

# Программирование на языке Python

Алгоритм и его свойства

Простейшие программы

Вычисления

Ветвления

Символьные строки

Циклические алгоритмы

Массивы (списки)

Поиск в массиве

# Программирование на языке Python

## 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 одушёвленное 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** (определенность) — 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.

**Массовость** — алгоритм можно использовать для разных исходных данных.

# Программирование на языке Python

## Простейшие программы

# Простейшая программа

```
# Это пустая программа
```



Что делает эта программа?

комментарии после #  
не обрабатываются

кодировка utf-8  
по умолчанию)

```
# coding: utf-8
```

```
# Это пустая программа
```

"""

Это тоже комментарий

"""

# Вывод на экран

- ▶ `print ( "2+2=?")`
- ▶ `print ( "Ответ: 4" )`

автоматический  
переход на новую  
строку

## Протокол:

2+2=?

Ответ: 4

```
print ( '2+2=?' )
print ( 'Ответ: 4' )
```

## Задания

---

«4»: Вывести на экран текст «лесенкой»

Вася

пошел

гулять

«5»: 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

25  
30

$25+30=55$

компьютер

пользователь

компьютер считает сам!



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

# Сумма: псевдокод

**ввести два числа  
вычислить их сумму  
вывести сумму на экран**

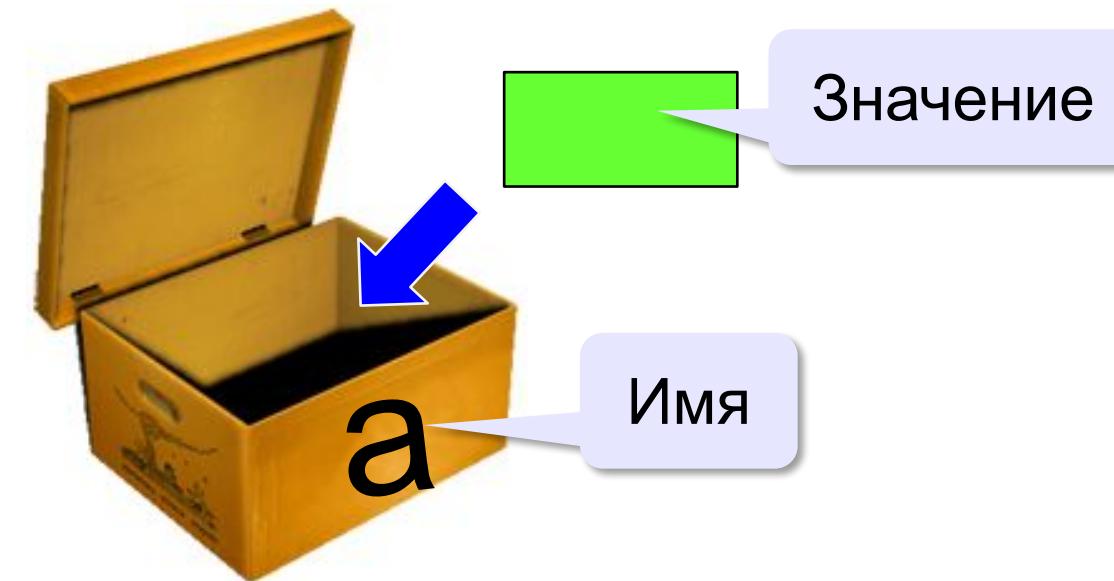
Псевдокод – алгоритм на русском языке с элементами языка программирования.



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, a-z)

заглавные и строчные буквы **различаются**

- русские буквы (**не рекомендуется!**)
- цифры

имя не может начинаться с цифры

- знак подчеркивания \_

**НЕЛЬЗЯ** использовать

- скобки
- знаки +, =, !, ? и др.

Какие имена правильные?

**AХby R&B 4Wheel Вася “PesBarbos”**

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

# Типы переменных

```
a = 4
```

```
print( type(a) )
```

```
<class 'int'>
```

целое число (*integer*)

```
a = 4.5
```

```
print( type(a) )
```

```
<class 'float'>
```

вещественное число

```
a = "Вася"
```

```
print( type(a) )
```

```
<class 'str'>
```

символьная строка

```
a = True
```

```
print( type(a) )
```

```
<class 'bool'>
```

логическая

# Зачем нужен тип переменной?

---

Тип определяет:

- область допустимых значений
- допустимые операции
- объём памяти
- формат хранения данных

# Как записать значение в переменную?

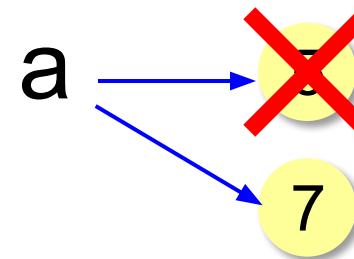
оператор  
присваивания

a = 5

a = 7



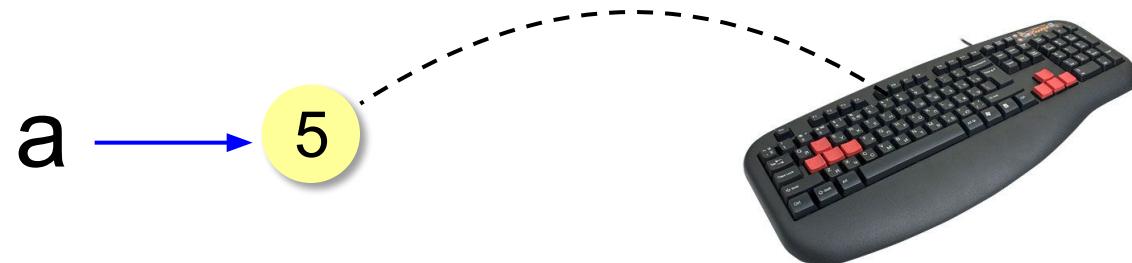
При записи нового значения  
старое удаляется из памяти!



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

**Оператор присваивания** – это команда для  
присваивания нового значения переменной.

# Ввод значения с клавиатуры



1. Программа ждет, пока пользователь введет значение и нажмет *Enter*.
2. Введенное значение записывается в переменную **a** (связывается с именем **a**)

# Ввод значения с клавиатуры

```
a = input()
```

ввести строку с клавиатуры  
и связать с переменной a

```
b = input()
```

```
c = a + b
```

```
print( c )
```

Протокол:

21

33

2133



? Почему?



Результат функции `input` – строка символов!

преобразовать в  
целое число

```
a = int( input() )
```

```
b = int( input() )
```

# Ввод с подсказкой

```
a = input ( "Введите число: " )
```

Введите число: 26

подсказка

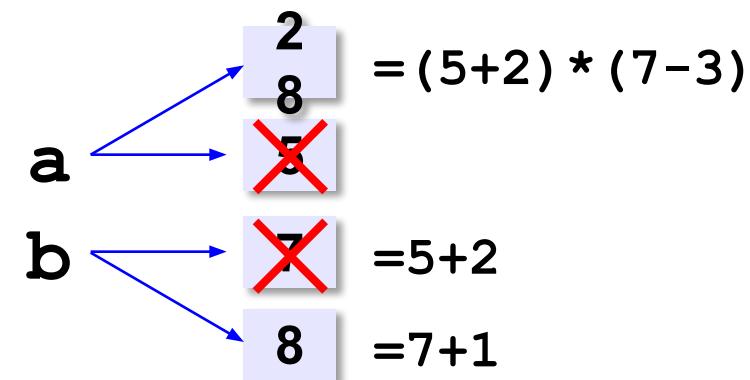


Что не так?

```
a = int( input("Введите число: " ) )
```

# Изменение значений переменной

```
a = 5  
b = a + 2  
a = (a + 2) * (b - 3)  
b = b + 1
```



# Вывод данных

```
print ( a )
```

значение  
переменной

```
print ( "Ответ: ", a )
```

значение и  
текст

перечисление через запятую

```
print ( "Ответ: ", a+b )
```

вычисление  
выражения

```
print ( a, "+", b, "=", c )
```

2 + 3 = 5

через пробелы

```
print ( a, "+", b, "=", c, sep = "" )
```

2+3=5

убрать разделители

## Вывод данных через `format`

```
print ( a, "+", b, "=", c, sep = "" )
```

2+3=5

```
print ( "{}+{}={}".format(a, b, c) )
```

# Сложение чисел: простое решение

```
a = int( input() )  
b = int( input() )  
c = a + b  
print( c )
```



Что плохо?

