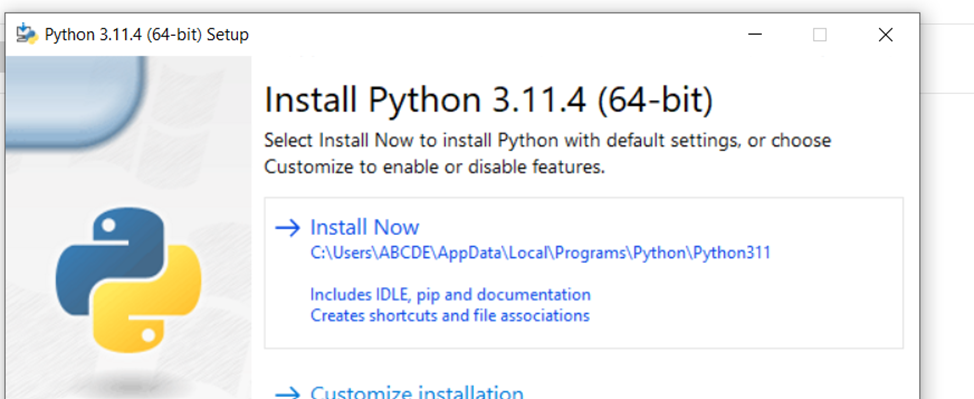
# Путь



Stuff: C/windows/system32/….

// Main: C:/users/ПК/AppData/Local/Programs/python311

// Main2: C:/users/AABBCCDD/AppData/Local/Programs/python311

Main

D:\stuff\programin\PY designes

Также Python работает через CMD (admin)

Путь конфига

C:\Users\ПК\.jupyter

Anaconda navigator и JN ярлыки:

C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Anaconda3 (64-bit)

# CMD

d: для перехода на d диск

cd \*путь\* - переход по пути

cd .. – как в командной строке вернуться выше по иерархии папок

# Python

1. python - для перехода на среду python в командной строке
2. Ctrl + Z, а затем нажать Enter. - выход из среды Python

python --version

посмотреть путь

import sys

print(sys.executable)

## CMD Не видит python

нужно переустановить библиотеку пайтона в ту же самую папку

# Jupyter Notebook

установка: pip install jupyter notebook

1. команда cd \*path\* открывает папку в который юпитер будет работать

cd C:\Users\ABCDE\Desktop\new folder\programin\Jupyter notebook

1. команда jupyter notebook (запускает JN в браузере) (ИНОГДА НАДО jupyter-notebook)
2. СTRL + C в CMD завершает работу JN / закрытие CMD также завершает JN

* можно установить Jupyter Notebook прямо на диск D. Вот как это сделать:

1. Открой командную строку (Command Prompt) или терминал (Terminal), в зависимости от операционной системы.
2. Введи следующую команду: pip install jupyter --install-option="--prefix=D:\stuff\programing\python" Эта команда установит Jupyter Notebook на диск D в папку D:\stuff\programing\python.

### Utils

ctrl + / - множественное комментирование

## JN настройка запуска

чтобы JN запускался сразу в браузере:

1. по пути вашего файла ввод: jupyter notebook --generate-config (ровно так, без доп пробелов)
2. перейдти на записанную папку
3. открыть .py через блокнот и найти строчку use\_redirect\_file и убрать в этой строчке # (начало строки должно быть без пробелов) + True ⇒ False

## открытие файлов

JN может открывать файлы только из корневой директории, откуда он был запущен

ML курс1:

cd /mnt/d/stuff/programin/JN,TC,SF\_UNIXvenv/#ML\_course1/JN – для ubuntu

## Команды

import os ⇒ import os # работает второй вариант

print os.getcwd() ⇒ print(os.getcwd()) #

## 

## Графики с несколькими осями

смотреть matplotlib для того чтобы код работал

как в matplotlib построить несколько графиков на одной плоскости, но чтобы ось x была 1, а осей y было несколько

Чтобы построить несколько графиков на одной плоскости с разными осями y, можно использовать функцию subplots из модуля matplotlib.pyplot.

