
```

```
кц
```

```
n := 2
```

```
нц для k от 1 до 10
```

```
    вывод n, нс
```

```
    n := n * 2
```

```
кц
```



Переменная  $k$  – целая!

# Цикл с переменной: другой шаг

цел **k**

целое

целое

целое

```
нц для k от 10 до 1 шаг -1
  вывод k*k, нс
кц
```

100

81

64

49

36

25

16

9

4

1



Что получится?

## Сколько раз выполняется цикл?

```
а := 1
```

```
нц для і от 1 до 3; а:=а+1 кц
```

a = 4

```
а := 1
```

```
нц для і от 3 до 1; а:=а+1 кц
```

a = 1

```
а := 1
```

```
нц для і от 1 до 3 шаг -1; а:=а+1 кц
```

a = 1

```
а := 1
```

```
нц для і от 3 до 1 шаг -1; а:=а+1 кц
```

a = 4

# Задачи

---

«А»: Найдите все пятизначные числа, которые при делении на 133 дают в остатке 125, а при делении на 134 дают в остатке 111.

«В»: Натуральное число называется **числом Армстронга**, если сумма цифр числа, возведенных в N-ную степень (где N – количество цифр в числе) равна самому числу. Например,  $153 = 1^3 + 5^3 + 3^3$ .  
Найдите все трёхзначные Армстронга.

# Задачи

---

«С»: Натуральное число называется автоморфным, если оно равно последним цифрам своего квадрата.  
Например,  $25^2 = 625$ . Напишите программу, которая получает натуральное число N и выводит на экран все автоморфные числа, не превосходящие N.

**Пример:**

Введите N:

1000

$$1 * 1 = 1$$

$$5 * 5 = 25$$

$$6 * 6 = 36$$

$$25 * 25 = 625$$

$$76 * 76 = 5776$$

# Вложенные циклы

**Задача.** Вывести все простые числа в диапазоне от 2 до 1000.

```
нц для n от 2 до 1000
  если число n простое то
    вывод n, нс
все
кц
```

нет делителей [2.. n-1]:  
проверка в цикле!



Что значит «простое число»?

# Вложенные циклы

```
нц для n от 2 до 1000
```

```
  count := 0
```

```
    нц для k от 2 до n-1
```

```
      если  $\text{mod}(n, k) = 0$  то
```

```
        count := count + 1
```

```
      все
```

```
    кц
```

```
  если count = 0 то
```

```
    вывод n, нс
```

```
  все
```

```
кц
```

ВЛОЖЕННЫЙ ЦИКЛ

# Вложенные циклы

```

нц для i от 1 до 4
  нц для k от 1 до i
    вывод i, ' ', k, нс
  кц
кц

```

```

1 1
2 1
2 2
3 1
3 2
3 3
4 1
4 2
4 3
4 4

```

**?** Как меняются переменные?

**!** Переменная внутреннего цикла изменяется быстрее!



# Поиск простых чисел – как улучшить?

$$n = k \cdot m, \quad k \leq m \Rightarrow k^2 \leq n \Rightarrow k \leq \sqrt{n}$$

```
нц пока k <= sqrt(n)
```

```
...
```

```
кц
```



Что плохо?

```
count := 0
```

```
k := 2
```

```
нц пока k*k <= n
```

```
  если mod(n, k) = 0 то
```

```
    count := count + 1
```

```
  все
```

```
  k := k + 1
```

```
кц
```



Как ещё улучшить?

