Программирование на языке Python

Алгоритм и его свойства
Простейшие программы
Вычисления
Ветвления
Символьные строки

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.

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

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

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

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.

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

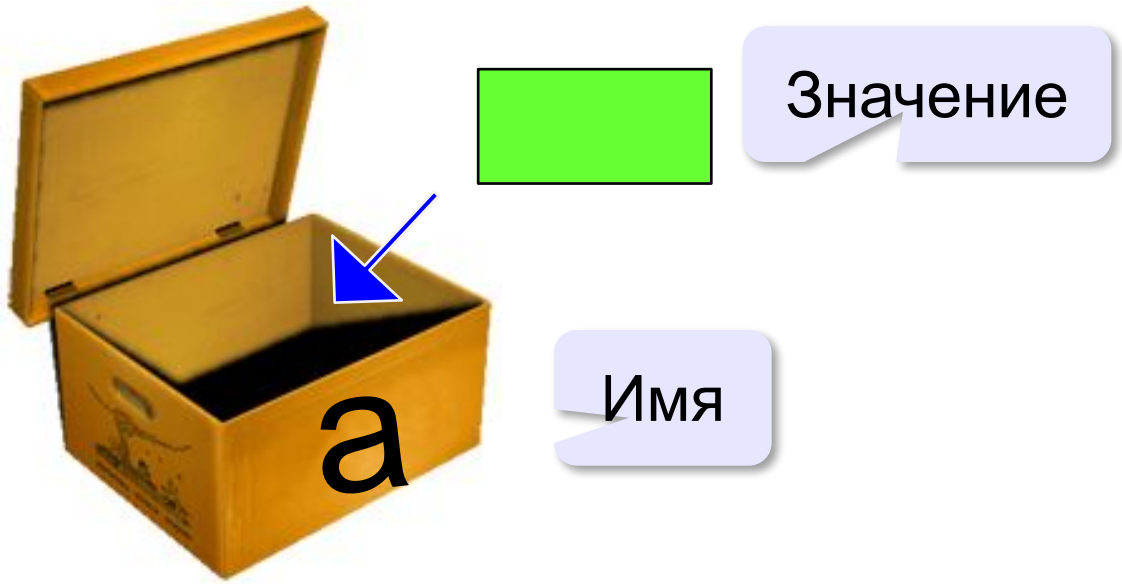
```
ввести два числа
вычислить их сумму
вывести сумму на экран
```

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?

AXby 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?

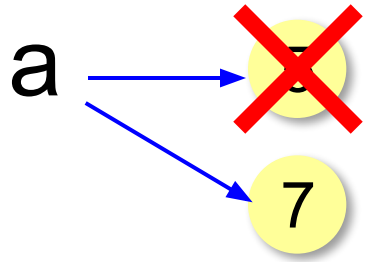
оператор присваивания

!