# Сложение чисел: полное решение

```
print ( "Введите два числа: " )  
a = int ( input () )  
b = int ( input () )  
c = a + b  
print ( a, "+", b, "=", c, sep="" )
```

подсказка

Протокол:

компьютер

Введите два целых числа

25

пользователь

30

25+30=55

# Задания

---

**«3»: Ввести три числа, найти их сумму.**

*Пример:*

Введите три числа:

4

5

7

$$4+5+7=16$$

**«4»: Ввести три числа, найти их сумму и произведение.**

*Пример:*

Введите три числа:

4

5

7

$$4+5+7=16$$

$$4*5*7=140$$

# Задания

---

**«5»:** Ввести три числа, найти их сумму, произведение и среднее арифметическое.

**Пример:**

Введите три числа:

4

5

7

$4+5+7=16$

$4*5*7=140$

$(4+5+7) / 3=5.333333$

# Программирование на языке Python

## Вычисления

# Арифметическое выражения

3    1    2    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

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

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

перенос на  
следующую строку

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

перенос внутри  
скобок разрешён

# Деление

Классическое деление:

```
a = 9; b = 6
x = 3 / 4      # = 0.75
x = a / b      # = 1.5
x = -3 / 4     # = -0.75
x = -a / b     # = -1.5
```

Целочисленное деление (округление «вниз»!):

```
a = 9; b = 6
x = 3 // 4      # = 0
x = a // b      # = 1
x = -3 // 4     # = -1
x = -a // b     # = -2
```

# Остаток от деления

$\%$  – остаток от деления

```
d = 85  
b = d // 10  
a = d % 10  
d = a % b  
d = b % a
```

```
a = 15  
b = 19  
d = a // b  
a = a % b
```

## Операторы // и %

```
a = 1234
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10 :
```

4

3

2

1

# Сокращенная запись операций

```
a += b # a = a + b  
a -= b # a = a - b  
a *= b # a = a * b  
a /= b # a = a / b  
a // = b # a = a // b  
a %= b # a = a % b
```

a += 1

увеличение на 1

# Ввод двух значений в одной строке

```
a, b = map ( int, input().split() )
```

21 33

input()

ввести строку с клавиатуры

21 33

input().split()

целые

применить

разделить строку на  
части по пробелам

21 33

map ( int, input().split() )

эту  
операцию

к каждой части

```
a, b = map ( int, input().split() )
```

# Задания

---

**«3»:** Ввести три числа: цену пирожка (два числа: рубли, потом – копейки) и количество пирожков. Найти сумму, которую нужно заплатить (рубли и копейки)

*Пример:*

Стоимость пирожка:

12 50

Сколько пирожков:

5

К оплате: 62 руб. 50 коп.

**«4»:** Ввести число, обозначающее количество секунд. Вывести то же самое время в часах, минутах и секундах.

*Пример:*

Число секунд:

8325

2 ч. 18 мин. 45 с

## Задания

---

**«5»:** Занятия в школе начинаются в 8-30. Урок длится 45 минут, перерывы между уроками – 10 минут. Ввести номер урока и вывести время его окончания.

*Пример:*

Введите номер урока:

6

13-50

# Случайные числа

## Случайно...

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

## Случайный выбор:

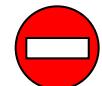
- жеребьевка на соревнованиях
- выигравшие номера в лотерее

## 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

# Линейный конгруэнтный генератор

$x = (a \cdot x + b) \% c$  | интервал от 0 до  $c-1$

$x = (x+3) \% 10$  | интервал от 0 до 9

$x = 0 \rightarrow 3 \rightarrow 6 \rightarrow 9 \rightarrow 2 \rightarrow 5 \rightarrow 8$

зерно

8 → 1 → 4 → 7 → 0

зацикливание



Важен правильный выбор параметров  
 $a$ ,  $b$  и  $c$ !

Компилятор GCC:

$a = 1103515245$

$b = 12345$

$c = 2^{31}$

# Генератор случайных чисел

```
import random
```

англ. *random* – случайный

Целые числа на отрезке [a,b]:

```
X = random.randint(1, 6) # псевдосл. число  
Y = random.randint(1, 6) # уже другое число!
```

Генератор на [0,1):

```
X = random.random() # псевдосл. число  
Y = random.random() # уже другое число!
```

Генератор на [a, b] (вещественные числа):

```
X = random.uniform(1.2, 3.5)  
Y = random.uniform(1.2, 3.5)
```

# Генератор случайных чисел

```
from random import *
```

подключить все!

англ. *random* – случайный

Целые числа на отрезке [a,b]:

```
X = randint(10, 60) # псевдослучайное число  
Y = randint(10, 60) # это уже другое число!
```

Генератор на [0,1):

```
X = random() # псевдослучайное число  
Y = random() # это уже другое число!
```

## Задачи

---

**«3»:** Игральный кубик бросается три раза (выпадает три случайных значения). Сколько очков в среднем выпало?

**Пример:**

Выпало очков:

5 3 1

$$(5+3+1) / 3 = 3$$

**«4»:** Игральный кубик бросается три раза (выпадает три случайных значения). Из этих чисел составляется целое число, программа должна найти его квадрат.

**Пример:**

Выпало очков:

