



Введение в программирование на Python

Лавприт Сингх-Пальчевская
младший научный сотрудник МГУ им. Ломоносова,
кафедра биоинженерии

Проверка связи



Отправьте «**+**», если меня видно и слышно

Если у вас нет звука или изображения:

- перезагрузите страницу
- попробуйте зайти заново
- откройте трансляцию в другом браузере



Лавприт Сингх-Пальчевская

- Младший научный сотрудник МГУ им. Ломоносова, кафедра биоинженерии
- Область экспертизы: Python, анализ данных, биоинформатика

Правила работы



Расписание семинаров:

- понедельник, 18:35—20:00

Во время семинара:

- подготовьте блокнот и ручку для ведения конспекта
- выключите микрофон
- активно задавайте вопросы: напишите вопрос в чат, чтобы задать вопрос голосом, дождитесь блока «Ответы на вопросы»
- если вы уже что-то знаете, семинар — это возможность расширить и систематизировать свои знания

Вы можете оставить отзыв на семинар в форме по ссылке:

<https://forms.yandex.ru/cloud/65f457cb2530c2ad9041476d/>

О чем поговорим сегодня

1. Рассмотрим сферы применения Python, почему язык так популярен
2. Узнаем, откуда и как устанавливать Python
3. Познакомимся со средами разработки для Python
4. Посмотрим ресурсы, где искать информацию по курсу
5. Рассмотрим примеры решения некоторых задач с помощью Python
6. Рассмотрим выражения, которые могут оказаться полезными при написании кода

Сферы применения Python

Напишите, пожалуйста, в чат

Был ли у вас опыт
в программировании?



Базовые знания программирования необходимы любому специалисту, это позволяет работать на стыке областей

- системное программирование
- веб-программирование
- автоматизация тестирования
- desktop-приложения
- приложения баз данных
- искусственный интеллект
- машинное обучение
- анализ данных
- научные вычисления
- компьютерные игры и робототехника

Причины популярности Python

- Простота языка, хорошо подходит для новичков
- Модульность языка и возможность интегрироваться с другими языками программирования, в том числе в составе сложных комплексных приложений и систем
- Высококачественные готовые бесплатные библиотеки для решения самых разных задач, включая анализ больших данных и научные расчеты
- Большое сообщество поддержки

В двух урнах находится соответственно 4 и 5 белых и 6 и 3 чёрных шаров. Наудачу выбирается одна урна и из неё наугад извлекается шар. Какова вероятность того, что этот шар чёрный?

```
b1 = {'w': 4, 'b': 6}
b2 = {'w': 5, 'b': 3}
black_probability = 0.5 * (b1['b'] / (b1['w'] + b1['b'])) +
b2['b'] / (b2['w'] + b2['b'])
print(black_probability)
```

Подсчитайте евклидово расстояние между точками 1 и 2 с координатами (x_1, y_1) и (x_2, y_2) , где $x_1=1$, $x_2=5$, $y_1=3$, $y_2=10$

```
d1 = (1, 3)
d2 = (5, 10)
((d1[0]-d2[0])**2+(d1[1]-d2[1])**2)**0.5

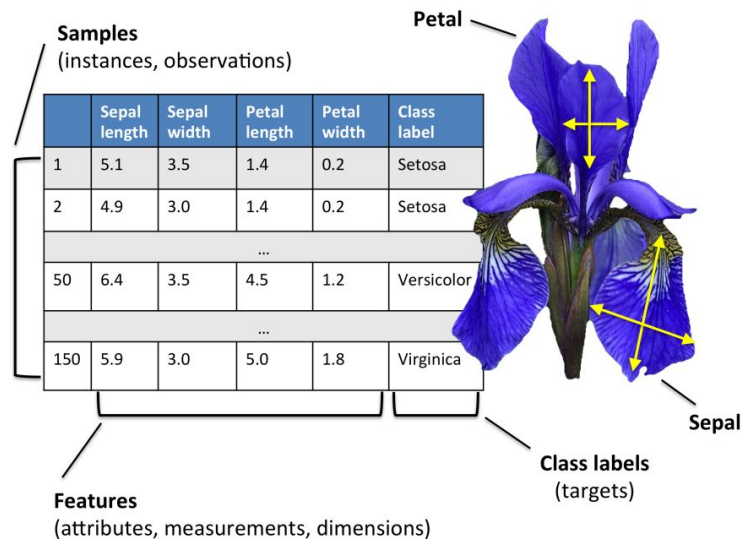
import math
math.sqrt(sum((px-qx)**2 for px, qx in
zip(d1, d2)))
```

Создание и обработка датасета

Набор данных состоит из:

- 150 образцов
- 3 класса: виды ириса (Iris setosa, Iris virginica и Iris versicolor)
- 4 признака: длина чашелистика, ширина чашелистика, длина лепестка, ширина лепестка (в см)

```
import pandas as pd
iris_data = pd.read_csv('iris.data',
header=None, names=['sepal length', 'sepal
width', 'petal length', 'petal width', 'class'])
```

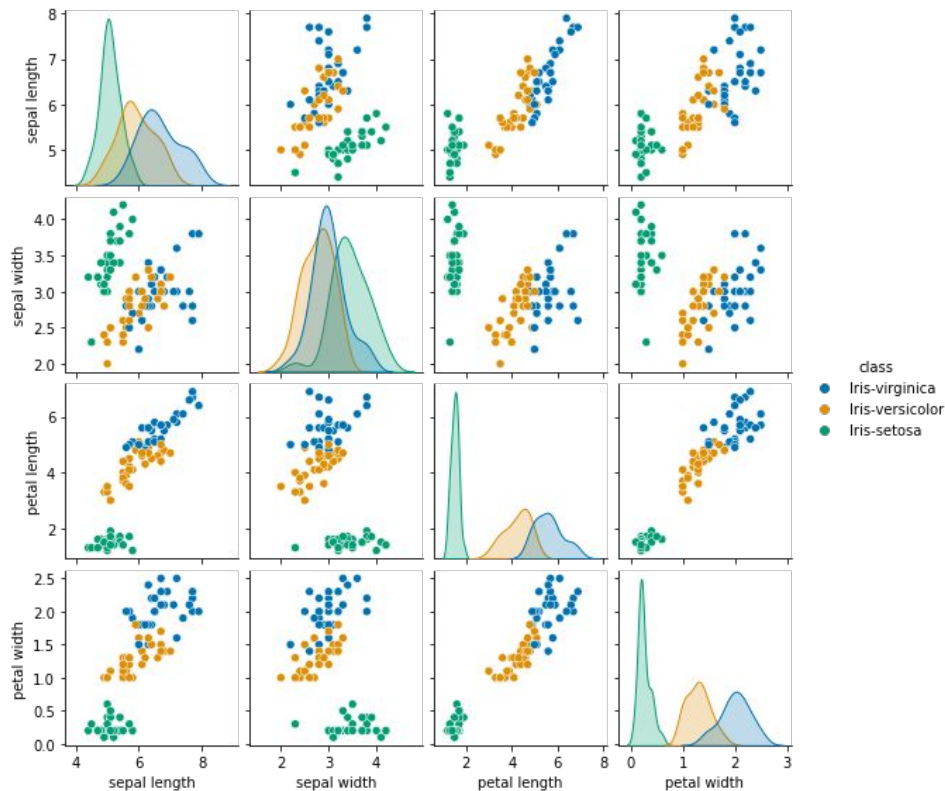


Эксплораторный анализ и построение графиков

```
import seaborn as sns
sns.pairplot(train, hue="class", height = 2,
palette = 'colorblind')
```

Что мы можем увидеть?

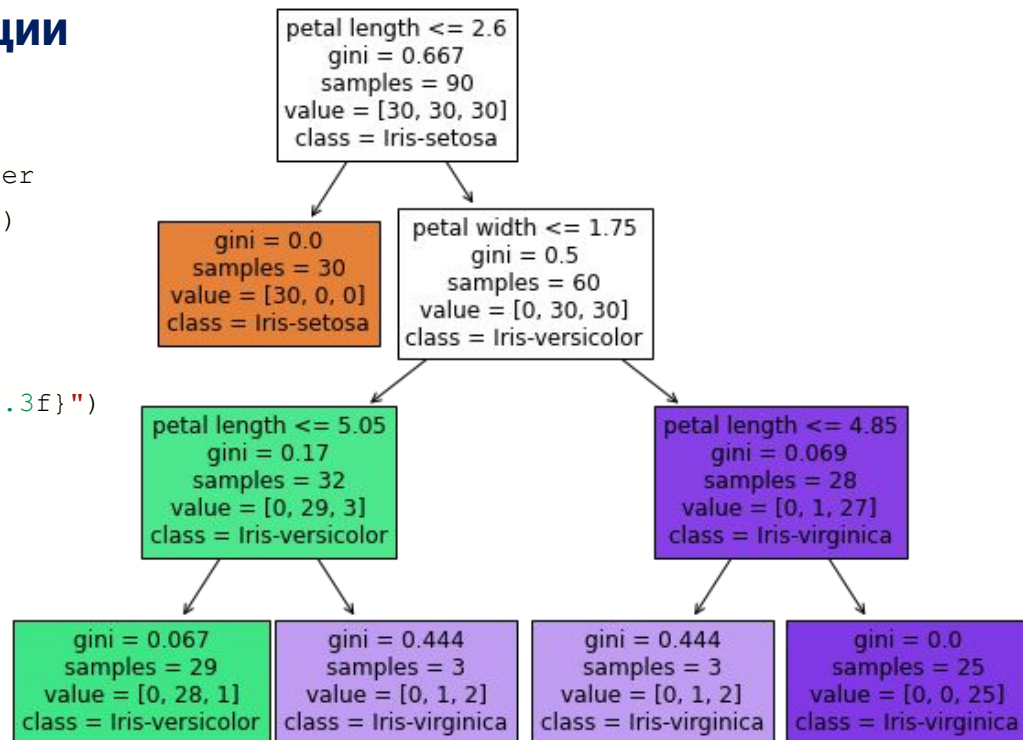
- в данных имеются строгие закономерности
- можно выделить некоторые правила классификации
- правила классификации легко интерпретировать



