

Python



Новая игра

Автор

Выйти

О чем курс?

Почему Python?

Что от вас требуется?

Ст.КБ-4: Арслан Тарланов

@levbrave

Lev.brave@gmail.com



Языки программирования

Язык программирования – это набор строгих правил, согласно которым компьютер может «понимать» команды и выполнять их. Текст программы, написанной на любом языке программирования, называется программным кодом или просто кодом.

JavaScript

Python

Java

C#

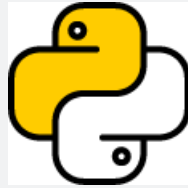
ActionScript

Perl

COBOL

Pascal

Интерпретатор Python



Интерпретатор Python идет по тексту программы, анализирует и переводит каждую отдельную команду в форму понятную компьютеру



Программа на языке Python



Кажется, мне
надо напечатать
приветствие...

Вывод данных
Программа "hello"

```
print('Hello')
```



```
print('Hello')
```

Hello

```
print('Hello')  
print('How', 'are', 'you?')  
print('How are you?')
```

```
Hello  
How are you?  
How are you?
```

```
cry = 'Ayyyy!'  
print(cry)  
print(cry)
```

```
Ayyyyy!  
Ayyyyy!
```

```
cry = 'Ayyyy!'  
print(cry)  
print(cry)  
cry = 'Yaaaaa!'  
print(cry)
```

```
Ayyyyy!  
Ayyyyy!  
Yaaaaa!
```

```
cry = 'Ayyyyy!'  
print(cry)  
print(cry)  
cry = 'Yaaaaa!'  
print(cry)  
cry2 = cry  
print(cry2)
```

```
Ayyyyy!  
Ayyyyy!  
Yaaaaa!  
Yaaaaa!
```

Ввод данных


```
print('Как тебя зовут?')  
name = input()  
print('Привет, ', name)
```



```
import math, sys;
```

Before

```
def example1():
```

```
####This is a long comment. This should be wrapped to fit within 72 characters.
some_tuple=( 1,2, 3,'a' );
some_variable={'long':'Long code lines should be wrapped within 79 characters.',
'other':[math.pi, 100,200,300,9876543210,'This is a long string that goes on'],
'more':{'inner':'This whole logical line should be wrapped.',some_tuple:[1,
20,300,40000,500000000,6000000000000000000]}}
return (some_tuple, some_variable)
```

```
import math
```

```
import sys
```

After

```
def example1():
```

```
# This is a long comment. This should be wrapped to fit within 72
# characters.
some_tuple = (1, 2, 3, 'a')
some_variable = {
    'long': 'Long code lines should be wrapped within 79 characters.',
    'other': [
        math.pi,
        100,
        200,
        300,
        9876543210,
        'This is a long string that goes on'],
    'more': {
        'inner': 'This whole logical line should be wrapped.',
        some_tuple: [
            1,
            20,
            300,
            40000,
            500000000,
            6000000000000000000]}}
return (some_tuple, some_variable)
```



git for windows



Python Code Automatic PEP8 Styling

```
print('Введите фамилию:')
surname = input()
print('Введите имя:')
name = input()
print(name, surname)
print('Введите новое имя:')
new_name = input()
print(name, surname)
print(new_name, name)
name = new_name
print(new_name, name)
print(name, surname)
```

```
Введите фамилию:
Максимов
Введите имя:
Артем
Артем Максимов
Введите новое имя:
Василий
Артем Максимов
Василий Артем
Василий Василий
Василий Максимов
```



Условный оператор

Условный оператор

Условный оператор используется, когда некая часть программы должна быть выполнена, только если верно какое-либо условие. Для записи условного оператора используются ключевые слова `if` и `else` («если», «иначе»), двоеточие, а также отступ в четыре пробела.

`if` условие:

действие, если условие верно

`else:`

действие, если условие неверно

```
print('Введите пароль:')  
password = input()  
if password == 'qwerty':  
    print('Доступ открыт.')  
else:  
    print('Ошибка, доступ закрыт!')
```



Логические операции

Логические операции

Чтобы задать, что два условия должны выполняться одновременно - используем **and** («и»), если достаточно выполнения одного из двух вариантов (или оба сразу), то используем **or** («или»), а если нужно убрать какой-то вариант, то используем **not** («не»)

Приоритет операций:


1. not
2. and
3. or

```
print('Как называются первая и последняя')
print('буквы греческого алфавита?')
greek_letter_1 = input()
greek_letter_2 = input()
if greek_letter_1 == 'альфа' and greek_letter_2 == 'омега':
    print('Верно.')
else:
    print('Неверно.')

print('Как греки или римляне называли')
print('главу своего пантеона - бога грома?')
ancient_god = input()
if ancient_god == 'Зевс' or ancient_god == 'Юпитер':
    print('Верно.')
else:
    print('Неверно.')
```

```
# Ex. #1
if (brother1 == 'Ромул' and brother2 == 'Рем' or brother1 == 'Кастор'
    and (brother2 == 'Поллукс' or brother2 == 'Полидевк')):
    print('Верно.')
else:
    print('Неверно.')
```

```
# Ex. #2
if brother1 == 'Ромул' and brother2 == 'Рем' or brother1 == 'Кастор' \
    and (brother2 == 'Поллукс' or brother2 == 'Полидевк'):
    print('Верно.')
else:
    print('Неверно.')
```



Блоки кода
Вложенные условия

```
print('Представься, о незнакомец!')
name = input()
if name == 'Цезарь' or name == 'Caesar':
    print('Аве, Цезарь!')
    print('Слава императору!')
else:
    print('Приветик.')
    print('Погода сегодня хорошая.')
print('Засим - заканчиваем.')
```

```
print('Представься, о незнакомец!')
name = input()
if name == 'Цезарь' or name == 'Caesar':
    print('Аве, Цезарь!')
    print('В честь какого бога устроим празднество?')
    god = input()
    if god == 'Юпитер':
        print('Ура Громовержцу!')
    elif god == 'Минерва':
        print('Ура мудрой воительнице!')
    else:
        print('Бога по имени', god,
              'мы не знаем, но слово Цезаря – закон.')
    print('Слава императору!')
else:
    print('Приветик.')
    print('Погода сегодня хорошая.')
print('Засим - заканчиваем.')
```

```
if условие1:
```

```
    ...
```

```
elif условие2:
```

```
    ...
```

```
elif условие3:
```

```
    ...
```