1 2 3

Число 123

Его квадрат 15129

# Задачи

---

**«5»:** Получить случайное трёхзначное число и вывести через запятую его отдельные цифры.

**Пример:**

Получено число 123

сотни: 1

десятки: 2

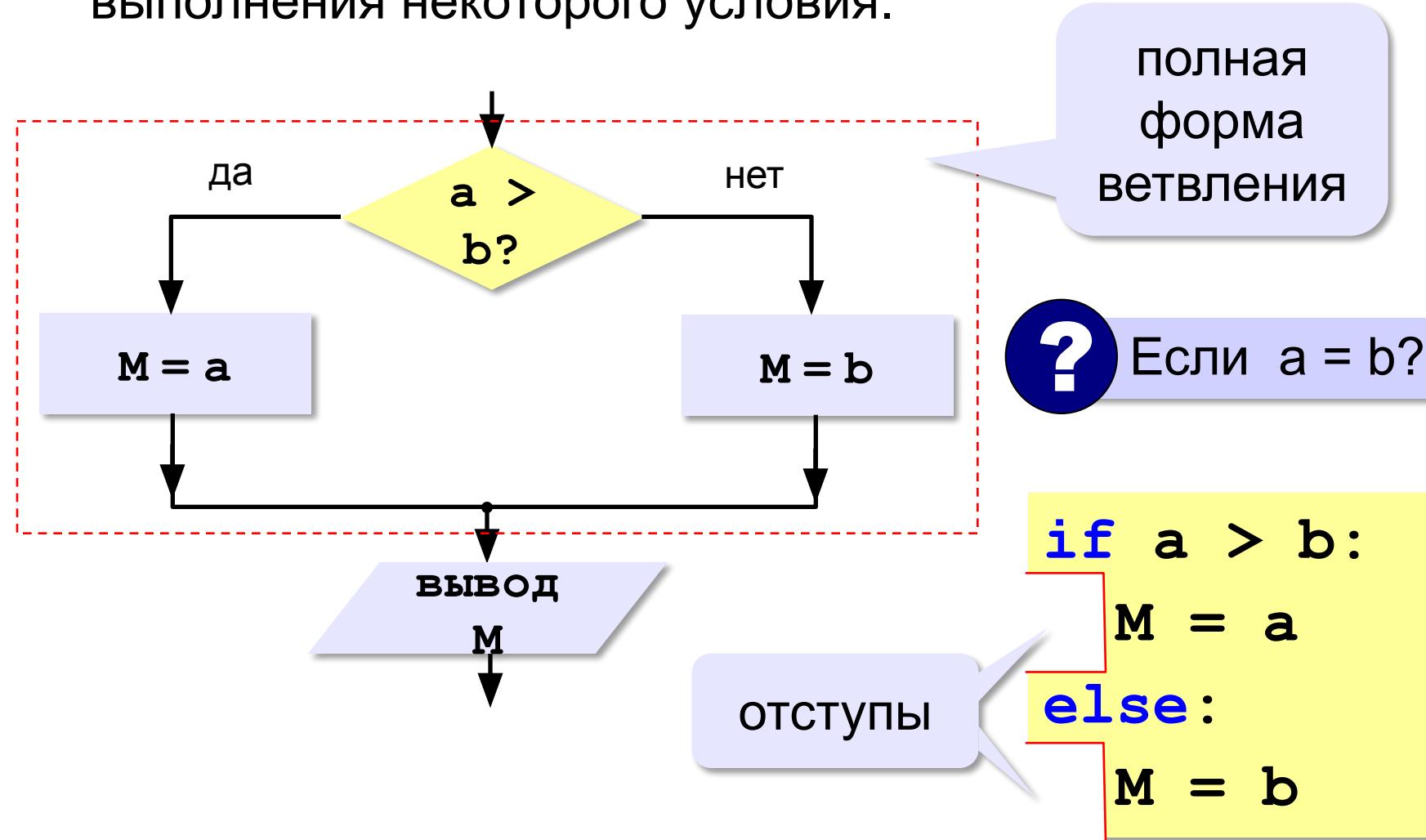
единицы: 3

# Программирование на языке Python

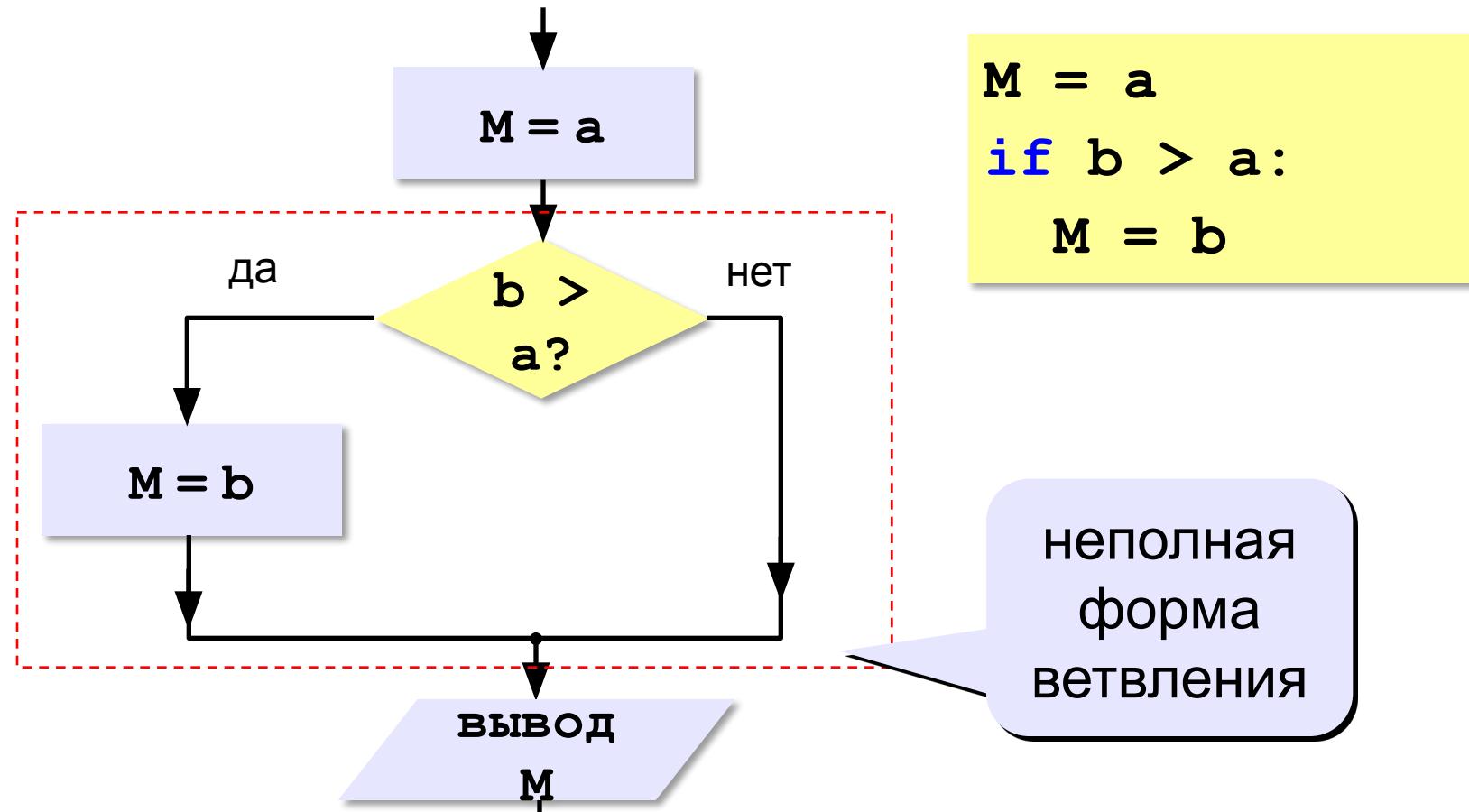
## Ветвления

# Условный оператор

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



# Условный оператор: неполная форма



```
M = a  
if b > a:  
    M = b
```

неполная  
форма  
ветвления

Решение в стиле Python:

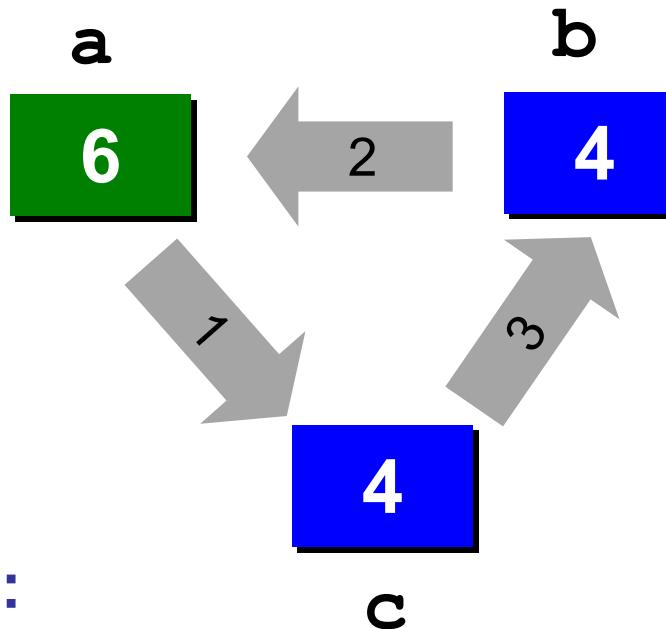
**M = max (a , b)**

**M=a if a>b else b**

# Условный оператор

```
if a < b:  
    c = a  
    a = b  
    b = c
```

Что делает?



Можно ли обойтись  
без переменной *c*?

Решение в стиле Python:

```
a, b = b, a
```

# 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?

```
if a > b:  
    print("Андрей старше")  
else:  
    if a == b:  
        print("Одного возраста")  
    else:  
        print("Борис старше")
```



Зачем нужен?