Вот пример, который показывает, как это сделать:

import matplotlib.pyplot as plt

старое описание кода, в КОДЕ реализованы сдвиги осей, а здесь нет, надо добавить

# Создание плоскости

fig, ax1 = plt.subplots()

# Построение графика 1 (ось y1)

x = [1, 2, 3, 4, 5]

y1 = [1, 4, 9, 16, 25]

ax1.plot(x, y1, label='График 1')

ax1.set\_ylabel('Ось y1')

# Создание дополнительной оси y

ax2 = ax1.twinx()

# Построение графика 2 (ось y2)

y2 = [1, 8, 27, 64, 125]

ax2.plot(x, y2, label='График 2', color='orange')

ax2.set\_ylabel('Ось y2')

ax3 = ax2.twinx()

# Построение графика 3 (ось y3)

y3 = [1, 16, 81, 256, 625]

ax3.plot(x, y3, label='График 3', color='green')

ax3.set\_ylabel('Ось y3')

# Задание оси x

ax1.set\_xlabel('Ось x')

# Добавление легенды

ax1.legend(loc='upper left')

ax2.legend(loc='upper right')

# Отображение графиков

plt.show()

### код

fig, ax1 = plt.subplots()

x = [1, 2, 3, 4, 5]

y1 = [1, 4, 9, 16, 25]

ax1.plot(x, y1, label='График 1')

ax1.set\_ylabel('Ось y1')

ax2 = ax1.twinx()

y2 = [1, 8, 27, 64, 125]

ax2.plot(x, y2, label='График 2', color='orange')

ax2.spines['right'].set\_position(('outward', 40)) # Задаем отступ для оси ax2

ax2.set\_ylabel('Ось y2')

ax3 = ax1.twinx()

y3 = [1, 16, 81, 256, 625]

ax3.plot(x, y3, label='График 3', color='green')

ax3.spines['right'].set\_position(('outward', 80)) # Задаем больший отступ для оси ax3

ax3.set\_ylabel('Ось y3')

ax1.set\_xlabel('Ось x')

plt.show()

# Библиотеки

## matplotlib

1. зашел в нужную папку
2. создал venv
3. зашел в новую папку venv
4. активировал venv
5. зашел в jupyter notebook
6. pip install matplotlib  
   pip install numpy  
   %matplotlib inline

import numpy as np

import matplotlib.pyplot as plt

plt.style.use('seaborn-whitegrid') /// НЕ работает

1. вставил код

### ultimate graphics code:

%matplotlib inline

import numpy as np

import matplotlib.pyplot as plt

fig, ax1 = plt.subplots()

x = [1.71, 1.76, 1.84, 1.86, 1.91, 1.92, 1.98, 2.11, 2.14]

y1 = [79.17, 78.57, 75.1, 72.94, 73.46, 71.11, 70.71, 71.77, 71.81]

ax1.plot(x, y1, label='η(P2)')

ax1.set\_ylabel('η')

ax1.set\_yticklabels([])

ax2 = ax1.twinx()

y2 = [2.16, 2.24, 2.45, 2.55, 2.6, 2.7, 2.8, 2.94, 2.98]

ax2.plot(x, y2, label='P1(P2)', color='purple')

ax2.set\_ylabel('P1')

ax2.spines['left'].set\_position(('outward', 40)) # Установка отрицательного смещения для левой оси спинов

ax2.spines['left'].set\_color('black') # Настраиваем цвет линии оси спинов

ax2.yaxis.set\_label\_position('left') # Позиционирование меток делений оси налево

ax2.yaxis.set\_ticks\_position('left') # Позиционирование делений оси налево

ax2.set\_yticklabels([])

ax3 = ax1.twinx()

y3 = [1305, 1245, 1170, 1140, 1110, 1050, 1020, 960, 930]

ax3.plot(x, y3, label='n2(P1)', color='purple')

ax3.set\_ylabel('n2')

ax3.spines['left'].set\_position(('outward', 80)) # Установка отрицательного смещения для левой оси спинов

ax3.spines['left'].set\_color('black') # Настраиваем цвет линии оси спинов

ax3.yaxis.set\_label\_position('left') # Позиционирование меток делений оси налево

ax3.yaxis.set\_ticks\_position('left') # Позиционирование делений оси налево

ax3.set\_yticklabels([]) # Убрал надписи значений с оси Y

ax4 = ax1.twinx()

y4 = [12.5, 13.5, 15, 16, 16, 17.5, 18.5, 21, 22]