```
нц пока k*k <= n и (count = 0)
```

```
...
```

```
кц
```

# Задачи

---

**«А»:** Напишите программу, которая получает натуральные числа  $A$  и  $B$  ( $A < B$ ) и выводит все простые числа в интервале от  $A$  до  $B$ .

**Пример:**

**Введите границы диапазона:**

10 20

11 13 17 19

**«В»:** В магазине продается мастика в ящиках по 15 кг, 17 кг, 21 кг. Как купить ровно 185 кг мастики, не вскрывая ящики? Сколькими способами можно это сделать?

# Задачи

---

«С»: Ввести натуральное число  $N$  и вывести все натуральные числа, не превосходящие  $N$  и делящиеся на каждую из своих цифр.

**Пример:**

Введите  $N$ :

15

1 2 3 4 5 6 7 8 9 11 12 15

# Программирование на алгоритмическом языке

## § 59. Процедуры

# Зачем нужны процедуры?

вывод 'Ошибка программы'

много раз!

алг С процедурой

нач

цел n

ввод n

если  $n < 0$  то Error все

...

кон

ВЫЗОВ  
процедуры

алг Error

нач

вывод 'Ошибка программы'

кон

# Что такое процедура?

**Процедура** – вспомогательный алгоритм, который выполняет некоторые действия.

- текст (расшифровка) процедуры записывается **после основной программы**
- в программе может быть **много процедур**
- чтобы процедура заработала, нужно **вызвать** её по имени из основной программы или из другой процедуры

# Процедура с параметрами

Задача. Вывести на экран запись целого числа (0..255) в 8-битном двоичном коде.

много раз!

Алгоритм:

$$178 \Rightarrow 10110010_2$$

? Как вывести первую цифру?

$n := 1\ 0\ 1\ 1\ 0\ 0\ 1\ 0_2$  разряды

7 6 5 4 3 2 1 0

$\text{div}(n, 128)$

$\text{mod}(n, 128)$

? Как вывести вторую цифру?

$\text{div}(n1, 64)$

# Процедура с параметрами

**Задача.** Вывести на экран запись целого числа (0..255) в 8-битном двоичном коде.

**Алгоритм:**

```

k := 128
нц пока k > 0
    вывод div(n, k)
    n := mod(n, k)
    k := div(k, 2)
кц
  
```

178 ⇒ 10110010



Результат зависит от n!

n	k	ВЫВОД
178	128	1



# Процедура с параметрами

```
алг Двоичный код  
нач  
    printBin(99)  
кон
```

значение параметра  
(аргумент)

```
алг printBin(цел n0)  
нач  
    цел n, k  
    n := n0  
    k := 128  
    нц пока k > 0  
        вывод div(n, k)  
        n := mod(n, k)  
        k := div(k, 2)  
    кц  
кон
```

Параметры – данные,  
изменяющие работу процедуры.

локальные  
переменные

# Несколько параметров

---

```
алг printSred(цел a, цел b)
```

```
нач
```

```
    вывод (a+b) / 2
```

```
кон
```

```
алг printSred(цел a, b)
```

```
нач
```

```
    вывод (a+b) / 2
```

```
кон
```

# Задачи

---

«А»: Напишите процедуру, которая принимает параметр – натуральное число  $N$  – и выводит на экран линию из  $N$  символов '–'.

**Пример:**

Введите  $N$ :

10

-----

«В»: Напишите процедуру, которая выводит на экран в столбик все цифры переданного ей числа, начиная с первой.

**Пример:**

Введите натуральное число:

1234

1

2

3

4

# Задачи

---

**«С»:** Напишите процедуру, которая выводит на экран запись переданного ей числа в римской системе счисления.

**Пример:**

**Введите натуральное число:**

**2013**

**ММХІІІ**

# Изменяемые параметры

**Задача.** Написать процедуру, которая меняет местами значения двух переменных.


```
алг Тест  
нач  
  цел  $x = 2, y = 3$   
  Обмен ( $x, y$ )  
  Вывод  $x, ' ', y$   
кон
```

 Почему не работает?

2 3

```
алг Обмен (цел  $a, b$ )  
нач  
  цел  $c$   
   $c := a; a := b; b := c$   
кон
```

передача по  
значению

 Процедура работает с копиями переданных значений параметров!

# Изменяемые параметры

переменные могут изменяться  
(**аргумент** и **результат**)

```
алг Обмен ( аргрез цел a, b)
```

```
нач
```

```
  цел c
```

```
  c := a; a := b; b := c
```

```
кон
```

передача по  
ССЫЛКЕ

## Вызов:

```
цел a, b
```

```
Обмен (a, b) | правильно
```

```
Обмен (2, 3) | неправильно
```

```
Обмен (a, b+3) | неправильно
```

# Задачи

---

**«А»:** Напишите процедуру, которая переставляет три переданные ей числа в порядке возрастания.

**Пример:**

**Введите три натуральных числа:**

**10 15 5**

**5 10 15**

**«В»:** Напишите процедуру, которая сокращает дробь вида  $M/N$ . Числитель и знаменатель дроби передаются как изменяемые параметры.

**Пример:**

**Введите числитель и знаменатель дроби:**

**25 15**

**После сокращения: 5/3**

# Задачи

---

**«С»:** Напишите процедуру, которая вычисляет наибольший общий делитель и наименьшее общее кратное двух натуральных чисел и возвращает их через изменяемые параметры.

**Пример:**

Введите два натуральных числа :

**10 15**

**НОД (10 , 15) =5**

**НОК (10 , 15) =30**



# Программирование на алгоритмическом языке

## § 60. Функции

# Что такое функция?

**Функция** – это вспомогательный алгоритм, который возвращает *значение-результат* (число, символ или объект другого типа).

**Задача.** Написать функцию, которая вычисляет сумму цифр числа.

**Алгоритм:**

```
сумма := 0
нц пока n <> 0
    сумма := сумма + mod (n, 10)
    n := div (n, 10)
кц
```

# Сумма цифр числа

```
алг Сумма цифр
нач
    ВЫВОД sumDigits (12345)
кон
```

```
алг цел sumDigits (цел n0)
нач
    тип результата
    цел sum = 0, n
    n := n0
    нц пока n <> 0
        sum := sum + mod (n, 10)
        n := div (n, 10)
    кц
    знач := sum
    передача результата
кон
```

# Использование функций

```
x := 2 * sumDigits (n+5)
```

```
z := sumDigits (k) + sumDigits (m)
```

```
если mod (sumDigits (n) , 2) = 0
```

```
    вывод 'Сумма цифр чётная' , нс
```

```
    вывод 'Она равна ' , sumDigits (n)
```

```
кц
```



Функция, возвращающая целое число, может использоваться везде, где и целая величина!

# Задачи

---

**«А»:** Напишите функцию, которая находит наибольший общий делитель двух натуральных чисел.

**Пример:**

Введите два натуральных числа:

**7006652 112307574**

$\text{НОД}(7006652, 112307574) = 1234.$

**«В»:** Напишите функцию, которая определяет сумму цифр переданного ей числа.

**Пример:**

Введите натуральное число:

**123**

Сумма цифр числа 123 равна 6.

# Задачи

---

**«С»:** Напишите функцию, которая «переворачивает» число, то есть возвращает число, в котором цифры стоят в обратном порядке.

**Пример:**

**Введите натуральное число:**

**1234**

**После переворота: 4321.**

# Логические функции

Задача. Найти все простые числа в диапазоне от 2 до 100.

```
алг Простые числа
нач
  цел i
  нц для i от 2 до 100
    если isPrime(i) то
      вывод i, нс
    все
  кц
кон
```

функция,  
возвращающая  
логическое  
значение (да/нет)

# Функция: простое число или нет?

**?** Какой алгоритм?

```
алг лог isPrime (цел n)
нач
  цел count = 0, k
  k := 2
  нц пока k*k <= n и count = 0
    если mod(n, k) = 0 то
      count := count + 1
    все
    k := k + 1
  кц
  знач := (count = 0)
кон
```

если count = 0 то  
 знач := да  
иначе знач := нет



# Логические функции: использование

---



Функция, возвращающая логическое значение, может использоваться везде, где и логическая величина!