вложенный  
условный оператор

# Каскадное ветвление

```
if a > b:  
    print("Андрей старше")  
elif a == b:  
    print("Одного возраста")  
else:  
    print("Борис старше")
```



**! elif = else if**

# Каскадное ветвление

```
cost = 1500
if cost<1000:
    print ( "Скидок нет." )
elif cost<2000:
    print ( "Скидка 2%." )
elif cost<5000:
    print ( "Скидка 5%." )
else:
    print ( "Скидка 10%." )
```

первое сработавшее  
условие



Что выведет?

Скидка 2%.

## Задачи (без функций **min** и **max**!)

---

**«3»:** Ввести два целых числа, найти наибольшее и наименьшее из них.

**Пример:**

Ведите два целых числа:

1 5

Наибольшее число 5

Наименьшее число 1

**«4»:** Ввести четыре целых числа, найти наибольшее из них.

**Пример:**

Ведите четыре целых числа:

1 5 4 3

Наибольшее число 5

# Задачи

---

**'5': Ввести пять чисел и найти наибольшее из них.**

*Пример:*

Введите пять чисел:

4      15      9      56      4

Наибольшее число 56

## Задачи

---

**«6»:** 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

```
if v >= 25 and v <= 40 :  
    print("подходит")  
else:  
    print("не подходит")
```

**and** «И»: одновременное выполнение  
всех условий!

# Сложные условия

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

сложное условие

```
if v < 25 or v > 40:  
    print("не подходит")  
else:  
    print("подходит")
```

or

«ИЛИ»: выполнение **хотя бы одного**  
из двух условий!

# Сложные условия

```
if not(a < b):  
    print("Старт!")
```



Как без «НЕ»?

**not** «НЕ»: если выполняется обратное условие

```
if a >= b:  
    print("Старт!")
```

**Приоритет :**

- 1) отношения (<, >, <=, >=, ==, !=)
- 2) **not** («НЕ»)
- 3) **and** («И»)
- 4) **or** («ИЛИ»)

## Задачи

---

**«3»:** Напишите программу, которая получает три числа - рост трёх спортсменов, и выводит сообщение «По росту.», если они стоят по возрастанию роста, или сообщение «Не по росту!», если они стоят не по росту.

**Пример:**

Ведите рост трёх спортсменов:

165 170 172

По росту.

**Пример:**

Ведите рост трёх спортсменов:

175 170 172

Не по росту!

## Задачи

---

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

**Пример:**

Ведите номер месяца:

5

Весна .

**Пример:**

Ведите номер месяца:

15

Неверный номер месяца .

## Задачи

---

**«5»:** 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:**

Ведите возраст: 18

Вам 18 лет.

**Пример:**

Ведите возраст: 21

Вам 21 год.

**Пример:**

Ведите возраст: 22

Вам 22 года.

# Программирование на языке Python

## Символьные строки

# Символьные строки

Начальное значение:

```
s = "Привет!"
```



Строка – это  
последовательность  
символов!

Вывод на экран:

```
print( s )
```

Сложение:

```
s1 = "Привет"
```

"Привет, Вася!"

```
s2 = "Вася"
```

```
s = s1 + ", " + s2 + "!"
```

Умножение:

```
s = "Ay"
```

```
s5 = s*5
```

s5 = s + s + s + s + s

ΔυΔυΔυΔυ



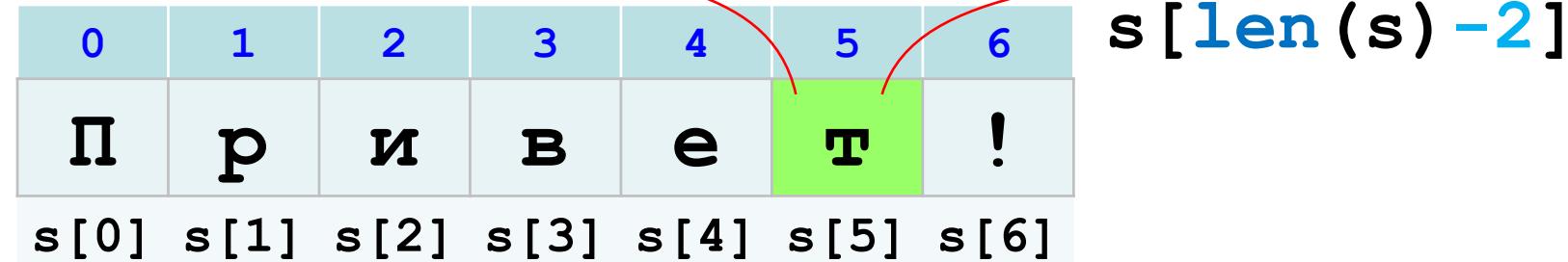
Что получим?

# Символьные строки

Вывод символа на экран:

```
print ( s[5] )
```

```
print ( s[-2] )
```



Длина строки:

```
n = len ( s )
```

# Символьные строки

Ввод с клавиатуры:

```
s = input( "Введите имя: " )
```

Изменение строки запрещено!

```
s[4] = "a"
```



Строка – это неизменяемый объект!

... но можно составить новую строку:

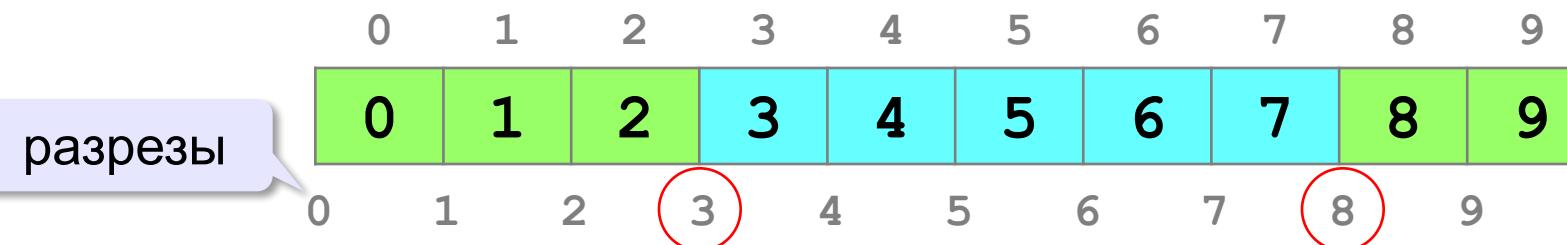
```
s1 = s + "а"
```

составить «КОТ»

```
s = "информатика"  
print(s[-2]+s[3]+s[-4])
```

# Срезы

```
s = "0123456789"  
s1 = s[3:8]           # "34567"
```



# Срезы строк

```
s = "0123456789"  
s1 = s[:8] # "01234567"
```

от начала строки

```
s = "0123456789"  
s1 = s[3:] # "3456789"
```

до конца строки

```
s1 = s[::-1] # "9876543210"
```

реверс строки

# Операции со строками

## Срезы с отрицательными индексами:

```
s = "0123456789"
```

```
s1 = s[ : -2] # "01234567"
```

`len(s) - 2`

```
s = "0123456789"
```

```
s1 = s[ -6 : -2] # "4567"
```

`len(s) - 6`

`len(s) - 2`

# Операции со строками

Удаление:

```
s = "0123456789"  
s1 = s[:3] + s[9:] # "0129"  
"012"    "9"
```

Вставка:

```
s = "0123456789"  
s1 = s[:3] + "ABC" + s[3:]  
"012ABC3456789"
```

# Задачи

---

**«3»:** Ввести с клавиатуры пароль (символьную строку), если его длина меньше, чем **6** символов, вывести сообщение «Слишком короткий пароль!», иначе вывести сообщение «OK».

**Пример:**

Ведите пароль :

**12345**

Слишком короткий пароль !

**Пример:**

Ведите пароль :

**123456789**

OK .

# Задачи

---

**«4»:** Ввести с клавиатуры пароль (символьную строку). Если его длина меньше, чем 6 символов, вывести сообщение «Слишком короткий пароль!». Если пароль начинается с букв «qwerty» вывести сообщение «Ненадёжный пароль!». Если ошибок не было, вывести сообщение «OK».

**Пример:**

Ведите пароль :

**qwerty12345**

Ненадёжный пароль !