ax4.plot(x, y4, label='M2(P1)', color='purple')

ax4.set\_ylabel('M2')

ax4.spines['left'].set\_position(('outward', 110))

ax4.spines['left'].set\_color('black')

ax4.yaxis.set\_label\_position('left')

ax4.yaxis.set\_ticks\_position('left')

ax4.set\_yticklabels([])

ax5 = ax1.twinx()

y5 = [10.5, 11, 12, 12.5, 13, 13.5, 14, 15, 15.5]

ax5.plot(x, y5, label='I2(P1)', color='purple')

ax5.set\_ylabel('I2')

ax5.spines['left'].set\_position(('outward', 150)) # Установка отрицательного смещения для левой оси спинов

ax5.spines['left'].set\_color('black') # Настраиваем цвет линии оси спинов

ax5.yaxis.set\_label\_position('left') # Позиционирование меток делений оси налево

ax5.yaxis.set\_ticks\_position('left') # Позиционирование делений оси налево

ax5.set\_yticklabels([])

ax1.set\_xlabel('P2')

ax1.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left')

ax2.legend(bbox\_to\_anchor=(1.05, 0.9), loc='upper left')

ax3.legend(bbox\_to\_anchor=(1.05, 0.8), loc='upper left')

ax4.legend(bbox\_to\_anchor=(1.05, 0.7), loc='upper left')

ax5.legend(bbox\_to\_anchor=(1.05, 0.6), loc='upper left')

plt.show()

### прерывистые линии

Чтобы сделать линию прерывистой, вы можете использовать значение "dashed" для параметра linestyle. Например:

import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]

y = [1, 4, 9, 16, 25]

plt.plot(x, y, linestyle="dashed")

plt.show()

А чтобы сделать линию курсивной, вы можете использовать значение "dashdot" для параметра linestyle. Например:

import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]

y = [1, 4, 9, 16, 25]

plt.plot(x, y, linestyle="dashdot")

plt.show()

Вы также можете настроить стиль линии, используя комбинацию этих значений, например, "dotted-dash". Вот пример:

import matplotlib.pyplot as plt

x = [1, 2, 3, 4, 5]

y = [1, 4, 9, 16, 25]

plt.plot(x, y, linestyle="dotted-dash")

plt.show()

# Работа с Pip

установка pip возможна через modify офиц. установщика

ПОЧЕМУ ТО PIP НЕ РАБОТАЕТ В СMD (админ) , а в обычном CMD работает

https://yandex.ru/q/question/kak\_udalit\_biblioteku\_v\_python\_cherez\_pip\_909f8a3e/

Pip main

Pip uninstall

Pip --version - показывает версию

pip install --upgrade pip - обновление pip

## Конкретная папка установки

Чтобы установить библиотеки через pip в конкретную папку, можно воспользоваться параметром --target. Вот как это сделать:

Открой командную строку (Command Prompt) или терминал (Terminal), в зависимости от операционной системы.

Введи следующую команду: pip install <название\_библиотеки> --target=<путь\_к\_папке> Замени <название\_библиотеки> на имя библиотеки, которую ты хочешь установить, а <путь\_к\_папке> на путь к папке, в которую ты хочешь установить эту библиотеку.

Например:

pip install numpy --target=D:\stuff\programing\python\libs

pip install numpy --target=D:\programs\PROGRAMMING\Python libs

# Удаление и установка

При установке библиотеки с помощью pip по умолчанию она будет установлена в директорию site-packages вашей установки Python. Расположение этой директории зависит от вашей операционной системы и способа установки Python.

Чтобы узнать местоположение директории site-packages в вашей установке Python, вы можете выполнить следующую команду в командной строке Python:

python

Copy

import site

print(site.getsitepackages())

# venv: Virtual env

1.Нужно установить среду virtualenv

pip install virtualenv

2. Затем создать папку для конкретного проекта, куда мы будем устанавливать библиотеку, которая будет действовать ТОЛЬКО в этой папке.