Цепочка elif'ов гораздо
удобнее, чем цепочка else:if'ов

```
if условие1:
```

```
    ...
```

```
else:
```

```
    if условие2:
```

```
        ...
```

```
    else:
```

```
        if условие3:
```

```
            ...
```

```
print('Любите ли вы котиков?')
answer1 = input()
print('Умеете ли вы программировать?')
answer2 = input()
if answer1 == 'да' and answer2 == 'да':
    print('Да вы просто идеал!')
elif answer1 == 'да' and answer2 == 'нет':
    print('Вы обладаете редкостной добротой.')
elif answer1 == 'нет' and answer2 == 'да':
    print('Вы обладаете незаурядным умом.')
elif answer1 == 'нет' and answer2 == 'нет':
    print('У вас большие перспективы.')
else:
    print('Ошибка: ожидалось ответы да/нет')
```



```
print('Любите ли вы котиков?')
answer1 = input()
print('Умеете ли вы программировать?')
answer2 = input()
if ((answer1 == 'да' or answer1 == 'нет') and
    (answer2 == 'да' or answer2 == 'нет')):
    if answer1 == 'да':
        if answer2 == 'да':
            print('Да вы просто идеал!')
        else:
            print('Вы обладаете редкостной добротой.')
    else:
        if answer2 == 'да':
            print('Вы обладаете незаурядным умом.')
        else:
            print('У вас большие перспективы.')
else:
    print('Ошибка: ожидалось ответы да/нет')
```

Операции над строками

```
# + конкатенация (склеивание)
```

```
x = '30'
```

```
y = '40'
```

```
print(x + y)
```

```
# * дублирование
```

```
x = '30'
```

```
y = '40'
```

```
print(x * 2 + y * 3)
```

```
# + конкатенация (склеивание)
```

```
x = '30'
```

```
y = '40'
```

```
print(x + y)
```

```
# * дублирование
```

```
x = '30'
```

```
y = '40'
```

```
print(x * 2 + y * 3)
```

```
3040
```

```
3030404040
```

in

'хорош'	in 'хорошо'	– верно
'хорош'	in 'эх, хорошо'	– верно
'хорош'	in 'плохо'	– неверно
'хорошо'	in 'хорош'	– неверно

```
text = input()
if 'хорош' in text and 'плох' not in text:
    print('Положительная эмоциональная окраска')
elif 'плох' in text and 'хорош' not in text:
    print('Отрицательная эмоциональная окраска')
else:
    print('Нейтральная или смешанная эмоциональная окраска')
```



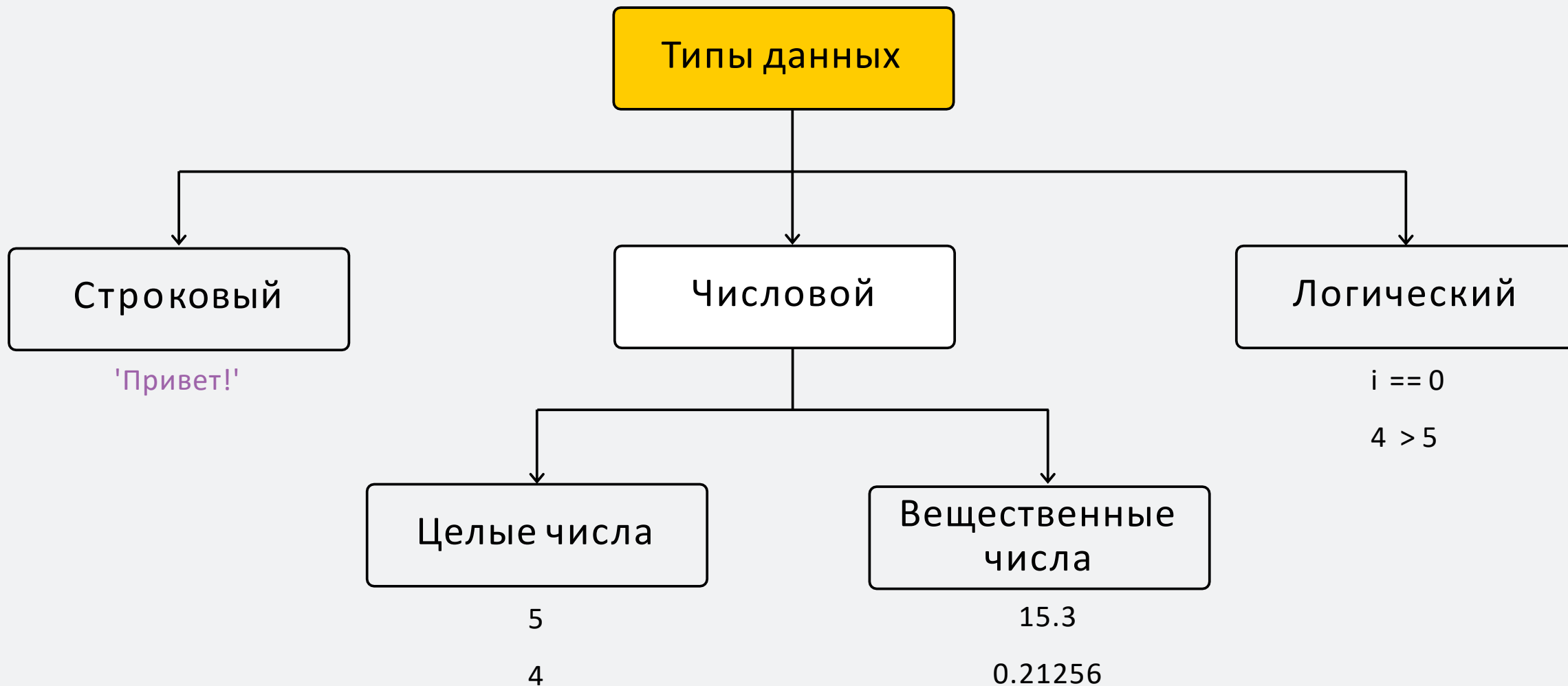
Типы данных


```
# Создаём атрибуты
# Переменные a и b содержат ссылки на два разных объекта –
# экземпляра класса Fruit, которые можно наделить разными
# атрибутами:
```

```
a.name = 'apple'
a.weight = 120
# теперь a - это яблоко весом 120 грамм
```

```
b.name = 'orange'
b.weight = 150
# b - это апельсин весом 150 грамм
```

Типы данных



Целочисленное деление

Для реализации целочисленного деления существуют два действия – деление нацело и остаток от деления нацело. Получение целой части от деления обозначается как удвоенный знак деления **//**, а остатка от деления нацело – **%**

Пример:

$$190 \text{ // } 27 = > 7$$

$$190 \% 27 = > 1$$

Приоритет операций

1. Возведение в степень (******).
2. Унарный минус (**-**). *Используется для получения, например, противоположного числа.*
3. Умножение, деление (***** **/** **%** **//**).
4. Сложение и вычитание (**+** **-**).
5. Операторы сравнения (**<=** **<** **>** **>=**)
6. Операторы равенства (**==** **!=**)
7. Операторы присваивания (**=**)
8. Логические операторы (**not** **or** **and**)

Простейшие функции

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





Цикл while


```
number = float(input())
while number > 0:
    print('Вы ввели положительное число! Вводите дальше.')
    number = float(input())
    print('Так-так, что тут у нас...')
print('Вы ввели отрицательное число или ноль. Всё.')
```

```
# Составной оператор присваивания
number = int(input()) # например, 5
number = number + 1    # number становится равным 6
print(number)

number += 1
```