**Пример:**

Ведите пароль :

**asdUTY7sakh**

OK .

# Задачи

---

**«5»:** Ввести с клавиатуры имя файла. Если расширение имени файла – `htm`, `html` или `php`, выдать сообщение «Это веб-страница!», иначе выдать сообщение «Что-то другое.»

**Пример:**

Ведите имя файла:

`C:\DOC\Сайт\index.html`

Это веб-страница !

**Пример:**

Ведите имя файла:

`C:\Документы\Приказ.doc`

Что-то другое .

# Программирование на языке Python

## 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?

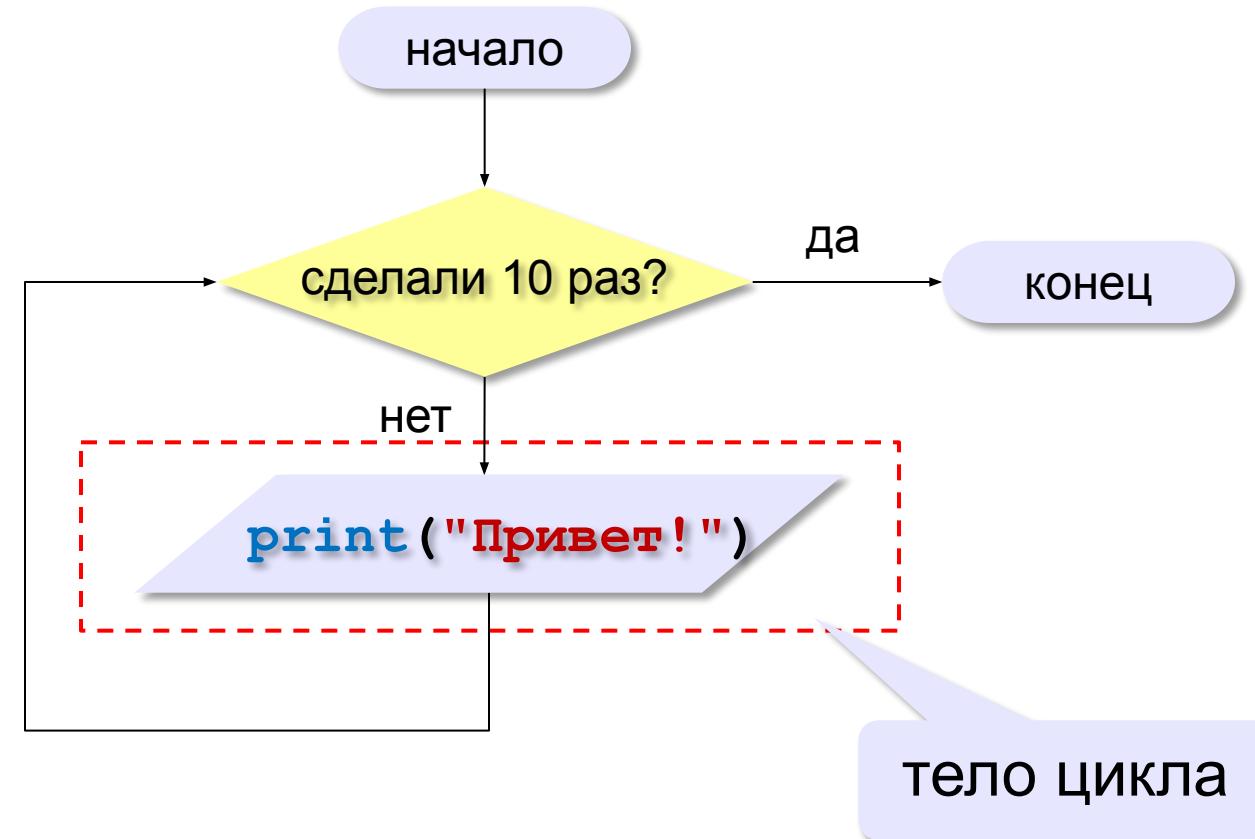
# Повторения в программе

```
print("Привет")
print("Привет")
...
print("Привет")
```



Что плохо?

# Блок-схема цикла



# Как организовать цикл?

```
счётчик = 0
пока счётчик < 10:
    print("Привет")
    увеличить счётчик на 1
```

```
k = 0
while k < 10:
    print( "Привет" )
    k += 1
```



Как по-другому?

```
счётчик = 10
пока счётчик > 0:
    print("Привет")
    уменьшить счётчик на 1
```

```
k = 10
while k > 0:
    print( "Привет" )
    k -= 1
```

# Сколько раз выполняется цикл?

```
a = 4; b = 6  
while a < b: a += 1
```

2 раза  
a = 6

```
a = 4; b = 6  
while a < b: a += b
```

1 раз  
a = 10

```
a = 4; b = 6  
while a > b: a += 1
```

0 раз  
a = 4

```
a = 4; b = 6  
while a < b: b = a - b
```

1 раз  
b = -2

```
a = 4; b = 6  
while a < b: a -= 1
```

зацикливание

# 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
```



Как отсечь последнюю цифру?

```
n = n // 10
```



Как увеличить счётчик на 1?

```
счётчик = счётчик + 1
```

n	счётчик
1234	0

```
счётчик += 1
```

# Цикл с условием

начальное значение  
счётчика

заголовок  
цикла

условие  
продолжения

```
count = 0
while n > 0:
    n = n // 10
    count += 1
```

тело цикла



Цикл с предусловием – проверка на входе в цикл!

# Задачи

---

**«3»:** Ввести с клавиатуры количество повторений и вывести столько же раз какое-нибудь сообщение.

**Пример:**

Сколько раз:

5

Привет!

Привет!

Привет!

Привет!

Привет!

# Задачи

---

**«4»:** Ввести с клавиатуры натуральное число и определить, сколько раз в его записи встречается цифра 1.

**Пример:**

**Введите число :**

**51211**

**3**

**«5»:** Ввести с клавиатуры натуральное число и найти сумму значений его цифр.

**Пример:**

**Введите число :**

**1234**

**Сумма цифр 10**

# Задачи

---

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

**Пример:**

Ведите натуральное число:

12342

Нет.

**Пример:**

Ведите натуральное число:

12245

Да.

# Алгоритм Евклида

**Алгоритм Евклида.** Чтобы найти НОД двух натуральных чисел, нужно вычесть из большего числа меньшее до тех пор, пока они не станут равны. Это число и есть НОД исходных чисел.

$$\text{НОД}(14,21) = \text{НОД}(14,7) = \text{НОД}(7, 7) = 7$$

```
пока a != b:  
    если a > b:  
        a -= b # a = a - b  
    иначе:  
        b -= a # b = b - a
```

```
while a != b:  
    if a > b:  
        a -= b  
    else:  
        b -= a
```

$$\text{НОД}(1998,2) = \text{НОД}(1996,2) = \dots = \text{НОД}(2, 2) = 2$$

# Алгоритм Евклида

**Модифицированный алгоритм Евклида.** Заменять большее число на остаток от деления большего на меньшее до тех пор, пока меньшее не станет равно нулю. Другое (ненулевое) число и есть НОД чисел.

$$\text{НОД}(1998, 2) = \text{НОД}(0, 2) = 2$$

пока `a != 0 and b != 0:`

если `a > b:`

`a = a % b`

иначе:

`b = b % a`

если `a != 0:`

вывести `a`

иначе:

вывести `b`



Какое условие?



Как вывести результат?

# Задачи

**«3»:** Ввести с клавиатуры два натуральных числа и найти их НОД с помощью алгоритма Евклида.

**Пример:**

Ведите два числа:

21 14

НОД(21, 14) = 7

**«4»:** Ввести с клавиатуры два натуральных числа и найти их НОД с помощью **модифицированного** алгоритма Евклида. Заполните таблицу:

a	64168	358853	6365133	17905514	549868978
b	82678	691042	11494962	23108855	298294835
НОД(a, b)					

## Задачи

---

**«5»:** Ввести с клавиатуры два натуральных числа и сравнить количество шагов цикла для вычисления их НОД с помощью обычного и модифицированного алгоритмов Евклида.

**Пример:**

Ведите два числа:

1998 2

НОД(1998, 2)=2

Обычный алгоритм: 998

Модифицированный: 1

# Обработка строк в цикле

Задача. Ввести строку и определить, сколько в ней цифр.

счётчик = 0

для каждого символа строки:

если символ – цифра:

счётчик += 1

s = input()

k = 0

for c in s:

if c.isdigit():

k += 1

для всех символов в  
строке

если c – это цифра

## Проверка символов

