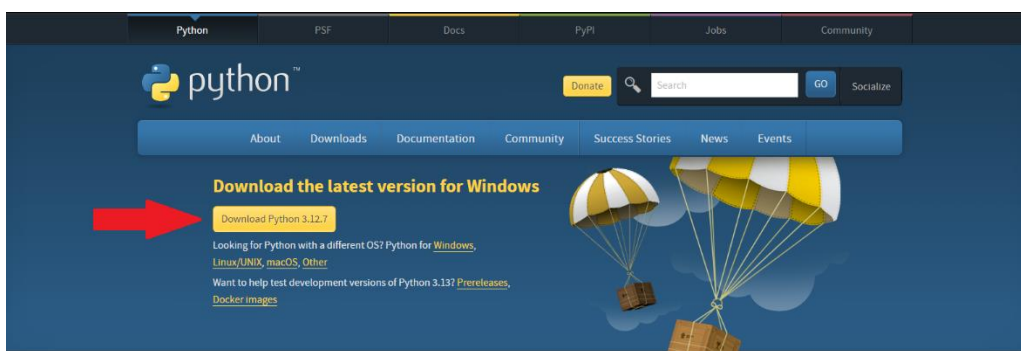


Windows

Эта часть гайда считается основной и описывает процесс установки Python 3 (а конкретно – CPython) на машину под управлением Windows 10. Если вы планируете устанавливать Python 3 под Linux, все равно прочитайте эту часть, так как часть про Linux будет содержать лишь отличия по отношению к этой.

Базовый Python.

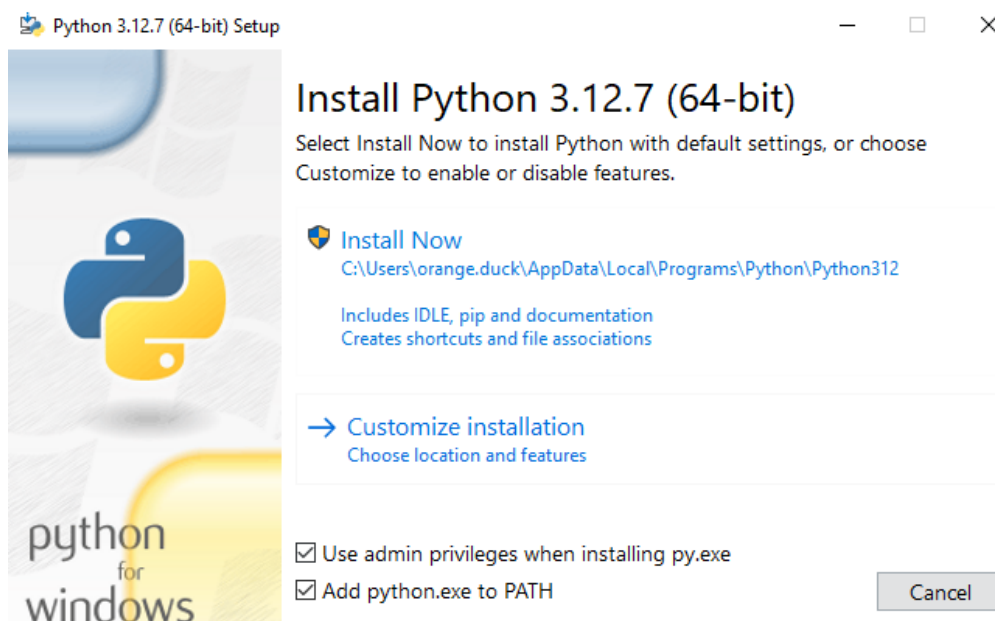
Скачать новейшую версию Python можно на официальном сайте – <https://www.python.org/downloads/>.