Составной оператор присваивания

$x = x + y$

$x += y$

$x = x - y$

$x -= y$

$x = x * y$

$x *= y$

$x = x / y$

$x /= y$

$x = x // y$

$x //= y$

$x = x \% y$

$x \% = y$

$x = x ** y$

$x ** = y$

Цикл for

```
n = int(input())  
for i in range(n):  
    print(i)
```



Именованные аргументы
функции print

```
print('При')  
print('вет!')
```

```
print('При', end='')  
print('вет!')
```

```
print('Раз', 'два', 'три')  
print('Раз', 'два', 'три', sep='--')
```

```
# sep - разделитель  
# end - окончание
```

```
При  
вет!  
Привет!  
Раз два три  
Раз--два--три
```



Two bool

Тип bool

True and True	==	True
True and False	==	False
False and False	==	False
True or True	==	True
True or False	==	True
False or False	==	False
not True	==	False
not False	==	True

```
str(True) == 'True'  
int(True) == 1 and int(False) == 0  
bool('') == False  
bool('любой непустой строки') == True  
bool(0) == False  
bool(любого ненулевого числа) == True
```



```
# в условиях приведение к типу bool автоматическое
n = 10
while n:
    n -= 1
    print(n)
print('Пуск!')
```

break и continue

```
for i in range(10):  
    print('Итерация номер', i, 'начинается...')  
    if i == 3:  
        print('Ха! Внезапный выход из цикла!')  
        break  
    print('Итерация номер', i, 'успешно завершена.')  
print('Цикл завершён.')
```

```
# Бесконечный цикл:  
while True:  
    word = input()  
    if word == 'стоп':  
        break  
    print('Вы ввели:', word)  
print('Конец.')
```



```
for i in range(10):  
    print('Итерация номер', i, 'начинается...')  
    if i == 3:  
        print('...но не завершается успешно.')  
        continue  
    print('Итерация номер', i, 'успешно завершена.')  
print('Цикл завершён.')
```



```
n = int(input())
for i in range(1, n+1):
    for j in range(1, n+1):
        print(i*j, end='\t')
        if j == 7:
            break
    print()
```



Множество

```
mammals = {'cat', 'dog', 'fox', 'elephant'}  
print(mammals)
```

```
{'fox', 'cat', 'dog', 'elephant'}
```

```
mammals = {'cat', 'dog', 'fox', 'elephant'}  
print(mammals)
```

```
{'fox', 'cat', 'dog', 'elephant'}  
{'dog', 'elephant', 'cat', 'fox'}
```

```
mammals = {'cat', 'dog', 'fox', 'elephant'}  
print(mammals)
```

```
{'fox', 'cat', 'dog', 'elephant'}  
{'dog', 'elephant', 'cat', 'fox'}  
{'elephant', 'fox', 'dog', 'cat'}
```

```
empty = set()  
print(empty)
```



```
m_nums = {'cat', 5, 'dog', 3, 'fox', 12, 'elephant', 4}  
print(m_nums)
```




```
birds = {'raven', 'sparrow', 'sparrow', 'dove', 'hawk'}  
print(birds)
```

```
{'raven', 'hawk', 'dove', 'sparrow'}
```

Операции над множеством

```
my_set = {'a', 'b', 'c', 1, 2, 3}  
n = len(my_set)  
print(n)
```

6

```
computer = {'Системный блок', 'Монитор', 'Клавиатура', 'Мышь'}  
for element in computer:  
    print(element, end=', ')
```

Мышь, Монитор, Системный блок, Клавиатура,

```
computer = {'Системный блок', 'Монитор', 'Клавиатура', 'Мышь'}  
print('Мышь' in computer)  
print('Блок' in computer)
```

```
True  
False
```

```
numbers = set()
for i in range(7):
    numbers.add(i)
print(numbers)
```

```
{0, 1, 2, 3, 4, 5, 6}
```

```
numbers = {7, 2, 1, 4, 3, 6, 5}
numbers.remove(1)
numbers.discard(8)
for i in range(len(numbers)):
    print(numbers.pop(), end=', ')
```

```
2, 3, 4, 5, 6, 7,
```

Операции над двумя множествами


```
firms = {'Apple', 'Acer', 'Blackberry', 'Samsung'}  
fruits = {'Apple', 'Mandarin', 'Pear', 'Blackberry'}  
print(len(firms & fruits))  
print(len(firms | fruits))
```

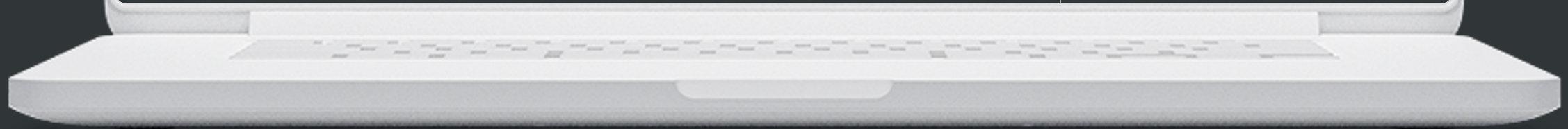
2

6

```
fib_numbers = {1, 2, 3, 5, 8, 13}  
odd_numbers = {1, 3, 5, 7, 9, 11, 13}  
print(len(fib_numbers - odd_numbers))  
print(len(odd_numbers ^ fib_numbers))
```

```
2  
5
```

`{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}`



```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}
```

```
True
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}
```

True

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}
```

```
True  
False
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{'one', 'three'} <= {'one', 'two', 'three'}
```

```
True  
False
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{ 'one', 'three' } <= { 'one', 'two', 'three' }
```

```
True  
False  
True
```



```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{ 'one', 'three' } <= { 'one', 'two', 'three' }  
{1, 3, 5, 7, 9, 11} > {3, 5, 7, 11, 13}
```

```
True  
False  
True
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{'one', 'three'} <= {'one', 'two', 'three'}  
{1, 3, 5, 7, 9, 11} > {3, 5, 7, 11, 13}
```

```
True  
False  
True  
False
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{'one', 'three'} <= {'one', 'two', 'three'}  
{1, 3, 5, 7, 9, 11} > {3, 5, 7, 11, 13}  
{'let', 'it', 'be'} < {'be', 'it', 'let'}
```

```
True  
False  
True  
False
```

```
{1, 2, 3, 4, 5, 6} == {6, 5, 3, 4, 1, 2}  
{2, 4, 6, 8, 10} != {10, 2, 6, 4, 8}  
{'one', 'three'} <= {'one', 'two', 'three'}  
{1, 3, 5, 7, 9, 11} > {3, 5, 7, 11, 13}  
{'let', 'it', 'be'} < {'be', 'it', 'let'}
```

```
True  
False  
True  
False  
False
```

Нумерация
Символов в строке

Нумерация символов в строке

```
word = 'Я_УЧУСЬ_В_РТУ'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У

```
print(word[0])  
print(word[5])  
print(word[13])
```

Нумерация символов в строке

```
word = 'Я_УЧУСЬ_В_РТУ'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У

```
print(word[0])  
print(word[5])  
print(word[13])
```

Я

С

IndexError: string index out of range

Нумерация символов в строке

```
word = 'Я_УЧУСЬ_В_РТУ'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(word[0])
```

Я