При записи нового значения старое удаляется из памяти!

```
a = 5
```

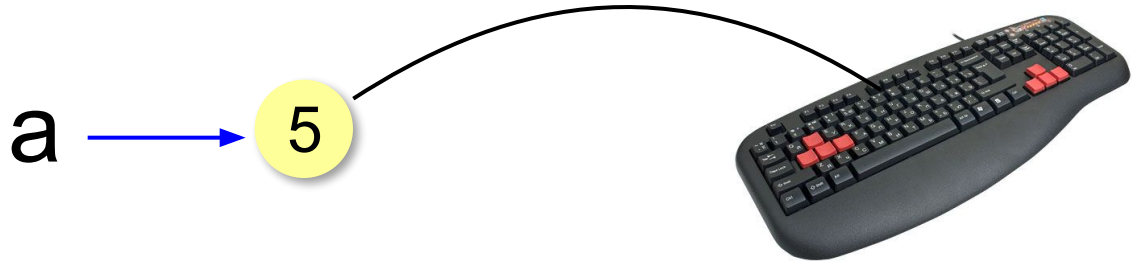
```
a = 7
```



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

ввести строку с клавиатуры и связать с переменной **a**

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
```

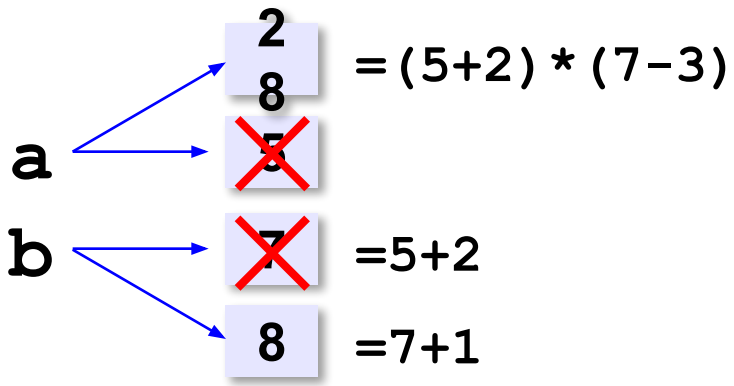
подсказка

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

компьютер

Протокол:

```
Введите два целых числа
25
30
25+30=55
```

пользователь

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

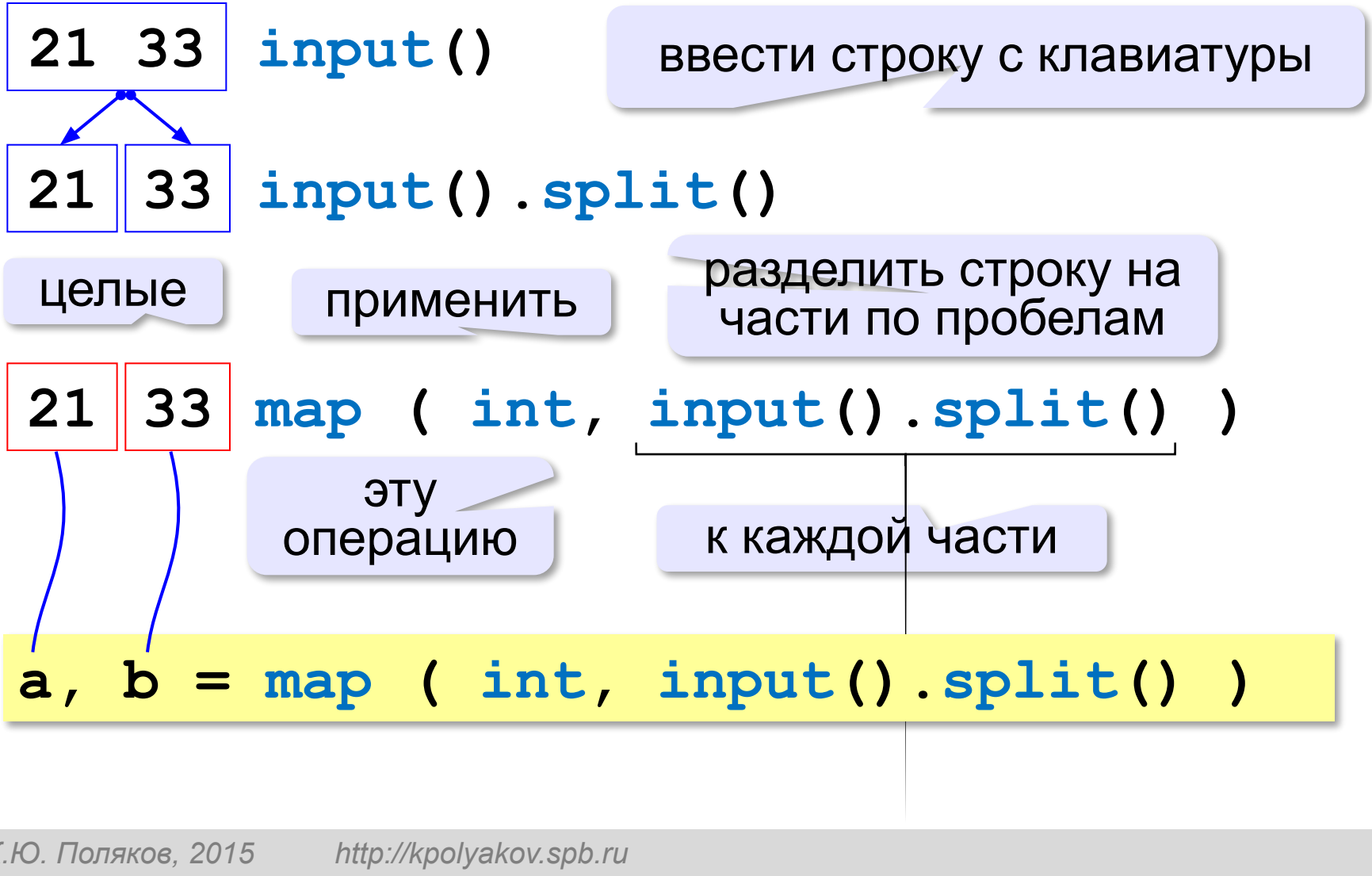
```
a += b   # a = a + b
a -= b   # a = a - b
a *= b   # a = a * b
a /= b   # a = a / b
a //= b  # a = a // b
a %= b   # a = a % b
```