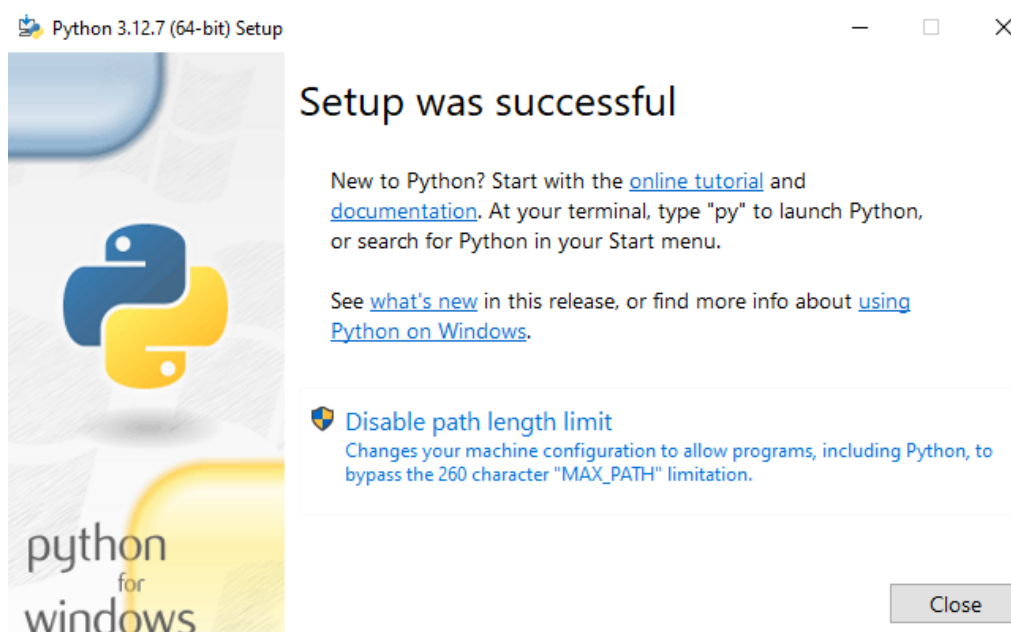
The screenshot shows the Python.org website. A red arrow points to the 'Download Python 3.12.7' button. Below the button, there are links for other operating systems and pre-releases. Below the main content, there is a table titled 'Active Python Releases'.

Python version	Maintenance status	First released	End of support	Release schedule
3.13	prerelease	2024-10-07 (planned)	2029-10	PEP 719
3.12	bugfix	2023-10-02	2028-10	PEP 693
3.11	security	2022-10-24	2027-10	PEP 664
3.10	security	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 586
3.8	security	2019-10-14	2024-10	PEP 569

Нажимаете кнопку Download Python 3... и после окончания скачивания запускаете инсталлятор. В нем в первом же окне прожмите галочку «Add python.exe to PATH». Так путь к интерпретатору Python будет сразу при установке добавлен в переменные окружения, иначе вам всегда придется указывать полный путь к интерпретатору при запуске программ. Если вы пока не понимаете, что это значит – тогда **обязательно прожмите галочку**.

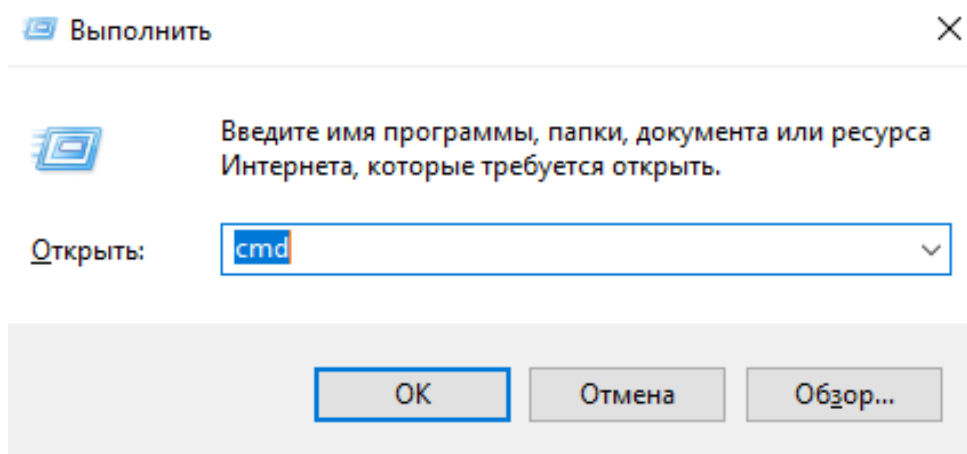


Затем нажимаете Install Now и ждете окончания установки. Также на всякий случай запишите путь установки (если вы не меняли его, то по умолчанию он будет выглядеть как **С:\<ваше_имя_пользователя>\AppData\Local\Programs\Python\Python<номер_вашей_версии_без_точек_и_последнего_числа>**) – может пригодиться, если при установке возникнут проблемы.