```
if c.isdigit():
    print("Цифра")
```

```
if c.isalpha():
    print("Буква")
```

```
if c.islower():
    print("Строчная буква")
```

```
if c.isupper():
    print("Заглавная буква")
```

```
if c in ["а", "б"]:
    print("Это а или б")
```

## Задачи

**«3»:** Ввести с клавиатуры число в двоичной системе счисления. Определить, сколько в его записи единиц и сколько нулей.

**Пример:**

Ведите число :

1010100

Нулей : 4

Единиц : 3

**«4»:** Ввести с клавиатуры символьную строку. Если это правильная запись двоичного числа, вывести сообщение «Да», иначе вывести сообщение «Нет».

**Пример:**

Ведите число :

1010100

Да .

Ведите число :

abcd10

Нет .

## Задачи

---

**«5»:** Ввести с клавиатуры символьную строку и составить новую строку, удалив из исходной все пробелы.

**Пример:**

**Введите строку:**

**Вася пошел гулять .**

**Васяпошелгулять .**

# Цикл с переменной

Задача. Вывести 10 раз слово «Привет!».



Можно ли сделать с циклом «пока»?

```
i = 0
while i < 10:
    print("Привет!")
    i += 1
```

Цикл с переменной:

```
for i in range(10):
    print("Привет!")
```

в диапазоне  
[0, 10)



Не включая 10!

`range(10)` → 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

# Цикл с переменной

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



Как сделать с циклом «пока»?

```
k = 1
while k <= 10
    print ( 2**k )
    k += 1
```

возвведение  
в степень

Цикл с переменной:

```
for k in range(1,11) :
    print ( 2**k )
```

в диапазоне  
[1, 11)



Не включая 11!

`range(1,11)` → 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

# Цикл с переменной: другой шаг

10, 9, 8, 7, 6, 5, 4, 3, 2, 1      шаг

```
for k in range(10, 0, -1)
    print ( k**2 )
```

100

81

64

49

36

25

16

9

1

9

25

49

81



Что получится?

1, 3, 5, 7, 9

```
for k in range(1, 11, 2):
    print ( k**2 )
```

100

81

64

49

36

25

16

9

1

4

1

# Сколько раз выполняется цикл?

```
a = 1  
for k in range( 3 ): a += 1
```

a = 4

```
a = 1  
for k in range( 3, 1 ): a += 1
```

a = 1

```
a = 1  
for k in range( 1, 3, -1 ): a += 1
```

a = 1

```
a = 1  
for k in range( 3, 1, -1 ): a += 1
```

a = 3

## Задачи

---

**«3»:** Ипполит задумал трёхзначное число, которое при делении на 15 даёт в остатке 11, а при делении на 11 даёт в остатке 9. Найдите все такие числа.

**«4»:** Вводится натуральное число N. Программа должна найти **факториал** (обозначается как **N!**) – произведение всех натуральных чисел от 1 до N.  
Например,  $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$ .

**Пример:**

Введите число:

5

5!=120.

## Задачи

---

**«5»:** 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.

# Программирование на языке Python

## Массивы (списки)

# Что такое массив?



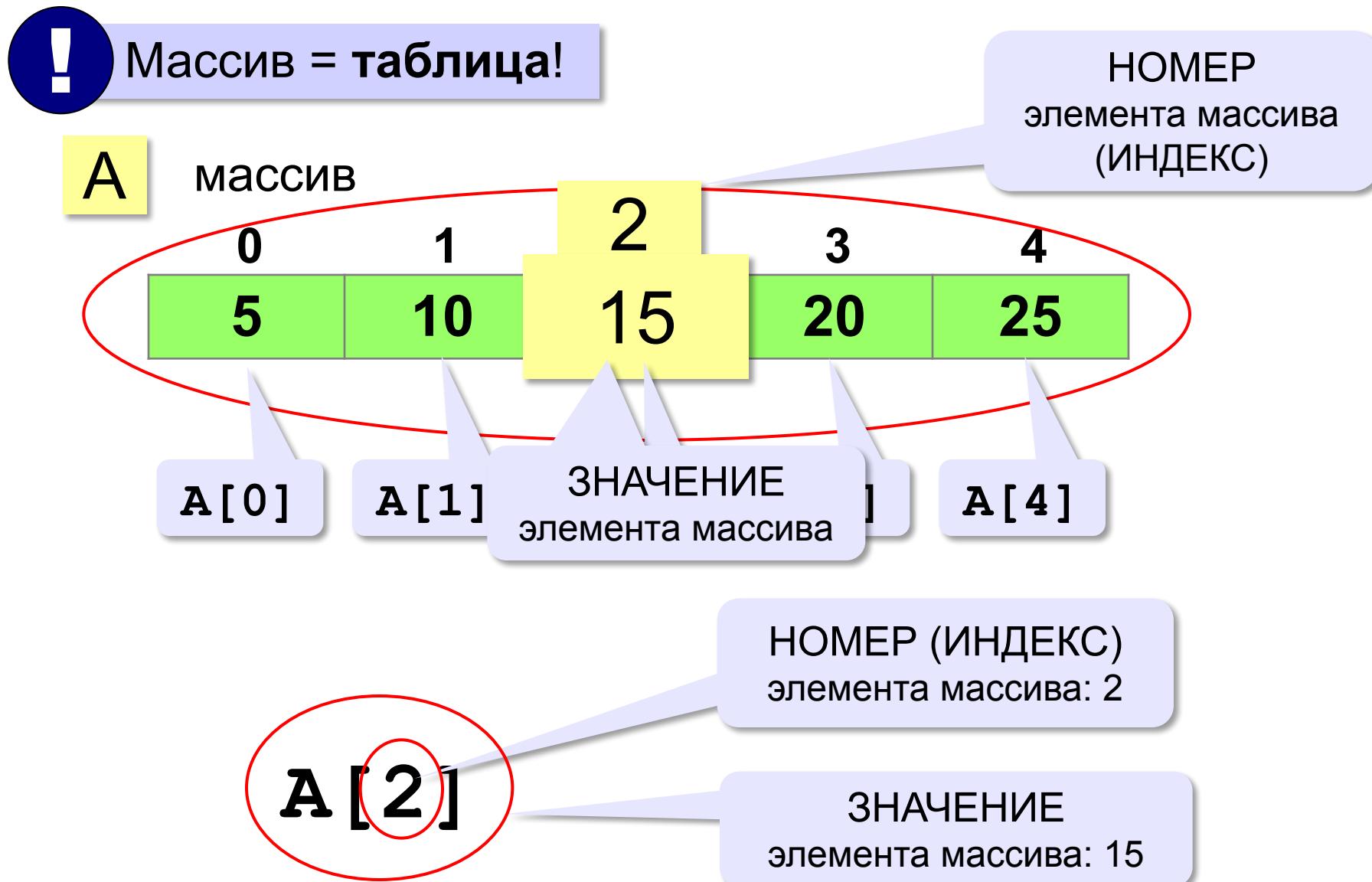
Как ввести 10000 переменных?

**Массив** – это группа переменных одного типа, расположенных в памяти рядом (в соседних ячейках) и имеющих общее имя. Каждая ячейка в массиве имеет уникальный номер (индекс).

## Надо:

- выделять память
- записывать данные в нужную ячейку
- читать данные из ячейки

# Что такое массив?



# Массивы в Python: списки