```
ввод n
нц пока isPrime (n)
    вывод 'простое число' , нс
ввод n
кц
```

# Задачи

---

**«А»:** Напишите логическую функцию, которая определяет, является ли переданное ей число совершенным, то есть, равно ли оно сумме своих делителей, меньших его самого.

**Пример:**

Введите натуральное число:

**28**

Число 28 совершенное.

**Пример:**

Введите натуральное число:

**29**

Число 29 не совершенное.

# Задачи

---

**«В»:** Напишите логическую функцию, которая определяет, являются ли два переданные ей числа взаимно простыми, то есть, не имеющими общих делителей, кроме 1.

**Пример:**

Введите два натуральных числа:

**28 15**

Числа 28 и 15 взаимно простые.

**Пример:**

Введите два натуральных числа:

**28 16**

Числа 28 и 16 не взаимно простые.

# Задачи

---

**«С»:** Простое число называется гиперпростым, если любое число, получающееся из него откидыванием нескольких цифр, тоже является простым. Например, число 733 – гиперпростое, так как и оно само, и числа 73 и 7 – простые. Напишите логическую функцию, которая определяет, верно ли, что переданное ей число – гиперпростое. Используйте уже готовую функцию `isPrime`, которая приведена в учебнике.

## Пример:

Введите натуральное число:

**733**

Число 733 гиперпростое.

## Пример:

Введите натуральное число:

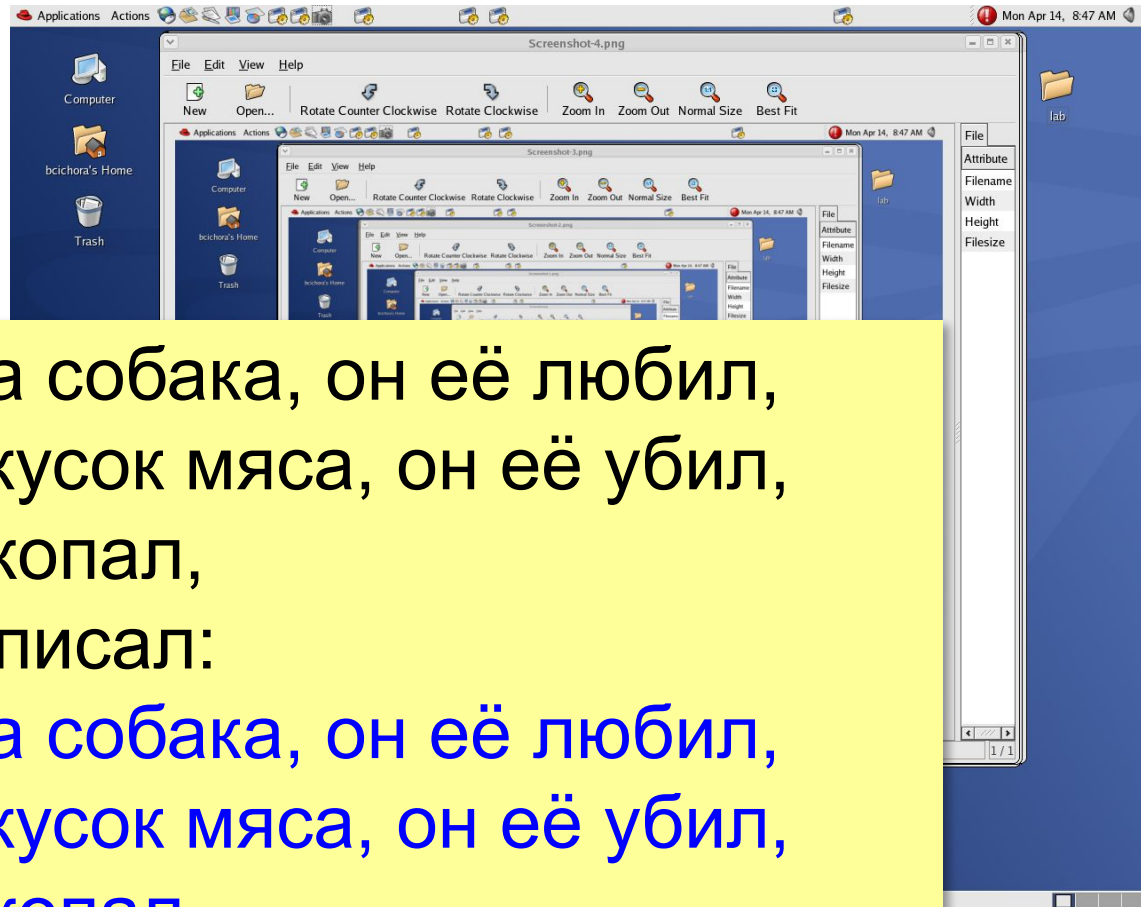
**19**

Число 19 не гиперпростое.

# Программирование на алгоритмическом языке

## § 61. Рекурсия

# Что такое рекурсия?



У попа была собака, он её любил,  
Она съела кусок мяса, он её убил,  
В землю закопал,  
Надпись написал:  
У попа была собака, он её любил,  
Она съела кусок мяса, он её убил,  
В землю закопал,  
Надпись написал:

...

# Что такое рекурсия?

## Натуральные числа:

- 1 – натуральное число
- если  $n$  – натуральное число, то  $n + 1$  – натуральное число

индуктивное  
определение

**Рекурсия** — это способ определения множества объектов через само это множество на основе заданных простых базовых случаев.

## Числа Фибоначчи:

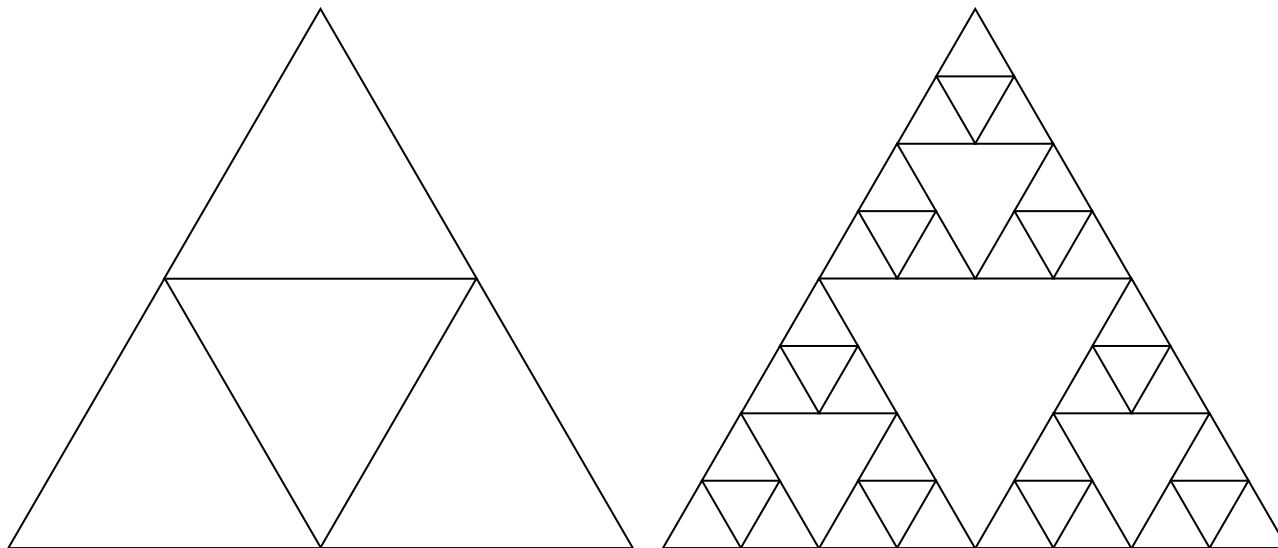
- $F_1 = F_2 = 1$
- $F_n = F_{n-1} + F_{n-2}$  при  $n > 2$

**1, 1, 2, 3, 5, 8, 13, 21, 34, ...**

# Фракталы

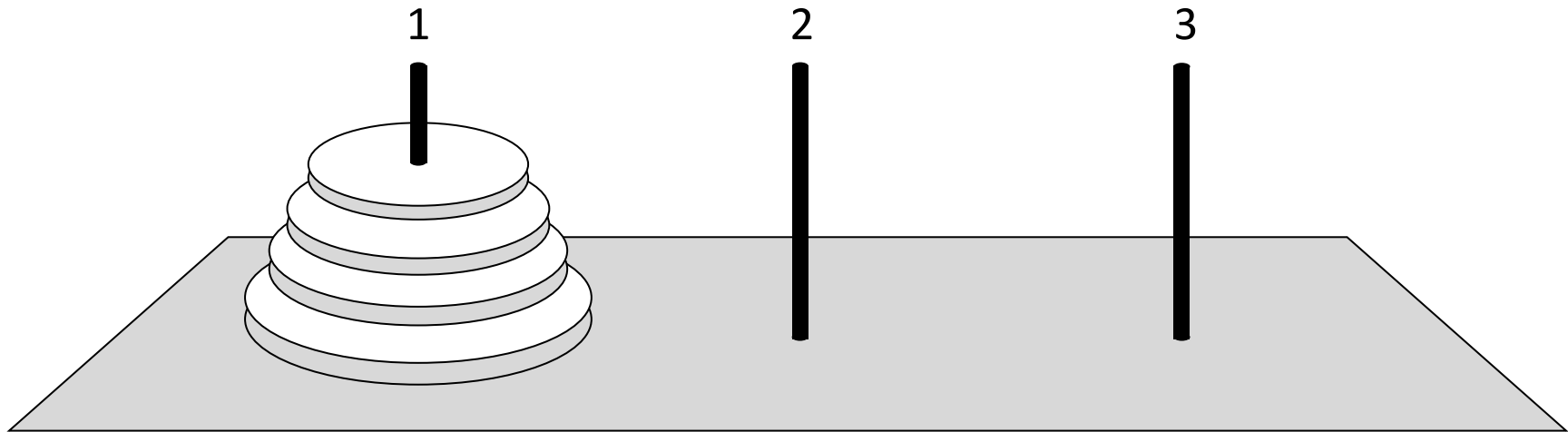
**Фракталы** – геометрические фигуры, обладающие самоподобием.

**Треугольник Серпинского:**





# Ханойские башни



- за один раз переносится один диск
- класть только меньший диск на больший
- третий стержень вспомогательный

перенести ( $n$ , 1, 3)

перенести ( $n-1$ , 1, 2)

1  $\rightarrow$  3

перенести ( $n-1$ , 2, 3)

# Ханойские башни – процедура

СКОЛЬКО

откуда

куда