Кнопка «Disable path length limit» опциональна, хуже от нее не будет, но и вряд ли вы в рамках курса столкнетесь с ситуацией, когда вам не хватит длины пути в 260 символов.

Теперь проверим корректность установки. Для этого откройте командную строку (сочетанием клавиш **Win+R**, откроется окошко «Выполнить», в нем введите **cmd** и нажмите **Enter**).



В открывшемся окне терминала вбиваете **python --version** и жмете **Enter**. Должна появиться следующая надпись:

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\orange.duck>python --version
Python 3.12.7

C:\Users\orange.duck>_
```

То же самое с **pip** (пакетный менеджер для Python):

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

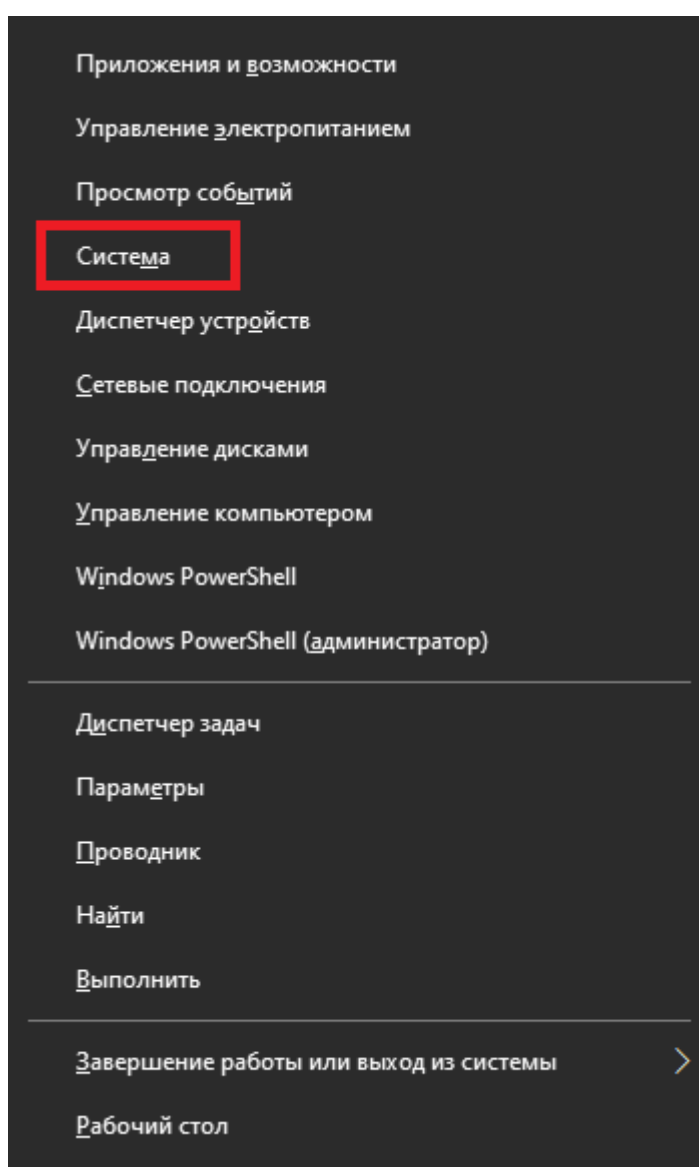
C:\Users\orange.duck>python --version
Python 3.12.7

C:\Users\orange.duck>pip --version
pip 24.2 from C:\Users\orange.duck\AppData\Local\Programs\Python\Python312\Lib\site-packages\pip (python 3.12)

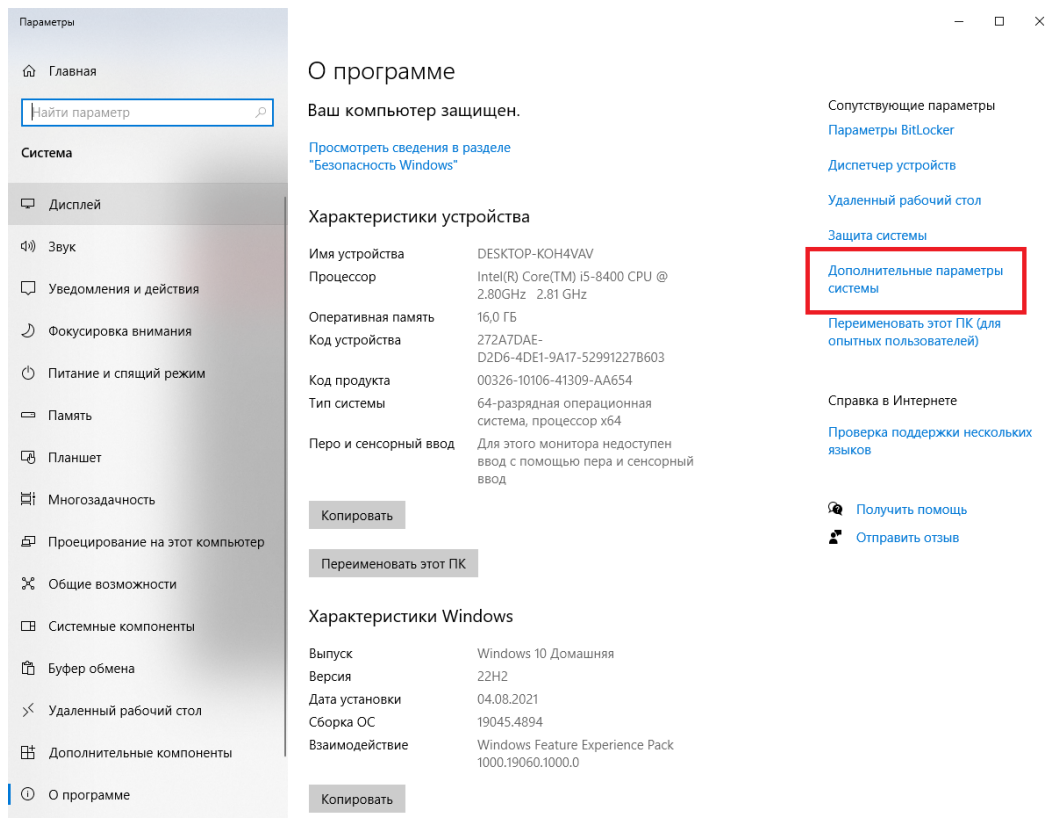
C:\Users\orange.duck>
```