Построение модели классификации

```
from sklearn.tree import DecisionTreeClassifier
mod_dt = DecisionTreeClassifier(max_depth = 3)
mod_dt.fit(X_train,y_train)
prediction=mod_dt.predict(X_test)
print('The accuracy of the Decision Tree is',
f"{metrics.accuracy_score(prediction,y_test):.3f}")
```

The accuracy of the Decision Tree is 0.983



Известная нейросеть **AlphaFold**, которая предсказывает пространственную структуру белка. Программа разработана компанией Google DeepMind на базе искусственного интеллекта в 2020 году. Над этой задачей ученые-биологи трудились более 50 лет.

Every protein is made up of a sequence of amino acids bonded together

These amino acids interact locally to form shapes like helices and sheets

These shapes fold up on larger scales to form the full three-dimensional protein structure

Proteins can interact with other proteins, performing functions such as signalling and transcribing DNA

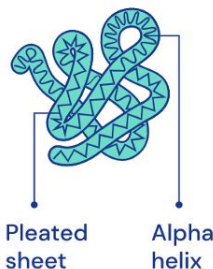
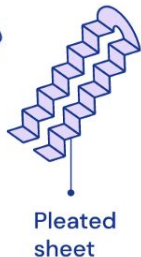
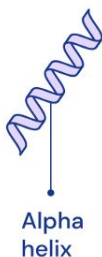
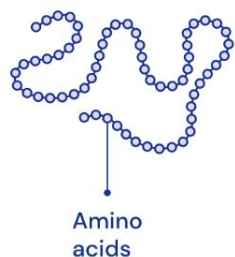
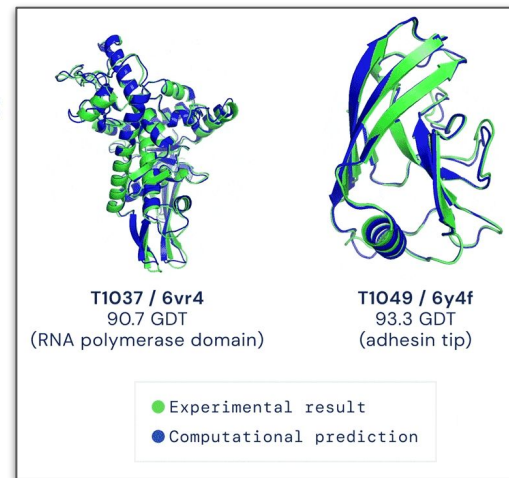
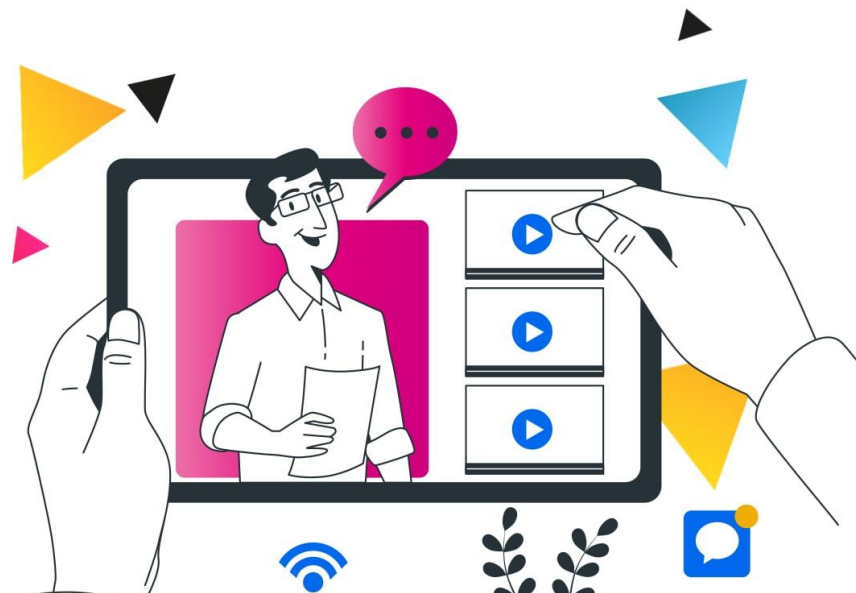


FIGURE 1: COMPLEX 3D SHAPES EMERGE FROM A STRING OF AMINO ACIDS.