C:\папка проектов\>python -m venv \*имя одним словом\* - создание папки \*\*\* venv в папке проектов

3. Активация

cd переходим в виртуальную папку (именно venv)

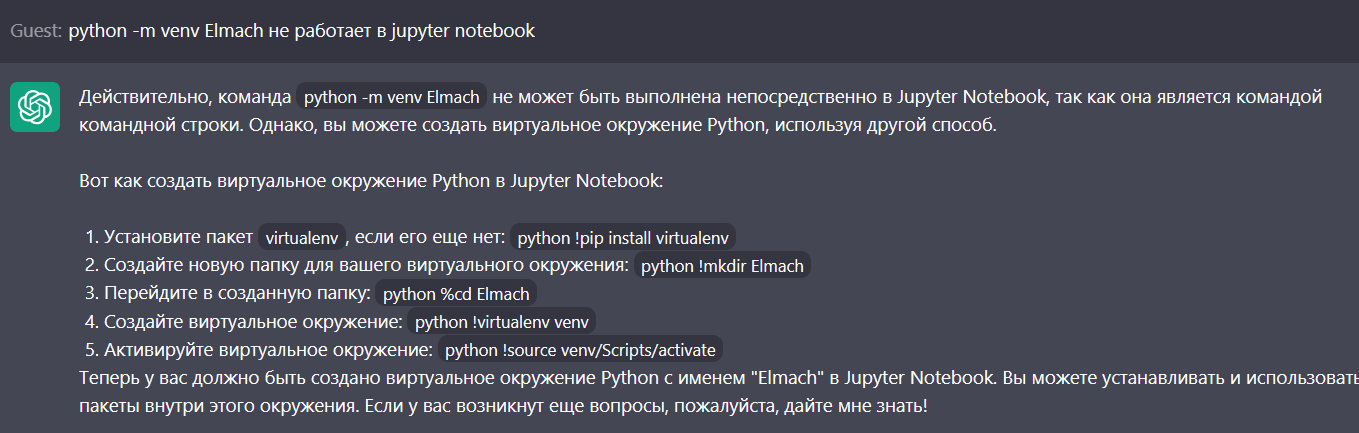
На windows : Scripts\activate

4. Деактивация

cd переходим в виртуальную папку (именно venv)

На windows: Scripts\deactivate

### Venv в JN



Действительно, команда python -m venv Elmach не может быть выполнена непосредственно в Jupyter Notebook, так как она является командой командной строки. Однако, вы можете создать виртуальное окружение Python, используя другой способ.

Вот как создать виртуальное окружение Python в Jupyter Notebook:

1. Установите пакет virtualenv, если его еще нет: python !pip install virtualenv
2. Создайте новую папку для вашего виртуального окружения: python !mkdir Elmach
3. Перейдите в созданную папку: python %cd Elmach
4. Создайте виртуальное окружение: python !virtualenv venv
5. Активируйте виртуальное окружение: python !source venv/Scripts/activate

Теперь у вас должно быть создано виртуальное окружение Python с именем "Elmach" в Jupyter Notebook. Вы можете устанавливать и использовать пакеты внутри этого окружения. Если у вас возникнут еще вопросы, пожалуйста, дайте мне знать!

### Установка другой версии пайтона на виртуальную среду:

Для Windows.

1. Установите ту версию python, которую хотите использовать в виртуальном пространстве. ВАЖНО! Если у вас есть основаная версия и вы не хотите конфликтов, то на моменте установки снимите галочку с опции Add python 3.6 to PATH
2. Далее введите (при условии, что у вас установлен python)

py -3.6 -m venv env

И вы получите виртуально пространство с python 3.6, которое будет лежать в папке env

## Установленные библиотеки

virtualenv (выше)

JN: pip install jupyter notebook

## Показать установленные библиотеки

Eсли вы планируете переустановить Python после удаления, я рекомендую сначала сохранить список всех установленных библиотек, чтобы после установки Python вы могли легко восстановить их. Для этого в командной строке введите команду:

basic

Copy

pip freeze > requirements.txt

Эта команда создаст файл requirements.txt, содержащий список всех установленных библиотек и их версий. После переустановки Python вы можете легко установить все библиотеки из этого файла, используя команду:

basic

Copy

pip install -r requirements.txt

Эта команда установит все библиотеки, перечисленные в файле requirements.txt.