Если все так, то поздравляю, вы успешно установили Python.

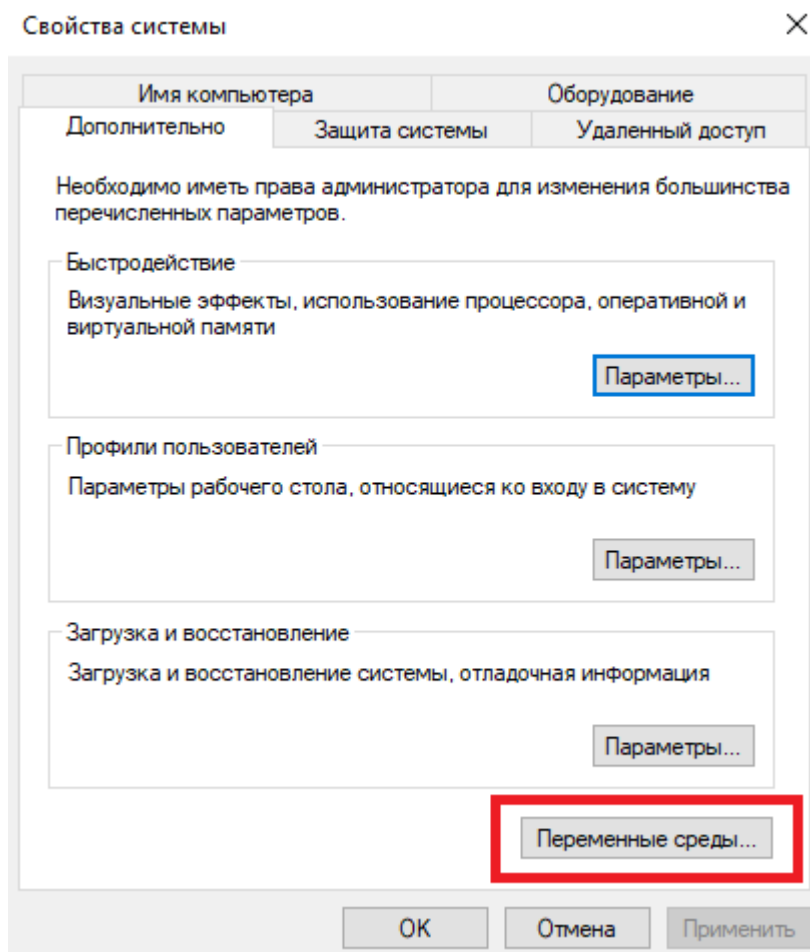
Если же вы видите надпись вида **«python не является внутренней или внешней командой, исполняемой программой или пакетным файлом»** – значит, у вас возникла проблема с добавлением Python в PATH. Произойти это могло по двум причинам – либо вы не прожали галочку, либо при добавлении в PATH пути к интерпретатору произошла какая-то ошибка. В таком случае придется добавить путь в PATH вручную (если, конечно, вы не решили не добавлять его туда сознательно – но в таком случае вы и так знаете, что с этим делать). Для этого нам как раз пригодится путь, который вы записали в начале установки (записали же, правда?). Скопируйте его, затем нажмите **Win + X**, откроется контекстное меню, в котором вам нужно выбрать **«Система»**.