```
print(word[5])
```

С

```
print(word[13])
```

IndexError: string index out of range

Нумерация символов в строке

word = 'Я_УЧУСЬ_В_РТУ'

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(word[0])
```

Я

```
print(word[5])
```

С

```
print(word[13])
```

IndexError: string index out of range

```
print(word[-1])
```

```
print(word[-8])
```

Нумерация символов в строке

word = 'Я_УЧУСЬ_В_РТУ'

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(word[0])
```

Я

```
print(word[5])
```

С

```
print(word[13])
```

IndexError: string index out of range

```
print(word[-1])
```

У

```
print(word[-8])
```

С

Нумерация символов в строке

word = 'Я_УЧУСЬ_В_РТУ'

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(word[0])      Я
print(word[5])      С
print(word[13])     IndexError: string index out of range
print(word[-1])     У
print(word[-8])     С
word[3] = 'Ы'
```

Нумерация символов в строке

```
word = 'Я_УЧУСЬ_В_РТУ'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
Я	_	У	Ч	У	С	Ь	_	В	_	Р	Т	У
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(word[0])
```

Я

```
print(word[5])
```

С

```
print(word[13])
```

IndexError: string index out of range

```
print(word[-1])
```

У

```
print(word[-8])
```

С

```
word[3] = 'Ы'
```

TypeError: 'str' object does not support item assignment

Хранение текстов в памяти компьютера

```
print(ord('5'))
```

```
1041
```

```
print(ord('Б'))  
print(ord('А'))  
print(ord('Г'))
```

1041

```
print(ord('Б'))  
print(ord('А'))  
print(ord('Г'))
```

```
1041  
1040  
1043
```



```
print(ord('Б'))  
print(ord('А'))  
print(ord('Г'))  
print(chr(1044))  
print(chr(1046))
```

```
1041  
1040  
1043
```

```
print(ord('Б'))  
print(ord('А'))  
print(ord('Г'))  
print(chr(1044))  
print(chr(1046))
```

1041

1040

1043

Д

Ж

Что выведет программа?

```
for i in range(26):  
    print(chr(ord('A') + i), end=',')
```



```
# Что выведет программа?  
for i in range(26):  
    print(chr(ord('A') + i), end=',')
```

A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,



Срезы строк

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])
```


Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])
```

Hello

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])  
print(text[7:12])
```

Hello

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])  
print(text[7:12])
```

```
Hello  
world
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])  
print(text[7:12])  
print(text[:5])
```

```
Hello  
world
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])    world
print(text[:5])       Hello
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])    world
print(text[:5])       Hello
print(text[7:])       world
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])     world
print(text[:5])       Hello
print(text[7:])        world!
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])     world
print(text[:5])        Hello
print(text[7:])         world!
print(text[:])          Hello, world!
```


Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])    world
print(text[:5])       Hello
print(text[7:])       world!
print(text[:])        Hello, world!
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])     world
print(text[:5])        Hello
print(text[7:])         world!
print(text[:])          Hello, world!
print(text[11:15])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!

```
print(text[0:5])      Hello
print(text[7:12])     world
print(text[:5])       Hello
print(text[7:])        world!
print(text[:])         Hello, world!
print(text[11:15])    d!
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[::-9])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])
```

Hello

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])
```

Hello

```
print(text[-5:-3])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])
```

Hell

```
print(text[-5:-3])
```

or

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])      Hell
print(text[-5:-3])    or
print(text[-6:])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])
```

Hello

```
print(text[-5:-3])
```

or

```
print(text[-6:])
```

world!

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])      Hello
print(text[-5:-3])    or
print(text[-6:])       world!
                        H
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])      Hello
print(text[-5:-3])    or
print(text[-6:])       world!
print(text[:1])        H
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])      Hello
print(text[-5:-3])    or
print(text[-6:])       world!
print(text[:1])        H
                        !
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[:-9])      Hello
print(text[-5:-3])    or
print(text[-6:])      world!
print(text[:1])       H
print(text[-1:])      !
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])      Hl o ol
```


Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])
```

Hlo ol

```
print(text[1:12:2])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])    Hlo ol
print(text[1:12:2])    el, wrd
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])      Hlo ol
print(text[1:12:2])      el, wrd
print(text[::4])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])      Hlo ol
print(text[1:12:2])      el, wrd
print(text[::4])          Hoo!
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])      Hlo ol
print(text[1:12:2])      el, wrd
print(text[::4])          Hoo!
print(text[::-1])
```

Срезы строк

```
text = 'Hello, world!'
```

0	1	2	3	4	5	6	7	8	9	10	11	12
H	e	l	l	o	,		w	o	r	l	d	!
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
print(text[0:12:2])
```

Hlo ol

```
print(text[1:12:2])
```

el, wrd

```
print(text[::4])
```

Hoo!

```
print(text[::-1])
```

!dlrow ,olleH



Списки (тип list)


```
empty1 = []  
empty2 = list()  
print('Пустые списки: ', empty1, empty2)
```

```
Пустые списки:  [] []
```

```
empty1 = []  
empty2 = list()  
print('Пустые списки: ', empty1, empty2)  
my_list = [0] * 5  
print('Список из нулей: ', my_list)
```

```
Пустые списки:  [] []
```

```
empty1 = []  
empty2 = list()  
print('Пустые списки: ', empty1, empty2)  
my_list = [0] * 5  
print('Список из нулей: ', my_list)
```

```
Пустые списки:  [] []  
Список из нулей:  [0, 0, 0, 0, 0]
```

```
#           0  1  2  3  4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
```



```
#           0  1  2  3  4  
primes = [2, 3, 5, 7, 11]  
print('Весь список:', primes)
```

```
Весь список: [2, 3, 5, 7, 11]
```

```
#           0  1  2  3  4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
```

```
Весь список: [2, 3, 5, 7, 11]
```

```
#           0   1   2   3   4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
```

```
Весь список: [2, 3, 5, 7, 11]
Сумма первых двух простых чисел: 5
```

```
#           0   1   2   3   4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
print('Последнее из простых чисел в нашем списке:', primes[-1])
```

```
Весь список: [2, 3, 5, 7, 11]
Сумма первых двух простых чисел: 5
```



```
#           0   1   2   3   4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
print('Последнее из простых чисел в нашем списке:', primes[-1])
```

```
Весь список: [2, 3, 5, 7, 11]
Сумма первых двух простых чисел: 5
Последнее из простых чисел в нашем списке: 11
```

```
#           0   1   2   3   4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
print('Последнее из простых чисел в нашем списке:', primes[-1])
primes.append(13)
print('Список с еще одним простым числом:', primes)
```

```
Весь список: [2, 3, 5, 7, 11]
Сумма первых двух простых чисел: 5
Последнее из простых чисел в нашем списке: 11
```

```
#           0   1   2   3   4
primes = [2, 3, 5, 7, 11]
print('Весь список:', primes)
print('Сумма первых двух простых чисел:', primes[0] + primes[1])
print('Последнее из простых чисел в нашем списке:', primes[-1])
primes.append(13)
print('Список с еще одним простым числом:', primes)
```