```
A = [1, 3, 4, 23, 5]
```

```
A = [1, 3] + [4, 23] + [5]
```

```
[1, 3, 4, 23, 5]
```

```
A = [0]*10
```



Что будет?

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

## Создание массива из N элементов:

```
N = 10
```

```
A = [0]*N
```

# Заполнение массива

Целыми числами (начиная с 0!):

```
N = 10      # размер массива
A = [0]*N   # выделить память
for i in range(N):
    A[i] = i
```

В краткой форме:

```
N = 10      # размер массива
A = [i for i in range(N)]
```



Как заполнить, начиная с 1?



Как заполнить квадратами чисел?

# Заполнение случайными числами

из библиотеки (модуля)  
random

взять функцию randint

```
from random import randint
N = 10          # размер массива
A = [0]*N      # выделить память
for i in range(N):
    A[i] = randint(20,100)
```

В краткой форме:

```
from random import randint
N = 10
A = [ randint(20,100)
      for i in range(N) ]
```

# Вывод массива на экран

Как список:

```
print( A ) [1, 2, 3, 4, 5]
```

В строчку через пробел:

```
for i in range(N):  
    print( A[i], end=" " ) 1 2 3 4 5
```

или так:

```
for x in A:  
    print( x, end=" " )
```

пробел после  
вывода  
очередного числа

1 2 3 4 5

или так:

```
print( *A ) ⇔ print(1, 2, 3, 4, 5)
```

разбить список  
на элементы

## Задачи

---

«3»: Ввести два натуральных числа **a** и **b** (**a < b**) и заполнить массив из 10 элементов случайными числами в диапазоне от **a** до **b**.

**Пример:**

Ведите границы диапазона:

5 10

10 9 10 6 8 5 9 6 10 9

«4»: Ввести два натуральных числа **a** и **b** и заполнить массив из 10 элементов случайными числами в диапазоне между **a** и **b** (**a** может быть больше **b**).

**Пример:**

Ведите границы диапазона:

10 5

10 9 10 6 8 5 9 6 10 9

## Задачи

---

**«5»:** Ввести два натуральных числа **a** и **b** и заполнить массив из 10 элементов: первая половина массива заполняется случайными числами в диапазоне между **a** и **b** (**a** может быть больше **b**), а вторая половина массива содержит их квадраты в том же порядке.

**Пример:**

Ведите границы диапазона:

10 5

5 8 7 10 6 25 64 49 100 36

# Ввод массива с клавиатуры

## Создание массива:

```
N = 10
```

```
A = [0]*N
```

## Ввод по одному элементу в строке:

```
for i in range(N) :  
    A[i] = int( input() )
```

## или кратко:

```
A = [int(input())  
      for i in range(N)]
```

# Ввод массива с клавиатуры

Ввод всех чисел в одной строке:

```
data = input()      # "1 2 3 4 5"  
s = data.split()   # ["1", "2", "3", "4", "5"]  
A = [ int(x) for x in s ]  
                      # [1,2,3,4,5]
```

или так:

```
A = [int(x) for x in input().split()]
```

# Как обработать все элементы массива?

**Создание массива:**

**N = 5**

**A = [0]\*N**

**Обработка:**

```
# обработать A[0]
# обработать A[1]
# обработать A[2]
# обработать A[3]
# обработать A[4]
```



- 1) если N велико (1000, 1000000)?
- 2) при изменении N программа не должна меняться!

# Как обработать все элементы массива?

## Обработка с переменной:

```
i = 0  
# обработать A[i]  
i += 1  
# обработать A[i]  
  
i += 1
```

## Обработка в цикле:

```
i = 0  
while i < N:  
    # обработать A[i]  
    i += 1
```

## Цикл с переменной:

```
for i in range(N):  
    # обработать A[i]
```

# Перебор элементов

Общая схема (можно изменять  $A[i]$ ):

```
for i in range(N):
    ... # сделать что-то с A[i]
```

```
for i in range(N):
    A[i] += 1
```

Если не нужно изменять  $A[i]$ :

```
for x in A:
    ... # сделать что-то с x
```

$x = A[0], A[1], \dots, A[N-1]$

```
for x in A:
    print(x)
```

# Что выведет программа?

```
A = [2, 3, 1, 4, 6, 5]
```

```
print( A[3] ) # 4
```

```
print( A[0]+2*A[5] ) # 12
```

```
A[1] = A[0] + A[5] # 7
```

```
print( 3*A[1]+A[4] ) # 27
```

```
A[2] = A[1]*A[4] # 18
```

```
print( 2*A[1]+A[2] ) # 22
```

```
for k in range(6):
    A[k] += 2      # [4,5,3,6,8,7]
print( 2*A[3]+3*A[4] ) # 36
```

# Подсчёт нужных элементов

**Задача.** В массиве записаны данные о росте баскетболистов. Сколько из них имеет рост больше 180 см, но меньше 190 см?



Как решать?

```
count = 0
for x in A:
    if 180 < x and x < 190:
        count += 1
```

# Перебор элементов

Задача. Найти сумму чётных элементов массива.

```
summa = 0
for x in A:
    if x % 2 == 0:
        summa += x
print ( summa )
```



Как определить, что  
элемент чётный?

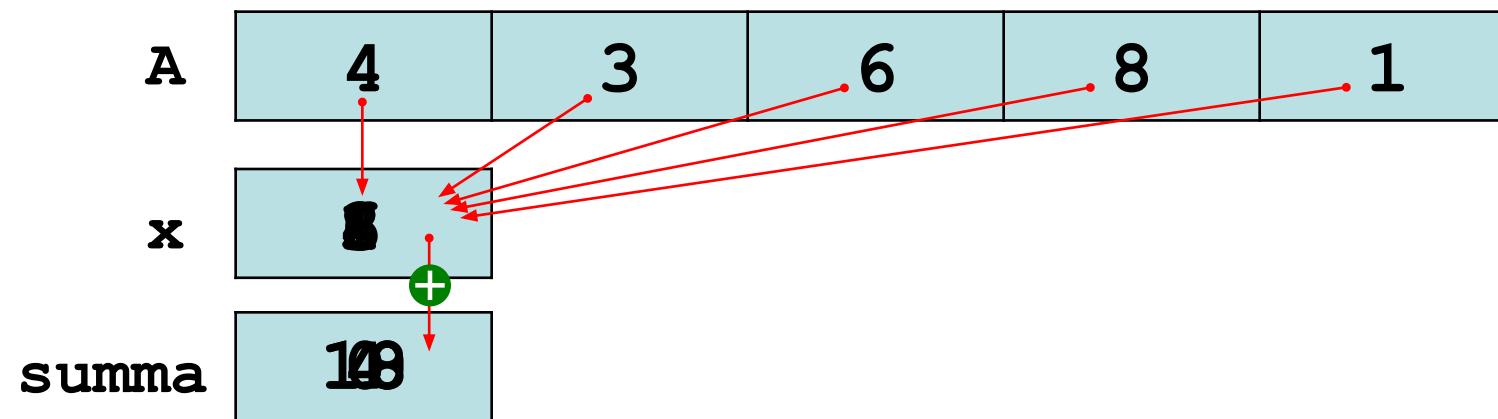
или так:

```
B = [x for x in A
      if x % 2 == 0]
print ( sum(B) )
```

сумма массива В

# Как работает цикл?

```
summa = 0  
for x in A:  
    if x % 2 == 0:  
        summa += x
```



# Среднее арифметическое

Задача. Найти среднее арифметическое элементов массива, которые оканчиваются на цифру 5.