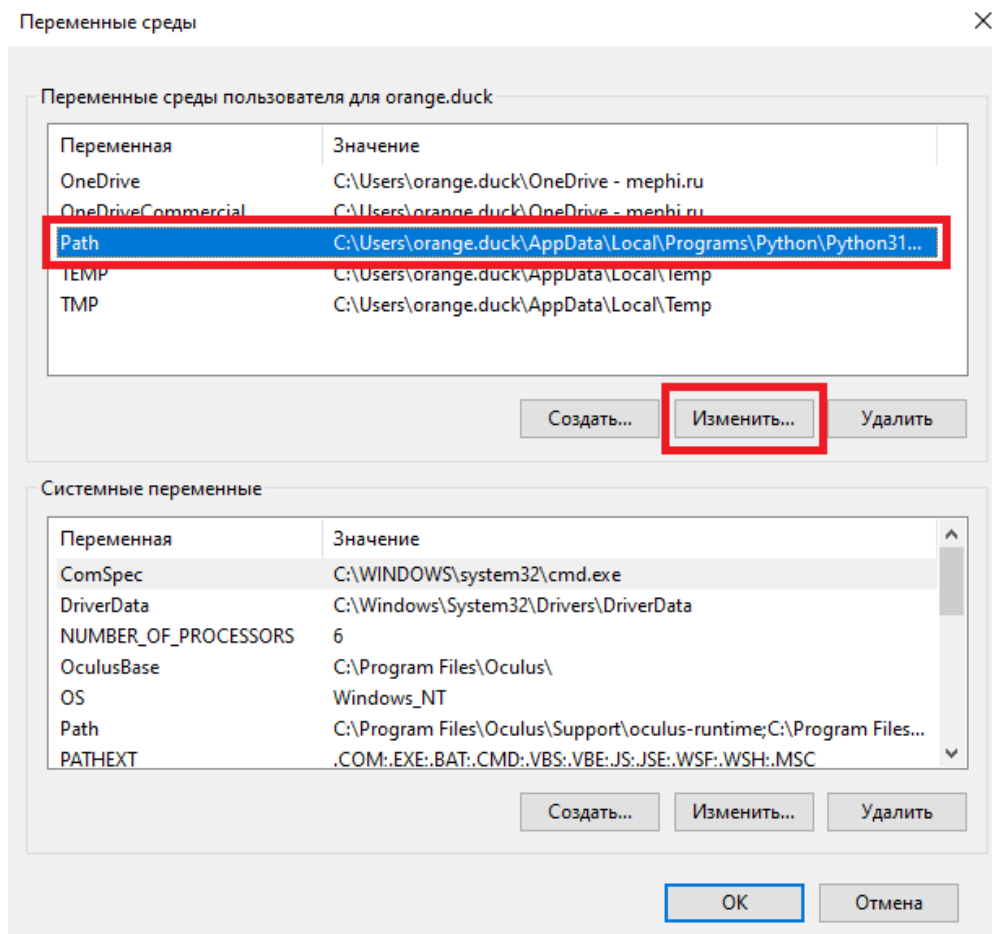
Затем в открывшемся окне ищите справа **«Дополнительные параметры системы»**, нажимаете. Если у вас это окошко выглядит не так, поищите, как для конкретно вашей версии Windows редактировать переменные окружения, этот раздел может быть спрятан где-то еще.