```
Весь список: [2, 3, 5, 7, 11]
Сумма первых двух простых чисел: 5
Последнее из простых чисел в нашем списке: 11
Список с еще одним простым числом: [2, 3, 5, 7, 11, 13]
```

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
```



```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
```

```
Дополненный список: [2, 3, 5, 7, 11, 13, 17]
```

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

Расширенный список: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
print('Количество элементов в списке:', len(primes))
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

Расширенный список: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]


```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
print('Количество элементов в списке:', len(primes))
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

Расширенный список: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

Количество элементов в списке: 10

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
print('Количество элементов в списке:', len(primes))
print(1 in primes)
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

Расширенный список: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

Количество элементов в списке: 10

```
primes = [2, 3, 5, 7, 11, 13]
primes.append(15) # ошиблись
primes[6] = 17    # ошибка исправлена
print('Дополненный список:', primes)
primes += [19, 23, 29]
print('Расширенный список:', primes)
print('Количество элементов в списке:', len(primes))
print(1 in primes)
```

Дополненный список: [2, 3, 5, 7, 11, 13, 17]

Расширенный список: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

Количество элементов в списке: 10

False

Перебор элементов списка

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]  
for prime in primes:  
    print(prime)
```



```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]
for prime in primes:
    print(prime)
```

```
2
3
5
7
11
13
17
19
23
29
```

```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]  
for i in range(len(primes)):  
    print(i, ': ', primes[i])
```



```
primes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29]  
for i in range(len(primes)):  
    print(i, ': ', primes[i])
```

```
0 : 2  
1 : 3  
2 : 5  
3 : 7  
4 : 11  
5 : 13  
6 : 17  
7 : 19  
8 : 23  
9 : 29
```


Срезы списков

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
spring = months[2:5]  
for month in spring:  
    print(month, end=' ')
```



```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
spring = months[2:5]  
for month in spring:  
    print(month, end=' ')
```

март апрель май

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
spring = months[2:5]  
for month in spring:  
    print(month, end=' ')  
print()  
winter = months[-1:] + months[:2]  
for month in winter:  
    print(month, end=' ')
```

март апрель май

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
spring = months[2:5]  
for month in spring:  
    print(month, end=' ')  
print()  
winter = months[-1:] + months[:2]  
for month in winter:  
    print(month, end=' ')
```

```
март апрель май  
декабрь январь февраль
```

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
print(months[::2])
```



```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
print(months[::2])
```

```
['январь', 'март', 'май', 'июль', 'сентябрь', 'ноябрь']
```

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
print(months[::2])  
print(months[1::2])
```

```
['январь', 'март', 'май', 'июль', 'сентябрь', 'ноябрь']
```



```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
print(months[::2])  
print(months[1::2])
```

```
['январь', 'март', 'май', 'июль', 'сентябрь', 'ноябрь']  
['февраль', 'апрель', 'июнь', 'август', 'октябрь', 'декабрь']
```

```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
months[5:8] = ['June', 'July', 'August']  
print(months)
```



```
months = ['январь', 'февраль', 'март', 'апрель', 'май', 'июнь',  
          'июль', 'август', 'сентябрь', 'октябрь', 'ноябрь', 'декабрь']  
months[5:8] = ['June', 'July', 'August']  
print(months)
```

```
['январь', 'февраль', 'март', 'апрель', 'май', 'June', 'July', 'August',  
'сентябрь', 'октябрь', 'ноябрь', 'декабрь']
```



Кортежи (тип `tuple`)

```
empty = ()  
card = ('7', 'пик')  
t = (18,)   
print(len(card))
```



```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))
```

2

```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))  
print(card[0])
```

2


```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))  
print(card[0])
```

```
2  
7
```

```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))  
print(card[0])  
print(card + t)
```

```
2  
7
```

```
empty = ()  
card = ('7', 'пик')  
t = (18,)   
print(len(card))  
print(card[0])  
print(card + t)
```

```
2  
7  
('7', 'пик', 18)
```

```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))  
print(card[0])  
print(card + t)  
card[1] = 'треф'
```

```
2  
7  
( '7', 'пик', 18)
```

```
empty = ()  
card = ('7', 'пик')  
t = (18,)  
print(len(card))  
print(card[0])  
print(card + t)  
card[1] = 'треф'
```

```
2  
7
```

```
('7', 'пик', 18)
```

```
TypeError: 'tuple' object does not support item assignment
```

Сравнение кортежей

`(1, 2, 3) == (1, 3, 2)`



```
(1, 2, 3) == (1, 3, 2)
```

```
False
```



```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)
```

False

```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)
```

```
False  
True
```

```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)  
(1, 2) <= (5,)
```

```
False  
True
```

```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)  
(1, 2) <= (5,)
```

```
False  
True  
True
```

```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)  
(1, 2) <= (5, )  
( '7', 'треф' ) > ( '7', 'червей' )
```

```
False  
True  
True
```

```
(1, 2, 3) == (1, 3, 2)  
(1, 2, 3) < (1, 2, 4)  
(1, 2) <= (5, )  
( '7', 'треф' ) > ( '7', 'червей' )
```

```
False  
True  
True  
False
```

```
(1, 2, 3) == (1, 3, 2)
(1, 2, 3) < (1, 2, 4)
(1, 2) <= (5,)
('7', 'треф') > ('7', 'червей')
(1, 2) != ('7', 'пик')
```

```
False
True
True
False
```

```
(1, 2, 3) == (1, 3, 2)
(1, 2, 3) < (1, 2, 4)
(1, 2) <= (5,)
('7', 'треф') > ('7', 'червей')
(1, 2) != ('7', 'пик')
```

```
False
True
True
False
True
```



```
(1, 2, 3) == (1, 3, 2)
(1, 2, 3) < (1, 2, 4)
(1, 2) <= (5,)
('7', 'треф') > ('7', 'червей')
(1, 2) != ('7', 'пик')
(3, 4) < ('5', 'бубен')
```

```
False
True
True
False
True
```

```
(1, 2, 3) == (1, 3, 2)
(1, 2, 3) < (1, 2, 4)
(1, 2) <= (5,)
('7', 'треф') > ('7', 'червей')
(1, 2) != ('7', 'пик')
(3, 4) < ('5', 'бубен')
```

False

True

True

False

True

TypeError: '<' not supported
between instances of 'int'
' and 'str'

Присваивание кортежей

```
(n, s) = (10, 'hello')
```



```
(n, s) = (10, 'hello')  
# то же самое, что  
n, s = 10, 'hello'
```



```
(n, s) = (10, 'hello')
```

```
# то же самое, что
```

```
n, s = 10, 'hello'
```

```
# то же самое, что
```

```
n = 10
```

```
s = 'hello'
```

Обмен значениями переменных

```
a, b = 1, 2  
a = b  
b = a  
print(a, b, sep=', ')
```




```
a, b = 1, 2  
a = b  
b = a  
print(a, b, sep=', ')
```

```
2, 2
```

```
a, b = 1, 2  
a = b  
b = a  
print(a, b, sep=', ')
```

2, 2

```
a, b = 1, 2  
t = a  
a = b  
b = t  
print(a, b, sep=', ')
```



```
a, b = 1, 2  
t = a  
a = b  
b = t  
print(a, b, sep=', ')
```

```
2, 1
```

```
a, b = 1, 2  
a, b = b, a  
print(a, b, sep=', ')
```



```
a, b = 1, 2  
a, b = b, a  
print(a, b, sep=', ')
```

```
2, 1
```

```
a, b = 1, 2
a, b = b, a
print(a, b, sep=', ')
a, b, c = 3, 2, 1
b, a, c = c, a, b
print(b, c, a, sep=', ')
```

2, 1