Ваши вопросы?



Установка Python

Установка Python



Последняя актуальная версия Python 3.10, скачать его можно на официальном сайте <https://www.python.org/downloads/>.

The screenshot shows the Python.org website. At the top, there's a navigation bar with links: About, Downloads, Documentation, Community, Success Stories, News, Events. Below this, a large banner features the text "Download the latest source release" and a button "Download Python 3.10.7". To the right of the text is an illustration of two parachutes carrying boxes. Below the banner, there's a section titled "Active Python Releases" with a link to the Python Developer's Guide. At the bottom, there's a table with the following data:

Python version	Maintenance status	First released	End of support	Release schedule
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569
3.7	security	2018-06-27	2023-06-27	PEP 537
2.7	end-of-life	2010-07-03	2020-01-01	PEP 373

Среды разработки для Python

Среды разработки для Python

Где и как можно редактировать код?

1. любой текстовый редактор
2. PyCharm (<https://www.jetbrains.com/ru-ru/pycharm/>)
3. Jupyter Notebook (<https://jupyter.org/>)
4. Visual Studio Code (<https://code.visualstudio.com/>)
5. командная строка

```
C:\Windows>python  
>>> exit()
```

6. Anaconda (дистрибутив языков программирования Python и R, включающий набор популярных свободных библиотек, объединённых проблематиками науки о данных и машинного обучения: <https://www.anaconda.com/products/individual>)
7. Google Colab (<https://colab.research.google.com/>)



Visual Studio Code



Многофункциональная среда PyCharm



The screenshot displays the PyCharm IDE interface with the following components:

- Main Editor:** Shows Python code for processing quotations. A TODO comment is present: `# TODO Cover these lines with tests` on line 392.
- Coverage Window:** Displays coverage statistics for the current run.

Element	Statistics, %
100% files, 92% lines covered in 'talon'	
signature	100% files, 91% lines covered
__init__.py	100% lines covered
constants.py	100% lines covered
html_quotations.py	98% lines covered
quotations.py	98% lines covered
utils.py	48% lines covered
- Run Window:** Shows the test results for the 'NoseTest_run' configuration.

Done: 111 of 111 (9.763 s)

Ran 111 tests in 2.117s

OK

Process finished with exit code 0
- Test Results Tree:** Lists the executed tests, including `tests.signature.learning.database`, `tests.signature.learning.feature`, `tests.signature.learning.helper`, `tests.signature.bruteforce_test`, `tests.signature.extraction_test`, `tests.html_quotations_test`, `test_quotation_splitter_insid`, and `test_quotation_splitter_outsid`.

Интерактивный блокнот Jupyter Notebook



jupyter tutorial Last Checkpoint: 3 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

PyCon 2018: Using pandas for Better (and Worse) Data Science

GitHub: <https://github.com/justmarkham/pycon-2018-tutorial>

```
In [1]: import matplotlib.pyplot as plt
import pandas as pd
pd.__version__
```

```
Out[1]: '0.24.1'
```

Dataset: Stanford Open Policing Project ([video](#))

```
In [2]: # ri stands for Rhode Island
ri = pd.read_csv('police.csv')
```

```
In [3]: # what does each row represent?
ri.head()
```

```
Out[3]:
```

	stop_date	stop_time	county_name	driver_gender	driver_age_raw	driver_age	driver_race	violation_raw	violation	search
0	2005-01-02	01:55	NaN	M	1985.0	20.0	White	Speeding	Speeding	
1	2005-01-18	08:15	NaN	M	1965.0	40.0	White	Speeding	Speeding	
2	2005-01-23	23:15	NaN	M	1972.0	33.0	White	Speeding	Speeding	
3	2005-02-20	17:15	NaN	M	1986.0	19.0	White	Call for Service	Other	

JupyterLab позволяет организовать рабочую зону с помощью блокнотов, текстовых файлов, терминалов и выводов блокнотов.