```
a += 1
```

увеличение на 1

Ввод двух значений в одной строке

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




```
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: 318458191041

209938992481

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

Линейный конгруэнтный генератор

```
X = (a*X+b) % c | интервал от 0 до c-1
```

```
X = (X+3) % 10 | интервал от 0 до 9
```

X = 0 → 3 → 6 → 9 → 2 → 5 → 8

8 → 1 → 4 → 7 → 0

зерно

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:

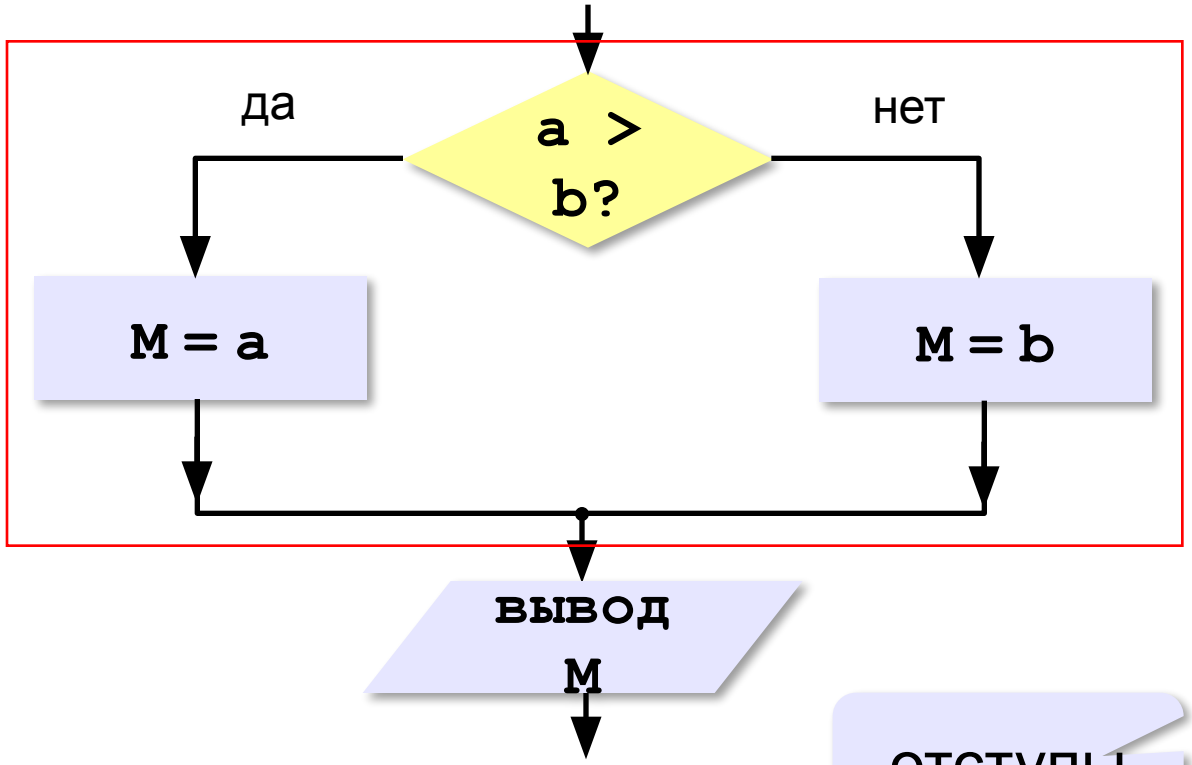
```
Получено число 123
сотни: 1
десятки: 2
единицы: 3
```