```
a, b = 1, 2
a, b = b, a
print(a, b, sep=', ')
a, b, c = 3, 2, 1
b, a, c = c, a, b
print(b, c, a, sep=', ')
```

```
2, 1
1, 2, 3
```



```
# Сортировка пузырьком
```

```
n = int(input())
```

```
a = []
```

```
for i in range(n):
```

```
    a.append(int(input()))
```

```
for i in range(n-1):
```

```
    for j in range(n-1-i):
```

```
        if a[j] > a[j+1]:
```

```
            a[j], a[j+1] = a[j+1], a[j]
```

```
print(a)
```

Преобразования
между коллекциями

```
writer = ('Лев Толстой', 1827)
a = list(writer)
a[1] = 1828
writer = tuple(a)
print(writer)
```



```
writer = ('Лев Толстой', 1827)
a = list(writer)
a[1] = 1828
writer = tuple(a)
print(writer)
```

```
('Лев Толстой', 1828)
```

```
a = [1, 2, 1, 1, 2, 2, 3, 3]  
print(len(set(a)))
```



```
a = [1, 2, 1, 1, 2, 2, 3, 3]  
print(len(set(a)))
```

3

```
a = [1, 2, 1, 1, 2, 2, 3, 3]
print(len(set(a)))
names = {'Иван', 'Петр', 'Сергей', 'Алексей'}
print(sorted(list(names)))
```

3

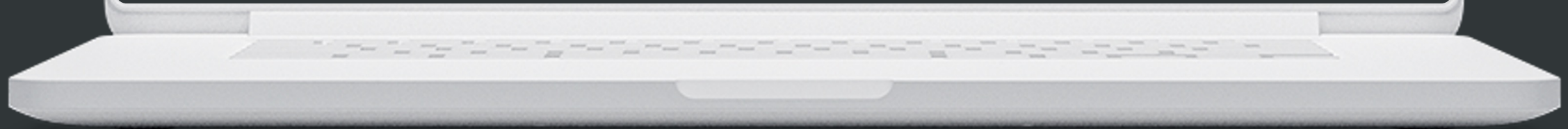
```
a = [1, 2, 1, 1, 2, 2, 3, 3]
print(len(set(a)))
names = {'Иван', 'Петр', 'Сергей', 'Алексей'}
print(sorted(list(names)))
```

```
3
['Алексей', 'Иван', 'Петр', 'Сергей']
```




Методы `split` и `join`

```
s = 'раз два три'  
print(s.split())
```



```
s = 'раз два три'  
print(s.split())
```

```
['раз', 'два', 'три']
```

```
s = 'раз два три'  
print(s.split())  
print(' one two three '.split())
```

```
['раз', 'два', 'три']
```

```
s = 'раз два три'  
print(s.split())  
print(' one two three '.split())
```

```
['раз', 'два', 'три']  
['one', 'two', 'three']
```

```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
```

```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
['192', '168', '1', '1']
```



```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
print(s.split('a'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
['192', '168', '1', '1']
```

```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
print(s.split('a'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
['192', '168', '1', '1']
['р', 'з дв', ' три']
```

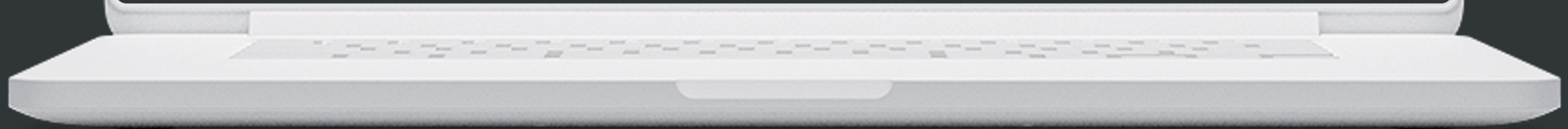
```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
print(s.split('a'))
print('A##B##C'.split('##'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
['192', '168', '1', '1']
['р', 'з дв', ' три']
```

```
s = 'раз два три'
print(s.split())
print(' one two three '.split())
print('192.168.1.1'.split('.'))
print(s.split('a'))
print('A##B##C'.split('##'))
```

```
['раз', 'два', 'три']
['one', 'two', 'three']
['192', '168', '1', '1']
['р', 'з дв', ' три']
['A', 'B', 'C']
```

```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))
```



```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))
```

ТотКогоНельзяНазывать

```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))
```

ТотКогоНельзяНазывать

```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))
```

```
ТотКогоНельзяНазывать  
Тот Кого Нельзя Называть
```



```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))  
print('-'.join(s))
```

```
ТотКогоНельзяНазывать  
Тот Кого Нельзя Называть
```

```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))  
print('-'.join(s))
```