File Edit View Run Kernel Tabs Settings Help

notebooks

Name Last Modified

- Data.ipynb an hour ago
- Fasta.ipynb a day ago
- Julia.ipynb a day ago
- Lorenz.ipynb seconds ago
- R.ipynb a day ago
- iris.csv a day ago
- lightning.json 9 days ago
- lorenz.py 3 minutes ago

In this Notebook we explore the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

Let's call the function once to view the solutions. For this set of parameters, we see the trajectories swirling around two points, called attractors.

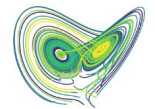
```
In [4]: from lorenz import solve_lorenz
t, x_t = solve_lorenz(N=10)
```

Output View

sigma 10.00

beta 2.67

rho 28.00



```
9 def solve_lorenz(N=10, max_t=10, sigma=10.0, beta=2.67, rho=28.0):
10     """plot a solution to the Lorenz differential equations."""
11     fig = plt.figure()
12     ax = fig.add_axes([0, 0, 1, 1], projection='3d')
13     ax.axis('off')
14
15     # prepare the axes limits
16     ax.set_xlim((-25, 25))
17     ax.set_ylim((-35, 35))
18     ax.set_zlim((5, 55))
19
20     # Compute the time-derivative of a Lorenz system.
21     def lorenz_deriv(x, y, z, t0, sigma=sigma, beta=beta, rho=rho):
22         """Compute the time-derivative of a Lorenz system."""
23         return [sigma * (y - x), x * (rho - z) - y, x * y - beta * z]
24
25     # Choose random starting points, uniformly distributed from -15 to 15
26     np.random.seed(1)
27     x0 = -15 + 30 * np.random.random((N, 3))
28
```

Многофункциональная среда Visual Studio Code



```
7  """ [markdown]
8  # Creating dataframe
9
10 Run Cell | Run Above | Debug cell
11 """
12 df = pd.DataFrame(data=np.random.randn(2000),
13                   index=pd.date_range('2001-01-01', periods=periods, other="non existent"),
14                   columns=['A', 'B'])
15
16 def gm(df, const):
17     v = (((df.A + df.B) + 1).cumprod() - 1) * const
18     return v.iloc[-1]
19
20 Run Cell | Run Above
21 """ [markdown]
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

example.py 1

Undefined variable: 'periods' Python(undefined-variable) [12, 47]

Python 3.7.3 64-bit 0 1

File Edit Selection View Go Debug Terminal Help HelloJupyter.ipynb - Jupytertest - Visual Studio Code

Variables

Name	Type	Count	Value
msg	str	11	'Hello world'
x	ndarray	(100,)	array([0. , 0.202002 , 0.404004 , ...])

Jupyter Notebook Editor

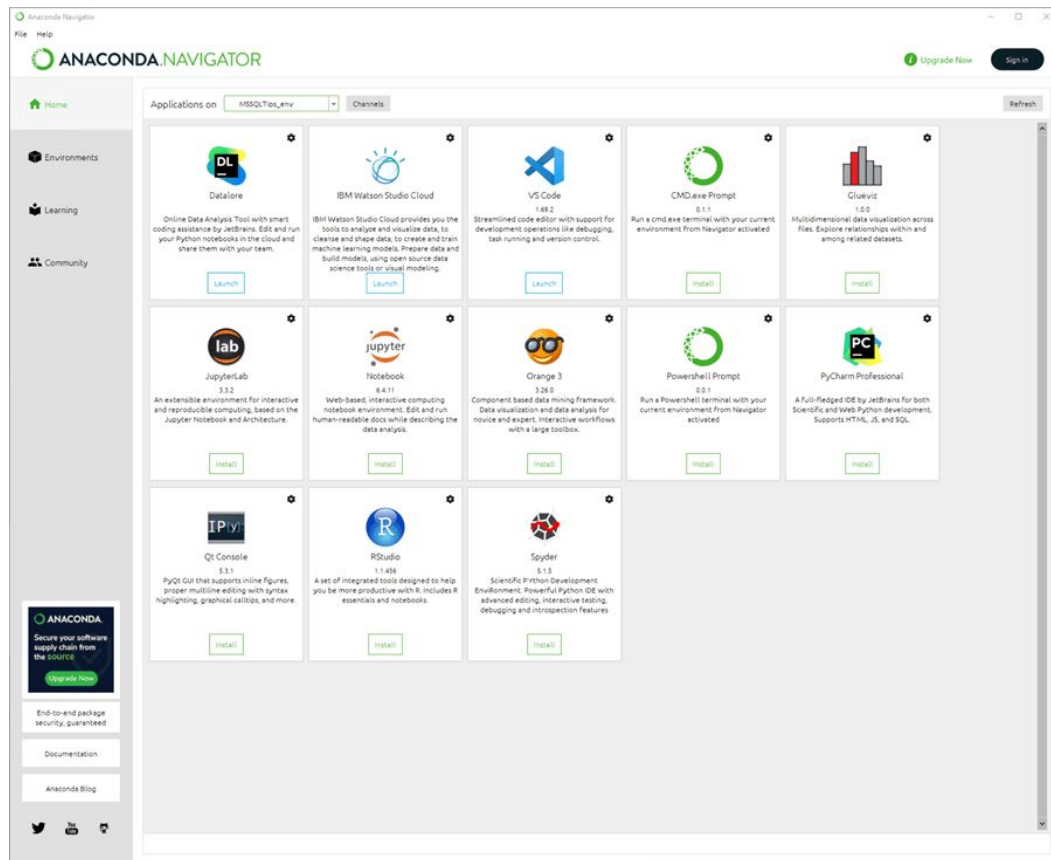
The Visual Studio Code Python extension supports Jupyter notebooks. You can use the Notebook Editor to create, view, modify, and run a Jupyter notebook and its code cells.