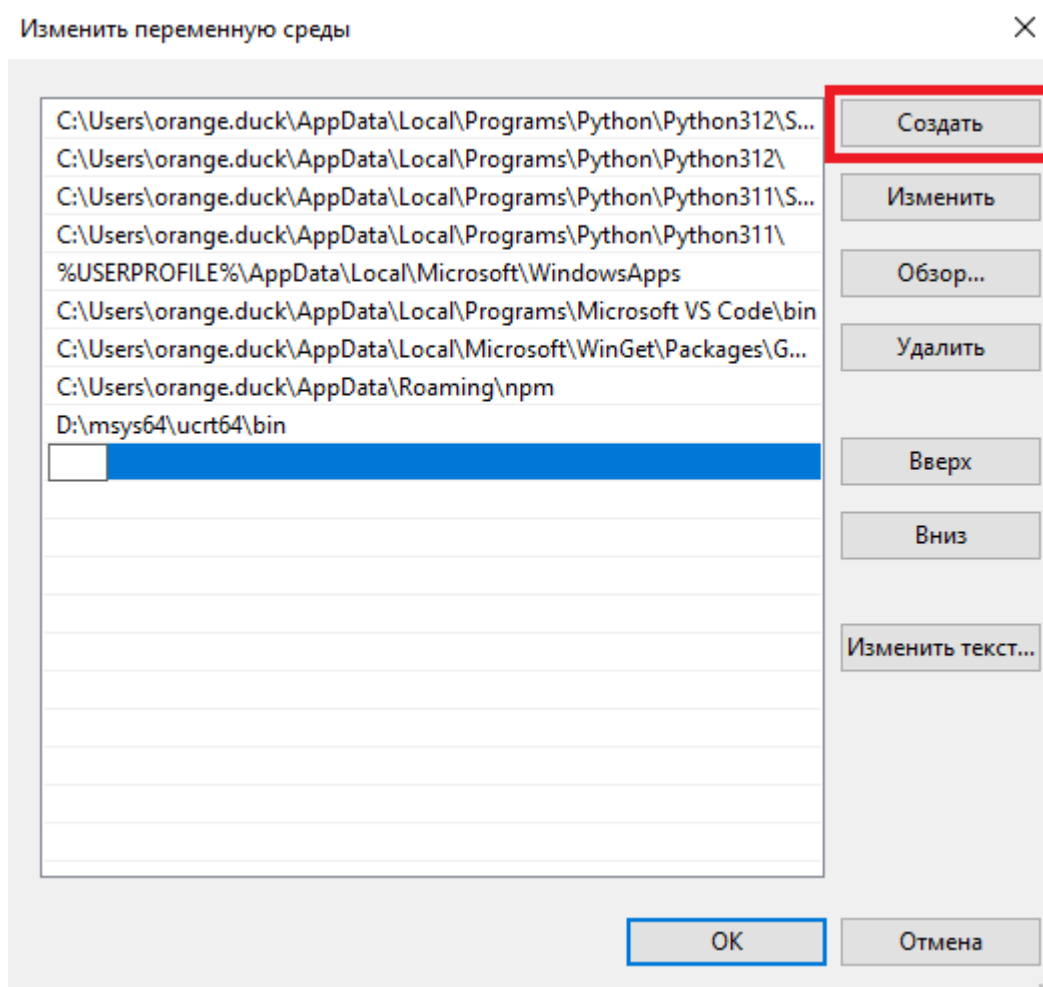
В очередном открывшемся окошке нажимаете «Переменные среды».



Теперь Path и «Изменить». Почти все!



«Создать» и копируете свой путь (картинка ниже, съехала на другую страницу). Еще раз «Создать», еще раз копируете свой путь и дописываете «\Scripts\». Готово! Нажимайте везде **Ок** и закрывайте лишние окошки. Закройте и откройте окно командной строки (*это важно*, переменные окружения «запоминаются» терминалом при запуске) и проверяйте заново корректность установки Python, как было предложено выше. Если все еще что-то идет не так – обратитесь к своему куратору.



Jupyter и другие.