## Диск D

Если вы хотите полностью перенести установку Python и все его библиотек на диск D, вам нужно выполнить следующие шаги:

1. Деинсталлируйте текущую версию Python через "Установку и удаление программ" в Панели управления.
2. Скачайте установщик Python с официального сайта ([https://www.python.org/downloads/) ↗](https://www.python.org/downloads/)) и запустите его.
3. В процессе установки выберите опцию "Customize installation" (Настроить установку).
4. Укажите установочный путь на диск D, например: "D:\Python".
5. Убедитесь, что опция "Add Python to environment variables" (Добавить Python в переменные среды) выбрана.
6. Нажмите "Install" (Установить), чтобы начать установку.

После установки Python на диск D, вы также можете установить любые нужные вам библиотеки с помощью команды "pip install" в командной строке. Они будут установлены в директорию "D:\Python\Lib\site-packages".

# Запуск из python.exe и менеджмент пути проекта

1. я установил пайтон на D и после запуска именно python.exe, а не командной строки он не команду "cd D:....." \ "cd C:....." - говорит, что С и D это invalid syntax

Команда "cd" в Python не используется для изменения директории в файловой системе компьютера. Вместо этого, вы можете использовать функцию os.chdir() из стандартной библиотеки Python os, чтобы изменить текущую рабочую директорию.

Например, если вы хотите изменить текущую директорию на "D:\my\_project", вы можете использовать следующий код:

python

import os

os.chdir('D:\\my\_project')

Обратите внимание, что в строке 'D:\\my\_project' используются двойные обратные слеши для экранирования, потому что одинарный обратный слеш \ в Python используется для символов экранирования.

Если вы хотите узнать текущую рабочую директорию, вы можете использовать функцию os.getcwd(), например:

python

import os

current\_dir = os.getcwd()

print(current\_dir)

Этот код выведет на экран текущую рабочую директорию.

Надеюсь, это поможет вам изменить текущую рабочую директорию в Python на диске D.

2. почему если при запуске из командной строки я сразу имею папку в которой работаю и могу "cd C:...." (при этом работать в папке из D я не могу). А если запускаю пайтон из python.exe, то нет

Когда вы запускаете командную строку, она открывается в рабочей директории по умолчанию, которая обычно соответствует директории профиля пользователя. Эту директорию можно изменить, используя команду "cd" в командной строке.

Однако, когда вы запускаете интерпретатор Python из файла python.exe без аргументов командной строки, он не знает, в какой директории вы хотите работать. По умолчанию, Python будет искать модули и файлы в рабочей директории, которая соответствует директории, из которой был запущен python.exe. Если вы хотите изменить рабочую директорию в Python, вы можете использовать функцию os.chdir() из модуля os, как я показал в предыдущем ответе.

Чтобы запустить интерпретатор Python в определенной директории, вы можете запустить командную строку и выполнить команду cd для изменения рабочей директории на нужную, а затем выполнить команду python для запуска интерпретатора Python в этой директории. Например:

Copy

cd D:\my\_project

python

# Переход на D:

1. Если все еще не работает, попробуйте выполнить команду D: в командной строке. Эта команда переключает текущий диск на D:, и после этого команда cd должна работать для директорий на диске D:.

# !Разница между python.exe и CMD

# WSL Ubuntu

## Доступ к генеральному диску Ubuntu

Для того чтобы получить доступ к файлам главного диска дистрибутива Ubuntu, который находится на виртуальном жестком диске с расширением .vhdx в Windows, вам нужно выполнить несколько шагов.

1. Установите программу "DiskInternals Linux Reader" на ваш компьютер Windows. Это бесплатная программа, которая позволяет получить доступ к файлам на разделах Linux, включая ext4.
2. Запустите программу "DiskInternals Linux Reader" и выберите в меню "File" опцию "Open Disk Image".
3. В диалоговом окне "Open Disk Image" выберите файл .vhdx, который содержит дистрибутив Ubuntu, и нажмите "Open".
4. После того, как файл .vhdx будет загружен в программу, выберите раздел, который содержит файлы дистрибутива Ubuntu, и нажмите "Mount".
5. После того, как раздел будет смонтирован, вы сможете просматривать его содержимое через файловый менеджер Windows.
6. Если вы хотите изменить файлы на разделе, вам нужно будет сохранить их на ваш компьютер Windows, внести необходимые изменения, а затем загрузить их обратно на раздел через программу "DiskInternals Linux Reader".

Обратите внимание, что при работе с файлами на разделе Linux в Windows может быть необходимо использовать права администратора, чтобы иметь доступ к некоторым файлам и папкам.