```
[2] import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 20, 100) # Create a list of evenly-spaced numbers over the range
plt.plot(x, np.sin(x))     # Plot the sine of each x point
plt.show()                 # Display the plot
```

Python 3.7.3 64-bit (base: conda) 0 1 jupyter HelloJupyter.ipynb

Дистрибутив Anaconda





Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share 

Table of contents 

+ Code + Text  Copy to Drive

Connect  Editing 

- Getting started
- Data science
- Machine learning
- More Resources
- Featured examples
- Section

What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
    seconds_in_a_day
```

86400

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Variables that you define in one cell can later be used in other cells:

```
[ ] seconds_in_a_week = 7 * seconds_in_a_day
    seconds_in_a_week
```

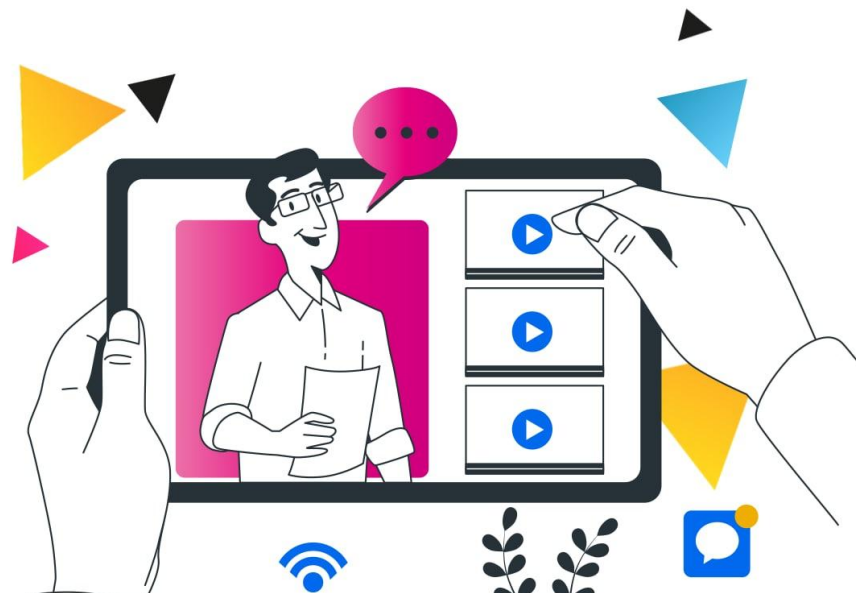
604800

Colab notebooks allow you to combine **executable code** and **rich text** in a single document, along with **images**, **HTML**, **LaTeX** and more. When you create your own Colab notebooks, they are stored in your Google Drive account. You can easily share your Colab notebooks with co-workers

<>



Ваши вопросы?



Встроенные типы данных

Встроенные типы данных

В сценариях Python все есть **объект**.
Встроенные типы данных:

- Числа (int)
- Строки (str)
- Списки (list)
- Словари (dict, OrderedDict)
- Кортежи (tuple)
- Файлы (handle)
- Множества (set, Collections)
- Булевы типы (bool)
- Сами типы (int, str, list, dict, tuple, ...)
- None, True, False
- Типы программных единиц (функции, модули, классы)

Источником типа является синтаксис.

Сама переменная — это всего лишь **ссылка** на объект. Операция присваивания - это выделение памяти под хранение ссылки на некоторый объект.

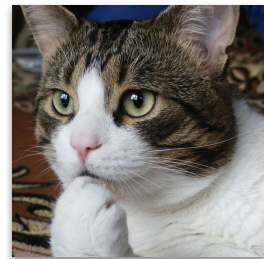
После объявления операции присваивания объект получает свой набор операций, характерный для его типа (**динамическая типизация**).

Одни и те же операторы с разными типами делают разные операции (**полиморфизм**).