```
count = 0
summa = 0
for x in A:
    if x % 10 == 5:
        count += 1
        summa += x
print ( summa/count )
```



Как определить, что оканчивается на 5?

среднее  
арифметическое

или так:

отбираем нужные

```
B = [ x for x in A
      if x % 10 == 5]
print ( sum(B)/len(B) )
```

## Задачи

**«3»:** Введите массив из 5 элементов с клавиатуры и найдите среднее арифметическое его значений.

**Пример:**

**Массив :**

1 2 3 4 5

**Среднее арифметическое 3.000**

**«4»:** Заполните массив из 10 элементов случайными числами в интервале [0,100] и подсчитайте отдельно среднее значение всех элементов, которые <50, и среднее значение всех элементов, которые  $\geq 50$ .

**Пример:**

**Массив :**

3 2 52 4 60 50 1 2 60 58 6

**Ср. арифм. элементов < 50: 3.000**

**Ср. арифм. элементов  $\geq 50: 56.000$**

# Задачи

---

**«5»:** Введите размер массива N и заполните массив из N элементов **числами Фибоначчи**. Первые два числа Фибоначчи равны 1, а каждое следующее равно сумме двух предыдущих.

**Пример:**

Ведите размер массива:

6

Числа Фибоначчи:

1 1 2 3 5 8

# Программирование на языке Си

## Поиск в массиве

# Поиск в массиве

Найти элемент, равный X:

```
i = 0
while A[i] != x:
    i += 1
print( "A[", i, "] =", x, sep = "" )
```



Что плохо?

```
i = 0
while i < N and A[i] != x:
    i += 1
if i < N:
    print( "A[", i, "] =", x, sep = "" )
else:
    print( "Не нашли!" )
```



Что если такого нет?

# Поиск в массиве

Вариант с досрочным выходом:

номер найденного  
элемента

```
nX = -1
for i in range( N ):
    if A[i] == x:
        nX = i
        break
if nX >= 0:
    print( "A[ ", nX, " ] =", x, sep = "" )
else:
    print( "Не нашли!" )
```

досрочный  
выход из цикла

# Поиск в массиве

Варианты в стиле Python:

```
for i in range( N ):
    if A[i] == x:
        print( "A[", i, "]=", x, sep="" )
        break
    else:
        print( "Не нашли!" )
```

если не было досрочного выхода из цикла

```
if x in A:
    nX = A.index(x)
    print( "A[", nX, "]=", x, sep="" )
else:
    print( "Не нашли!" )
```

## Задачи

---

**«3»:** Заполните массив из 10 элементов случайными числами в диапазоне [100,200]. Найдите первое число в массиве, у которого последняя цифра – 2. Если такого числа нет, вывести ответ «Не нашли».

**Пример:**

**Массив:**

131 180 117 170 162 111 109 155 159 137

**Нашли:** A[4]=162

**Пример:**

**Массив:**

131 180 117 170 163 111 109 155 159 137

**Не нашли.**

# Задачи

**«4»:** Заполните массив из 10 элементов случайными числами в интервале [0,5]. Введите число X и найдите все значения, равные X.

**Пример:**

**Массив:**

1 2 3 1 2 4 2 5 1 3

**Что ищем:**

2

A[2]=2

A[5]=2

**Пример:**

**Массив:**

1 2 3 1 2 4 2 5 1 3

**Что ищем:**

6

**Не нашли.**

# Задачи

---

**«5»:** Заполните массив из 10 элементов случайными числами в интервале [0,5]. Найдите пару одинаковых элементов, стоящих рядом.

**Пример:**

**Массив:**

1 2 3 3 4 1 5 1 3 2

$A[2]=A[3]=3$

**Пример:**

**Массив:**

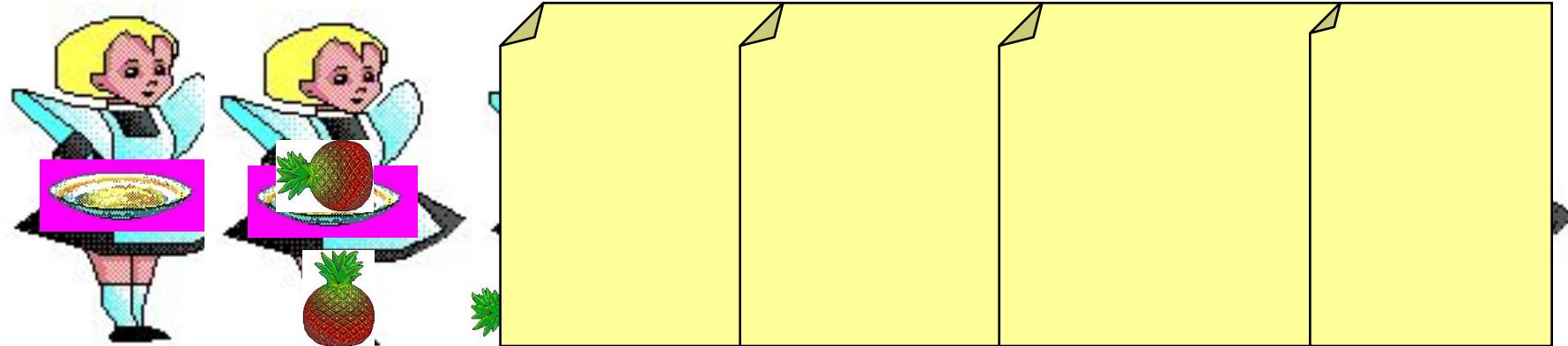
1 2 3 4 2 1 5 1 2 3

**Нет.**

# Максимальный элемент

**Задача:** найти в массиве максимальный элемент.

**Алгоритм:**



**Решение:**

- 1) считаем, что первый элемент – максимальный
- 2) просмотреть остальные элементы массива:  
если очередной элемент  $> M$ ,  
то записать  $A[i]$  в  $M$
- 3) вывести значение  $M$

# Максимальный элемент

```
M = A[0]
for i in range(1, N):
    if A[i] > M:
        M = A[i]
print(M)
```



Если `range(N)` ?

Варианты в стиле Python:

```
M = A[0]
for x in A:
    if x > M:
        M = x
```



Как найти его номер?

```
M = max(A)
```

# Максимальный элемент и его номер

```
M = A[0]; nMax = 0
for i in range(1,N):
    if A[i] > M:
        M = A[i]
        nMax = i
print ( "A[", nMax, "]=", M, sep = "" )
```



Что можно улучшить?



По номеру элемента можно найти значение!

```
nMax = 0
for i in range(1,N):
    if A[i] > A[nMax]:
        nMax = i
print ( "A[", nMax, "]=", A[nMax], sep = "" )
```

# Максимальный элемент и его номер

Вариант в стиле Python:

```
M = max (A)  
nMax = A.index (M)  
print ( "A[ ", nMax, "]=", M, sep = " " )
```

номер заданного  
элемента (первого из...)

## Задачи (без функций **min** и **max**)

**«3»:** Заполнить массив из 10 элементов случайными числами в интервале [10,100] и найти минимальный и максимальный элементы массива и их номера.

**Пример:**

**Массив:**

39 52 84 77 45 32 19 38 49 85

**Минимальный элемент:** A[6]=19

**Максимальный элемент:** A[9]=85

## Задачи (без функций **min** и **max**)

---

**«4»:** Заполнить массив из 10 элементов случайными числами в интервале [10,100] и найти минимальный и максимальный элементы из **чётных** элементов массива.

**Пример:**

**Массив:**

39 52 84 77 45 32 19 38 49 85

**Минимальный чётный:** 32

**Максимальный чётный:** 84

## Задачи (без функции `max`)

---

**«5»:** Ввести с клавиатуры массив из 5 элементов и найти два максимальных элемента массива и их номера.

**Пример:**

**Массив:**

5 5 3 4 1

**Максимальный элемент:  $A[1]=5$**

**Второй максимум:  $A[2]=5$**

# Задачи

---

**«6»:** Введите массив с клавиатуры и найдите (за один проход) количество элементов, имеющих максимальное значение.

**Пример:**

**Массив:**

3 4 5 5 3 4 5

**Максимальное значение 5**

**Количество элементов 3**

# Конец фильма

---

**ПОЛЯКОВ Константин Юрьевич**

д.т.н., учитель информатики

ГБОУ СОШ № 163, г. Санкт-Петербург

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