## ##Установка

Windows Subsystem for Linux

некоторые приложения и библиотеки не поддерживаются на Win

1. можно установить WSL через cmd на виндовс (сборки 1903 и выше | можно проверить winver.exe в поиске) через cmd командой: wsl --install

# можно через Microsoft Store

## Команды

открыть папку

1. cd /mnt
2. cd ‘диск’
3. cd ‘путь’

Запустить python : python3

Для завершения работы с python : Ctrl + D

## изменение директории

cd /mnt/d/stuff/programin/JN,TC,SF\_UNIXvenv

JN для МL курс1 :

cd /mnt/d/stuff/programin/JN,TC,SF\_UNIXvenv/#ML\_course1/JN

активация виртуальной среды

source bin/activate

## Библиотеки

Turi create : (venv) pip install -U turicreate

## установка пайтона

sudo apt update

sudo apt install python3

sudo apt-get remove python3

sudo apt-get autoremove

sudo apt-get purge python3

## Путь Ubuntu

C:\Users\ABCDE\AppData\Local\Packages\CanonicalGroupLimited.Ubuntu20.04LTS\_79rhkp1fndgsc

основные пути в LocalState

## проблемы с Pip

[Python 3.x - python3-pip не имеет кандидата на установку - Stack Overflow](https://stackoverflow.com/questions/63591163/python3-pip-has-no-installation-candidate)

* sudo add-apt-repository universe
* sudo apt-get update
* sudo apt install python3-pip

# Python3

Изменение директории

import os

os.chdir('/path/to/new/directory')

Важно учитывать, что в среде ubuntu путь указывается по другому:

/mnt/d/…..

Вы можете получить текущую рабочую директорию с помощью функции

import os

os.getcwd().

# Turi create

Список базовых функций TC( + основы Python) по: D:\stuff\programin\JN,TC,SF\_UNIXvenv\#ML\_course1\JN

## Функции

работа со строками и присвоение в оригинальные строки:

1. def transform\_country(country):

if country == 'USA':

return 'United States'

else:

return country

2.sf['Country'].apply(transform\_country)

3.sf['Country'] = sf['Country'].apply(transform\_country)

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

О том как можно построить график из прямоугольников и как использовать данные из Excel в пайтоне можно посмотреть в D:\stuff\programing\python\PY designes\matplotlib\matplotlib\_venv\OPD.ipynv

# \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Программы

## Текущая цена криптовалют

1)файл jupyter notebook находится в D:\stuff\1 code\programing\python\PY designes\crypto currency\crypto price

2) требуется установать pip install requests

я установил jupyter и requests напрямую в библиотеку корневой папки пайтона, чтобы они были доступны всегда, а не в venv

3) Для работы API я взял ключ из моего аккаунта coinmarketcap api <https://pro.coinmarketcap.com/account>

4) код:

#нужно установить pip requests

import requests

API\_KEY = "4f308210-a355-475d-b073-366d6a8b3f7d"

#взял ключ из моего аккаунта coinmarketcap api https://pro.coinmarketcap.com/account

def get\_crypto\_prices(crypto\_list):

url = "https://pro-api.coinmarketcap.com/v1/cryptocurrency/quotes/latest"

headers = {

"X-CMC\_PRO\_API\_KEY": API\_KEY

}

params = {

"symbol": ",".join(crypto\_list),

"convert": "USD"

}

response = requests.get(url, headers=headers, params=params)

data = response.json()

prices = {}

for crypto in crypto\_list:

if crypto.upper() in data["data"]:

prices[crypto] = data["data"][crypto.upper()]["quote"]["USD"]["price"]

return prices

crypto\_list = ["btc", "eth", "ton", "apt", "stg", "ape", "xrp", "blur", "sol", "arb", "hft", "strk", "w", "zk", "xai", "op", "matic", "cyber", "xlm", "avax", "uni", "aave", "dot", "link", "ada", "wld", "hbar"]

prices = get\_crypto\_prices(crypto\_list)

for crypto, price in prices.items():

print(f"{crypto}: ${price}")

print(f"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

for crypto, price in prices.items():

rounded\_price = round(price, 3) # Округляем до двух знаков после запятой

formatted\_price = f"{rounded\_price}".replace('.', ',')

print(f"{formatted\_price}")