Потренируемся в определении типов данных

Какой тип имеет переменная «None»?

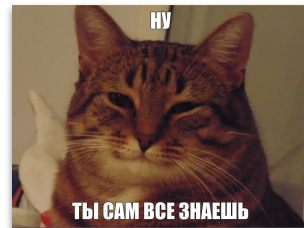
```
type (None)
```



Потренируемся в определении типов данных

Какой тип имеет переменная «None»?

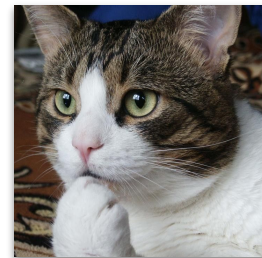
```
type (None)
```



Ответ: *NoneType*

Отличие операторов «==» и «is»

Чем отличаются операторы «==» и «is»?



Отличие операторов «==» и «is»

Чем отличаются операторы «==» и «is»?

Оператор «is» сравнивает **идентификаторы** двух объектов, а оператор «==» сравнивает **значения** двух объектов.

Оператор «is» принимает значение true, если переменные по обе стороны оператора указывают на **один и тот же объект**, и false в противном случае.

Оператор «==» используется, когда **значения** двух операндов равны, тогда условие становится истинным.

Отличие операторов «==» и «is»

Чем отличаются операторы «==» и «is»?

Например,

```
a = 10  
b = 100/10  
print(a is b)  
print(a == b)
```

Результат:

False

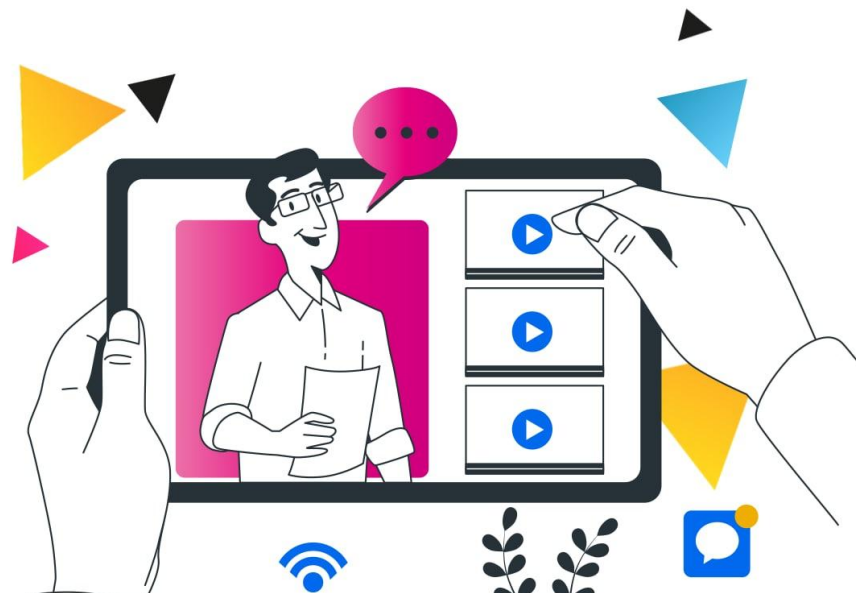
True

```
id(a), id(b)
```

(11122880, 139675133429680)



Ваши вопросы?



Практика в решении задач

Задача

Есть коэффициенты квадратного уравнения. Например, $a, b, c = 5., 2., 1.$

Необходимо найти его корни. Уравнение имеет вид: $ax^2 + bx + c = 0$, где a, b, c — коэффициенты.

1. Найдите дискриминант по формуле $D = b^2 - 4ac$
2. Если дискриминант получится меньше 0, напишите «Решений нет»
3. В противном случае, если дискриминант равен 0, найдите корень уравнения по формуле $x = -b/(2a)$ и напечатайте «Найдено одно решение <значение корня>»
4. Во всех остальных случаях найдите два корня уравнения по формулам, напечатайте «Найдено два решения <значение корня 1>; <значения корня 2>»