```
алг Hanoi (цел n, k, m)
```

```
нач
```

```
  цел p
```

рекурсия

```
  p := 6 - k - m
```

```
  Hanoi (n-1, k, p)
```

рекурсия

```
  вывод k, ' -> ', m, нс
```

```
  Hanoi (n-1, p, m)
```

```
кон
```

номер вспомогательного  
стержня (1+2+3=6!)



Что плохо?



**Рекурсия никогда не остановится!**

# Ханойские башни – процедура

Рекурсивная процедура (функция) — это процедура (функция), которая вызывает сама себя напрямую или через другие процедуры и функции.

```
алг Hanoi (цел n, k, m)
```

```
нач
```

```
если n = 0 то выход все
```

```
цел p
```

```
p := 6 - k - m
```

```
Hanoi (n-1, k, p)
```

```
вывод k, ' -> ', m, нс
```

```
Hanoi (n-1, p, m)
```

```
кон
```

условие выхода из  
рекурсии

```
алг Ханойские башни
```

```
нач
```

```
Hanoi (4, 1, 3)
```

```
кон
```

# Вывод двоичного кода числа

```
алг printBin (цел n)
нач
  если n = 0 то выход все
  printBin ( div (n, 2) )
  вывод mod ( n, 2 )
кон
```

условие выхода из  
рекурсии

напечатать все  
цифры, кроме  
последней

ВЫВЕСТИ  
последнюю цифру

```
printBin ( 0 )
```



**?** Как без рекурсии?

# Вычисление суммы цифр числа

```
алг цел sumDig(цел n)
```

```
нач
```

```
  знач := mod(n, 10)
```

```
  если n >= 10 то
```

```
    знач := знач + sumDig( div(n, 10) )
```

```
  все
```

```
кон
```

последняя цифра

рекурсивный вызов

?

Где условие окончания рекурсии?