Программирование на языке Python

Ветвления

Условный оператор

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



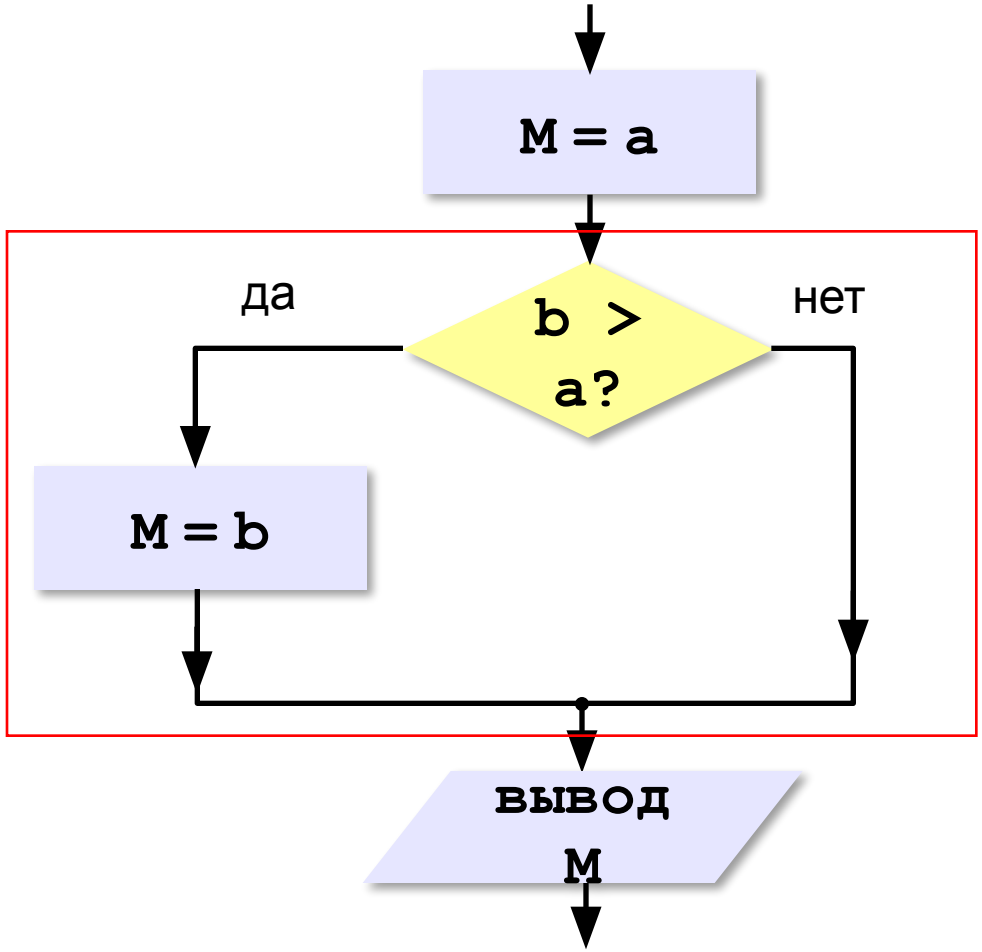
полная форма ветвления

? Если a = b?

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

отступы

Условный оператор: неполная форма




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

неполная форма ветвления

Решение в стиле Python:

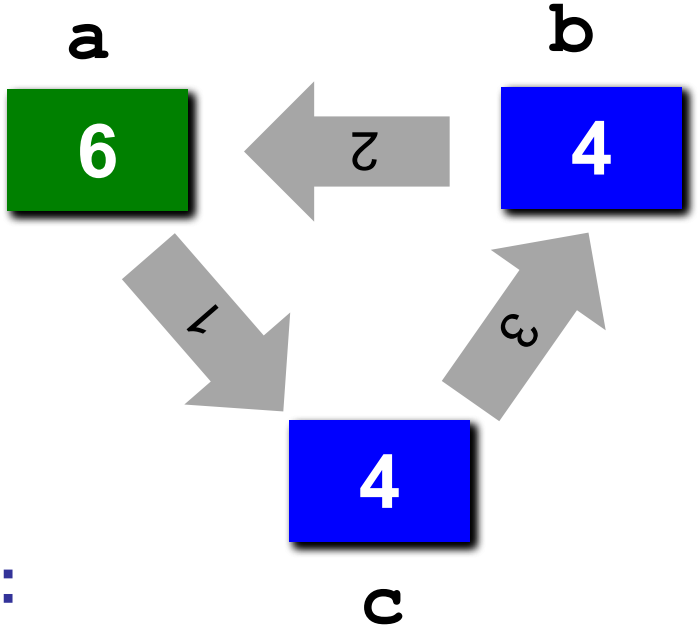
```
M = max(a, b)
```

```
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:

```
Введите два целых числа:
1 5
Наибольшее число 5
Наименьшее число 1
```

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

Пример:

```
Введите 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:

```
Введите рост трёх спортсменов:
165 170 172
По росту.
```

Пример:

```
Введите 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:

```
Введите номер месяца:
5
Весна.
```

Пример:

```
Введите 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:

```
Введите возраст: 18
Вам 18 лет.
```

Пример:

```
Введите возраст: 21
Вам 21 год.
```

Пример:

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

"Привет, Вася!"

Умножение:

```
s = "АУ"
s5 = s*5
```

```
s5 = s + s + s + s + s
```

АУАУАУАУАУ

Что получим?

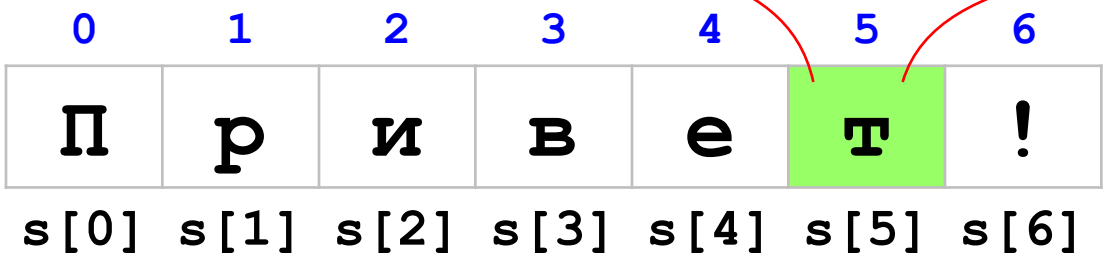
Символьные строки

Вывод символа на экран:

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

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

s[len(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:

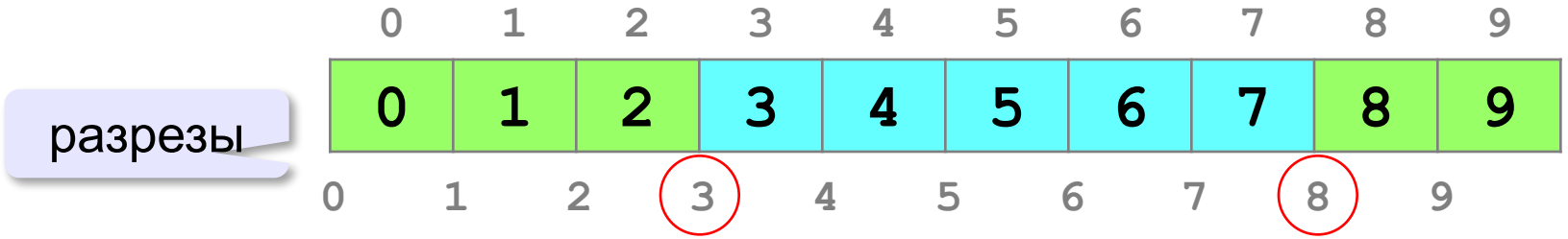
```
s1 = s + "a"
```

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 «ОК».

Пример:

```
Введите пароль:
12345
Слишком короткий пароль!
```

Пример:

```
Введите пароль:
123456789
ОК.
```

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 «ОК».

Пример:

```
Введите пароль:
qwerty12345
Ненадёжный пароль!
```

Пример:

```
Введите пароль:
asdUTY7sakh
ОК.
```

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:

```
Введите имя файла:
C:\DOC\Сайт\index.html
Это веб-страница!
```

Пример:

```
Введите имя файла:
C:\Документы\Приказ.doc
Что-то другое.
```