```
ТотКогоНельзяНазывать  
Тот Кого Нельзя Называть  
Тот-Кого-Нельзя-Называть
```

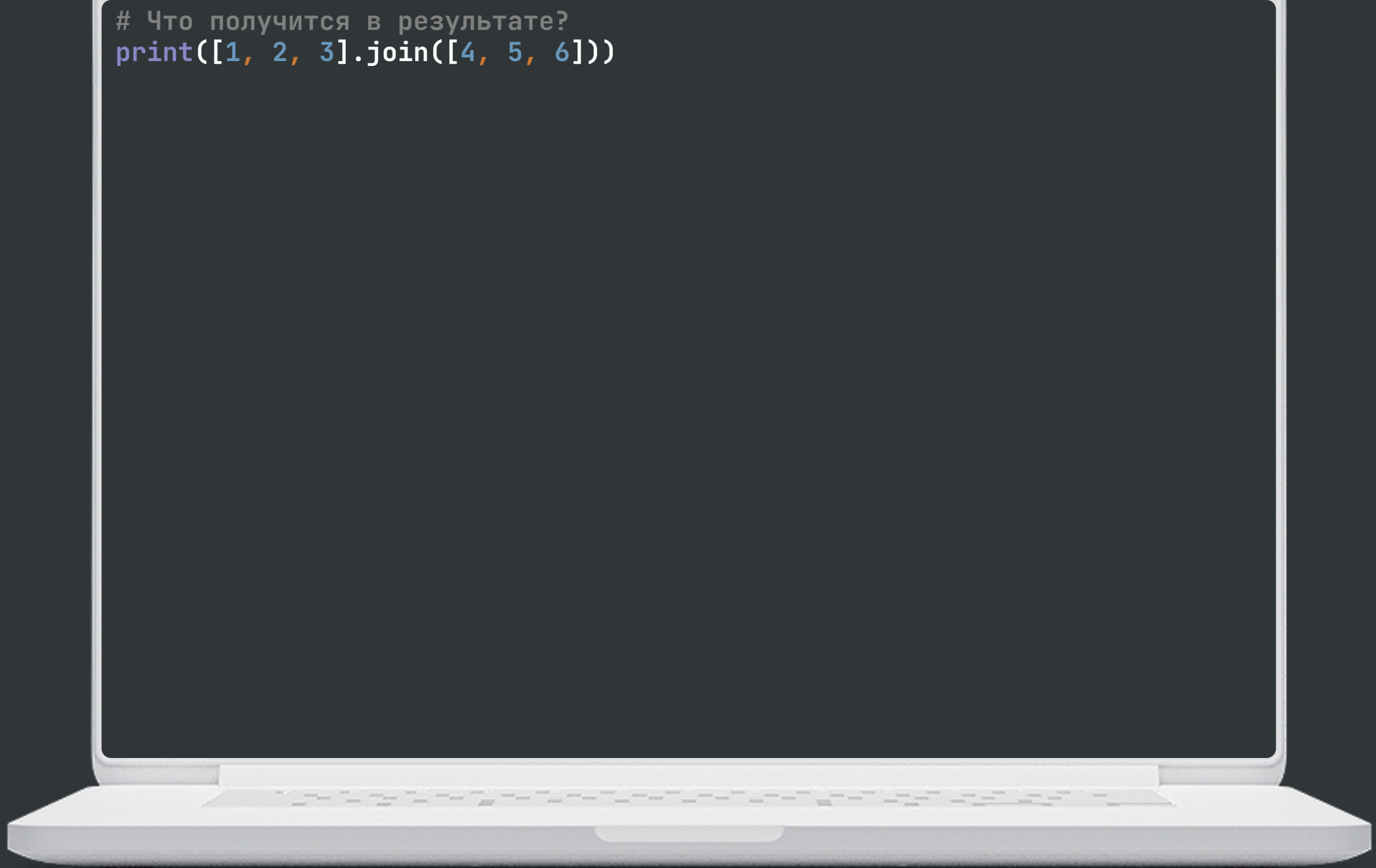
```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))  
print('-'.join(s))  
print('! '.join(s))
```

```
ТотКогоНельзяНазывать  
Тот Кого Нельзя Называть  
Тот-Кого-Нельзя-Называть
```

```
s = ['Тот', 'Кого', 'Нельзя', 'Называть']  
print(''.join(s))  
print(' '.join(s))  
print('-'.join(s))  
print('! '.join(s))
```

```
ТотКогоНельзяНазывать  
Тот Кого Нельзя Называть  
Тот-Кого-Нельзя-Называть  
Тот! Кого! Нельзя! Называть
```

```
# Что получится в результате?  
print([1, 2, 3].join([4, 5, 6]))
```



```
# Что получится в результате?  
print([1, 2, 3].join([4, 5, 6]))
```

```
AttributeError: 'list' object has no attribute 'join'
```

```
# Что получится в результате?  
print([1, 2, 3].join([4, 5, 6]))  
print({1, 2, 3}.join({4, 5, 6}))
```

```
AttributeError: 'list' object has no attribute 'join'
```

```
# Что получится в результате?  
print([1, 2, 3].join([4, 5, 6]))  
print({1, 2, 3}.join({4, 5, 6}))
```

```
AttributeError: 'list' object has no attribute 'join'  
AttributeError: 'set' object has no attribute 'join'
```


Списочные выражения

```
squares = []  
for i in range(8):  
    squares.append(i ** 2)  
print(squares)
```



```
squares = []  
for i in range(8):  
    squares.append(i ** 2)  
print(squares)
```

```
[0, 1, 4, 9, 16, 25, 36, 49]
```

```
squares = []  
for i in range(8):  
    squares.append(i ** 2)  
print(squares)  
  
# с помощью списочного выражения:  
squares = [i ** 2 for i in range(8)]  
print(squares)
```

```
[0, 1, 4, 9, 16, 25, 36, 49]
```

```
squares = []  
for i in range(8):  
    squares.append(i ** 2)  
print(squares)  
  
# с помощью списочного выражения:  
squares = [i ** 2 for i in range(8)]  
print(squares)
```

```
[0, 1, 4, 9, 16, 25, 36, 49]
```

```
[0, 1, 4, 9, 16, 25, 36, 49]
```

```
even_squares = []  
for i in range(10):  
    if i % 2 == 0:  
        even_squares.append(i ** 2)  
print(even_squares)
```



```
even_squares = []  
for i in range(10):  
    if i % 2 == 0:  
        even_squares.append(i ** 2)  
print(even_squares)
```

```
[0, 4, 16, 36, 64]
```

```
even_squares = []
for i in range(10):
    if i % 2 == 0:
        even_squares.append(i ** 2)
print(even_squares)

# с помощью списочного выражения:
even_squares2 = [i ** 2 for i in range(10) if i % 2 == 0]
print(even_squares2)
```

```
[0, 4, 16, 36, 64]
```



```
even_squares = []
for i in range(10):
    if i % 2 == 0:
        even_squares.append(i ** 2)
print(even_squares)

# с помощью списочного выражения:
even_squares2 = [i ** 2 for i in range(10) if i % 2 == 0]
print(even_squares2)
```

```
[0, 4, 16, 36, 64]
```

```
[0, 4, 16, 36, 64]
```

```
print([i * j for i in range(3) for j in range(3)])
```



```
print([i * j for i in range(3) for j in range(3)])
```

```
[0, 0, 0, 0, 1, 2, 0, 2, 4]
```

```
a = [int(x) for x in '976 929 289 809 7677'.split()]  
print(a[3])
```



```
a = [int(x) for x in '976 929 289 809 7677'.split()]  
print(a[3])
```

809

```
a = [int(x) for x in '976 929 289 809 7677'.split()]  
print(a[3])  
evil, good = [int(x) for x in '666 777'.split()]  
print(evil, good, sep='\n')
```

809

```
a = [int(x) for x in '976 929 289 809 7677'.split()]  
print(a[3])  
evil, good = [int(x) for x in '666 777'.split()]  
print(evil, good, sep='\n')
```

809

666

777

```
print(', '.join(f'2^{i}={i ** 2}' for i in range(1, 10)))
```

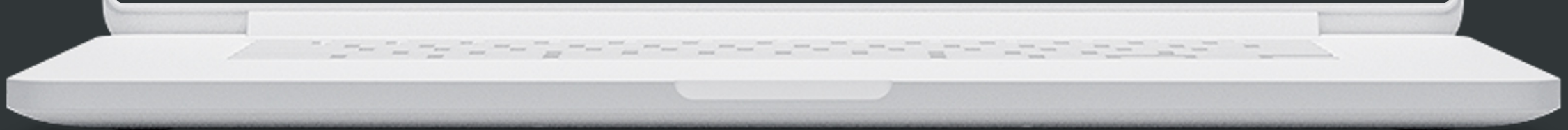