```
sumDig( 1234 )
```

```
4 + sumDig( 123 )
```

```
4 + 3 + sumDig( 12 )
```

```
4 + 3 + 2 + sumDig( 1 )
```

```
4 + 3 + 2 + 1
```

# Алгоритм Евклида

**Алгоритм Евклида.** Чтобы найти НОД двух натуральных чисел, нужно вычитать из большего числа меньшее до тех пор, пока меньшее не станет равно нулю. Тогда второе число и есть НОД исходных чисел.

```
алг цел NOD (цел a, b)
нач
  если a = 0 или b = 0 то
    знач := a + b
    выход
  все
  если a > b то
    знач := NOD (a - b, b)
  иначе знач := NOD (a, b - a)
  все
кон
```

условие окончания  
рекурсии

рекурсивные вызовы

# Задачи

---

**«А»:** Напишите рекурсивную функцию, которая вычисляет НОД двух натуральных чисел, используя модифицированный алгоритм Евклида.

**Пример:**

Введите два натуральных числа :

**7006652 112307574**

**НОД (7006652 , 112307574) = 1234 .**

**«В»:** Напишите рекурсивную функцию, которая раскладывает число на простые сомножители.

**Пример:**

Введите натуральное число :

**378**

**378 = 2\*3\*3\*3\*7**

# Задачи

---

**«С»:** Дано натуральное число  $N$ . Требуется получить и вывести на экран количество всех возможных *различных* способов представления этого числа в виде суммы натуральных чисел (то есть,  $1 + 2$  и  $2 + 1$  – это один и тот же способ разложения числа 3). Решите задачу с помощью рекурсивной процедуры.

**Пример:**

**Введите натуральное число:**

**4**

**Количество разложений: 4.**



# Как работает рекурсия?

**Факториал:**

$$N! = \begin{cases} 1, & N = 1 \\ N \cdot (N-1)!, & N > 1 \end{cases}$$

```

алг цел Fact (цел N)
нач
  вывод '-> N = ', N, нс
  если N <= 1 то
    знач := 1
  иначе знач := N * Fact (N-1)
  все
  вывод '<- N = ', N, нс
кон
  
```