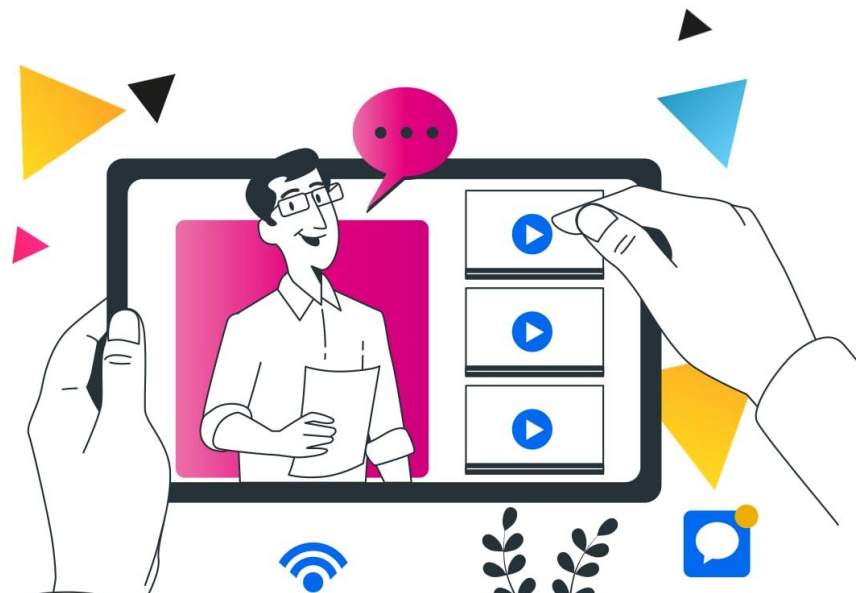
Формулы для нахождения корней
квадратного уравнения

$$x_1 = \frac{-b + \sqrt{D}}{2a} \text{ и } x_2 = \frac{-b - \sqrt{D}}{2a};$$

Решение задачи

```
# a, b, c = 5., 2., 1. # Решений нет
# a, b, c = 1., 2., 1. # Найдено 1 решение: -1.0
# a, b, c = 1., 2., 10. # Решений нет
a, b, c = 1., 2., -9. # Найдено 2 решения: 2.1622776601683795,
-4.16227766016838
d = b**2-4*a*c
if d < 0: print('Решений нет')
elif d==0:
    x = -b/(2*a)
    print(f'Найдено 1 решение: {x} ')
else:
    x1 = (d**(.5)-b)/(2*a)
    x2 = (-d**(.5)-b)/(2*a)
    print(f'Найдено 2 решения: {x1}, {x2}')
```

Ваши вопросы?



Полезные выражения

Полезные выражения по теме занятия

Методы работы со строкой

Существует несколько способов форматирования строк:

- метод `print()` и конкатенация строк

```
name = "Eric"
age = 74
print('Hello, ', name, '. You are', age, '.')
```

0

```
name = "Eric"
age = 74
print('Hello, ' + str(name) + '. You are ' + str(age) + '.')
```

1

Методы работы со строкой

Существует несколько способов форматирования строк:

- метод `print()` и конкатенация строк
- использования специального знака `%` и Template-строка
- метод `format()`
- f-строка

```
name = "Eric"
age = 74
print('Hello, ', name, '. You are', age, '.')
```

```
name = "Eric"
age = 74
print('Hello, ' + str(name) + '. You are ' + str(age) + '.')
```


Полезные выражения по теме занятия

f-строка

Форматирование, которое появилось в Python 3.6. Этот способ похож на форматирование с помощью метода `format()`, но гибче, читабельней и быстрее

```
a = '1256' 1
f'{int(a)}'
```

```
name = "Eric" 0
age = 74
f"Hello, {name}. You are {age}."
```

```
a = f"{2 * 37}" 2
```

```
F"Hello, {name}. You are {age}." 3
```

```
name = "Eric" 4
profession = "comedian"
affiliation = "Monty Python"
message = (
    f"Hi {name}. \n"
    f"You are a {profession}. "
    f"You were in {affiliation}."
)
print(message)
```

Итоги занятия

Итоги занятия



1. Знаем сферы применения Python и почему язык так популярен
2. Посмотрели популярные задачи с применением Python
3. Узнали, как и с какого ресурса установить Python
4. Познакомились с актуальными средами разработки для Python
5. Попрактиковались в решении задач, использующих встроенные типы данных
6. Познакомились с выражениями, которые могут оказаться полезными при написании кода
7. Материалы семинара доступны по ссылкам:
 - a. https://colab.research.google.com/drive/1Lce6eZ7AidqsWWZWxxS_l58icn3Hlxpt?usp=sharing
 - b. <https://colab.research.google.com/drive/1At0-M4nNermc7ooBGzijQpgRa79LAKVU?usp=sharing>

Напишите в чат

Сможете ли вы использовать Python
в своей деятельности / учебе / работе?



Дополнительные материалы

1. Марк Лутц. Изучаем Python
2. Дэн Бейдер. Чистый Python. Тонкости программирования для профи
3. Гарри Персиваль. Python. Разработка на основе тестирования
4. Андреас Мюллер и Сара Гвидо. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными
5. Прадик Джоши. Искусственный интеллект с примерами на Python
6. Брюс Питер, Брюс Эндрю. Практическая статистика для специалистов Data Science



Ваши вопросы?