```
print(', '.join(f'2^{i}={i ** 2}' for i in range(1, 10)))
```

2^1=1, 2^2=4, 2^3=9, 2^4=16, 2^5=25, 2^6=36, 2^7=49, 2^8=64, 2^9=81



```
actors = ['Джонни Депп', 'Эмма Уотсон', 'Билли Пайпер']  
print(actors[1])
```



```
actors = [  
    ('Джонни Депп', 'Джон Кристофер Депп Второй родился 9 июня...'),  
    ('Сильвестр Сталлоне', 'Сильвестр Гарденцио Сталлоне родился...'),  
    ('Эмма Уотсон', 'Эмма Шарлотта Дуерр Уотсон родилась в семье...'),  
    # ...  
]
```

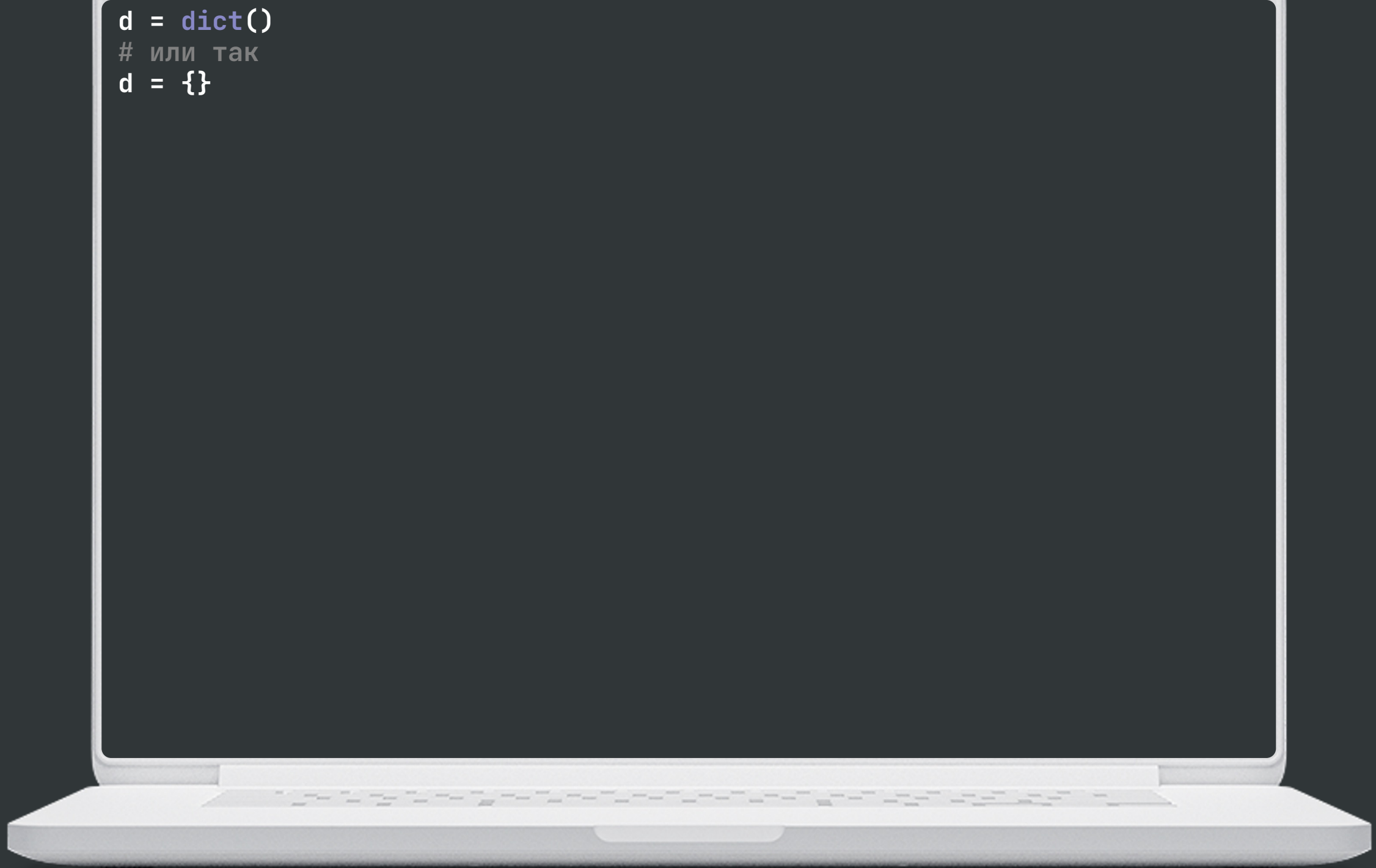
```
actors = {  
    'Джонни Депп': 'Джон Кристофер Депп Второй родился 9 июня...',  
    'Сильвестр Сталлоне': 'Сильвестр Гарденцио Сталлоне родился...',  
    'Эмма Уотсон': 'Эмма Шарлотта Дуерр Уотсон родилась в семье...',  
    # ...  
}
```



```
d = dict()
```

```
# или так
```

```
d = {}
```



```
print(actors['Эмма Уотсон'])
```



```
print(actors['Эмма Уотсон'])
```

Эмма Шарлотта Дуерр Уотсон родилась в семье...


```
print(actors['Эмма Уотсон'])  
print(actors['Несуществующий ключ'])
```

Эмма Шарлотта Дуерр Уотсон родилась в семье...

KeyError: 'Несуществующий ключ'

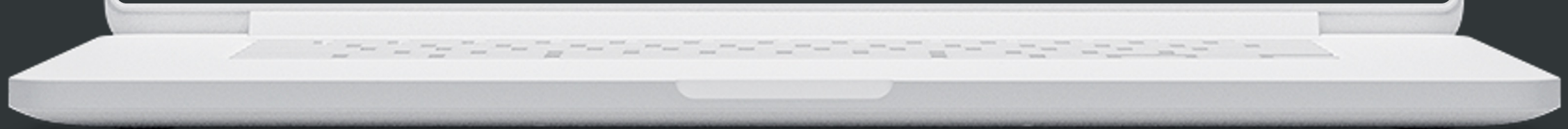
```
actors['Брэд Питт'] = 'Уильям Брэдли Питт, более известный как...'
```



```
del actors['Джонни Депп']
```



```
del actors['Джонни Депп']  
print(actors['Джонни Депп'])
```



```
del actors['Джонни Депп']  
print(actors['Джонни Депп'])
```

```
KeyError: 'Джонни Депп'
```

```
deleted_value = actors.pop('Джонни Депп')
```



```
deleted_value = actors.pop('Джонни Депп', None)
```



```
if 'Джонни Депп' in actors:  
    print('У нас есть статья про Джонни Деппа')
```




```
article = actors.get('Джонни Депп')
```



```
article = actors.get('Джонни Депп', 'Статья о Джонни Деппе не найдена')
```



```
actors_names = list(actors.keys())
```



```
actors_names = list(actors.keys())  
all_articles = list(actors.values())
```



```
actors_names = list(actors.keys())  
all_articles = list(actors.values())  
for key, val in actors.items():  
    pass
```



TO BE CONTINUED...



100% - IT IS DONE!