```

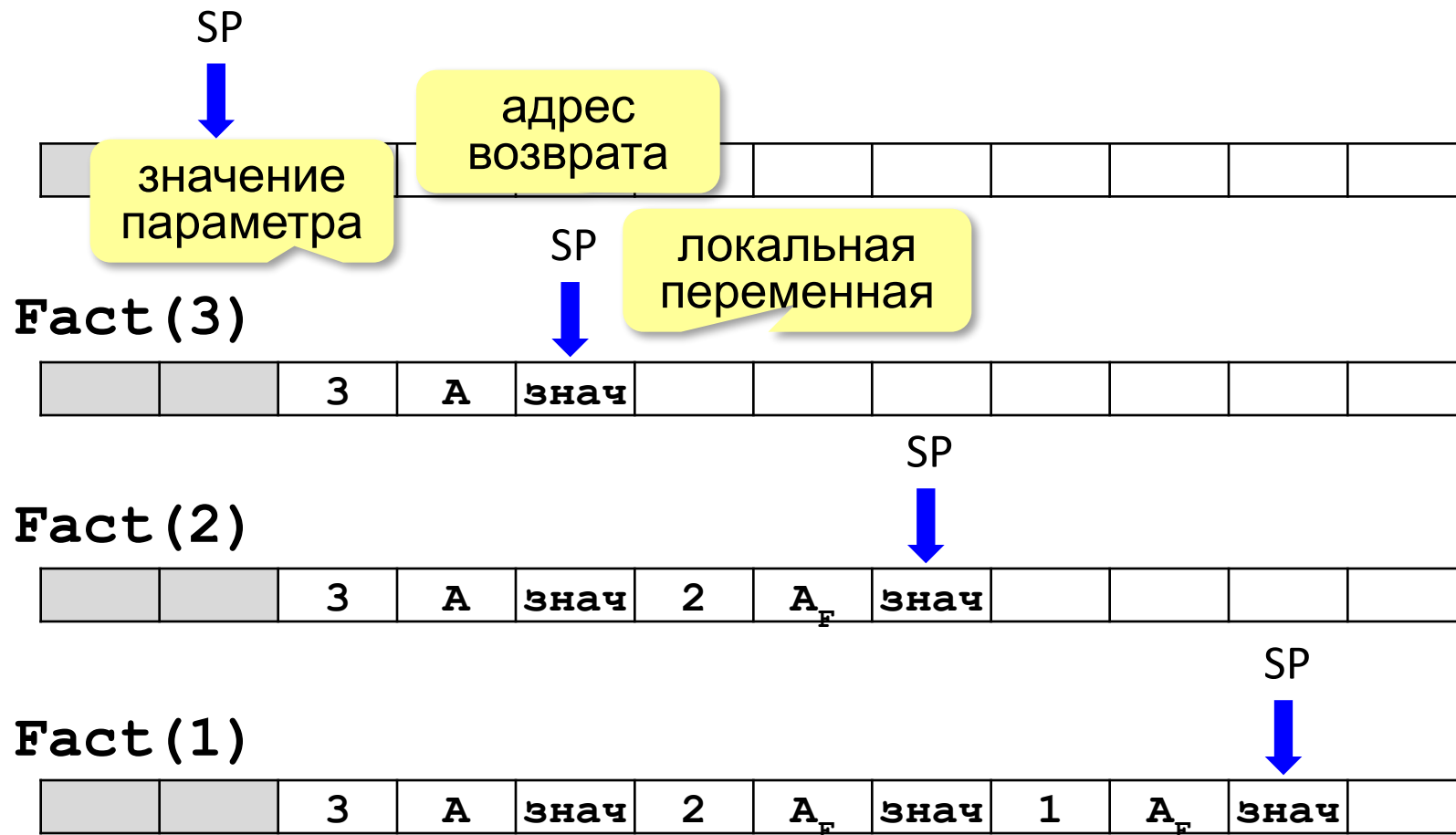
-> N = 3
  -> N = 2
    -> N = 1
      <- N = 1
        <- N = 2
          <- N = 3
  
```



Как сохранить состояние функции перед рекурсивным вызовом?

# Стек

**Стек** – область памяти, в которой хранятся локальные переменные и адреса возврата.



# Рекурсия – «за» и «против»

- с каждым новым вызовом расходуется память в стеке (возможно переполнение стека)
- затраты на выполнение служебных операций при рекурсивном вызове



▪ программа становится более короткой и понятной



- возможно переполнение стека
- замедление работы



Любой рекурсивный алгоритм можно заменить нерекурсивным!

итерационный  
алгоритм

```
алг цел Fact (цел N)
нач
  цел i
  знач := 1
  нц для i от 1 до N
    знач := знач * i
  кц
кон
```

# Конец фильма

---

**ПОЛЯКОВ Константин Юрьевич**

д.т.н., учитель информатики

ГБОУ СОШ № 163, г. Санкт-Петербург

[kpolyakov@mail.ru](mailto:kpolyakov@mail.ru)

**ЕРЕМИН Евгений Александрович**

к.ф.-м.н., доцент кафедры мультимедийной

дидактики и ИТО ПГГПУ, г. Пермь

[eremin@pspu.ac.ru](mailto:eremin@pspu.ac.ru)

# Источники иллюстраций

---

1. [old-moneta.ru](http://old-moneta.ru)
2. [www.random.org](http://www.random.org)
3. [www.allruletka.ru](http://www.allruletka.ru)
4. [www.lotterypros.com](http://www.lotterypros.com)
5. [logos.cs.uic.edu](http://logos.cs.uic.edu)
6. [ru.wikipedia.org](http://ru.wikipedia.org)
7. иллюстрации художников издательства «Бином»
8. авторские материалы