Теперь установим некоторые библиотеки, которые точно понадобятся вам на курсе (замечу, что не вообще все, а только основные). Прежде всего, хорошей практикой считается установка библиотек в виртуальное окружение, а не глобально.

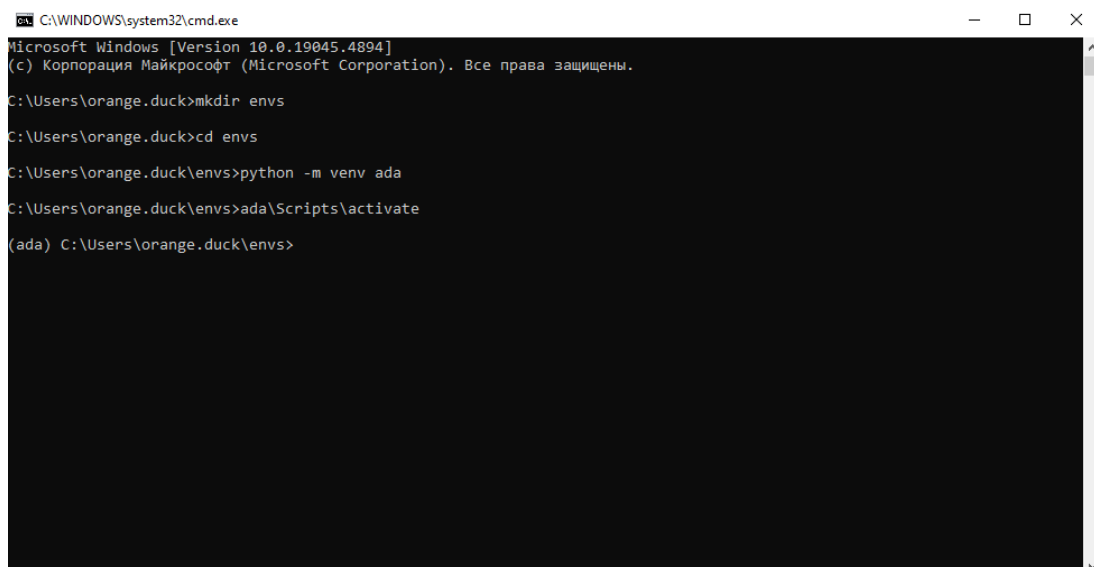
*Краткая справка, что это вообще значит. Если вы создаете виртуальное окружение Python и устанавливаете в него какие-либо библиотеки, то они доступны только изнутри этого виртуального окружения. Если же вы устанавливаете библиотеку глобально, то она доступна без активации конкретной среды. Минусы глобальной установки – при возникновении конфликта установленных библиотек вы получаете *dead kernel* на глобальном интерпретаторе (если вы не знаете, что это такое, то поверьте на слово, что это очень плохо). Кроме того, если один из ваших проектов привязан к конкретной версии библиотеки, то вы теряете возможность обновить ее в принципе. Поэтому лучше всего иметь «чистый» глобальный интерпретатор, а все библиотеки*

устанавливать в виртуальные окружения, создаваемые под конкретный проект.

Поэтому начнем установку с создания виртуального окружения. Сделать это можно с помощью команды **python -m venv** **<полный_путь_к_вашей_новой_среде>** .

*Краткая справка про полный путь. Пути бывают абсолютные и относительные. Если путь начинается с имени диска («C:\»), то он абсолютный. Иначе относительный. Если путь абсолютный, то с ним все понятно. А вот если относительный, то он строится от текущей директории, которая указана в приглашении командной строки. Так, в примере ниже текущая директория – это «C:\Users\orange.duck» (это, к слову, абсолютный путь) для первых двух команд, «C:\Users\orange.duck\envs» для третьей и четвертой (команда *cd* отвечает за перемещение между директориями и в качестве аргумента получает относительный (абсолютный тоже можно) путь – директорию «envs»). «ada» (от Applied Data Analysis) – это тоже относительный путь, просто состоящий из одного имени директории. Если поставить вместо него «C:\Users\orange.duck\envs\ada» – результат будет тем же.*

Удобно все виртуальные среды создавать в какой-нибудь одной директории вроде «envs» и называть их в соответствии с проектами, к которым они относятся. Однако это совершенно не обязательно и при желании вы можете раскидать их где угодно.



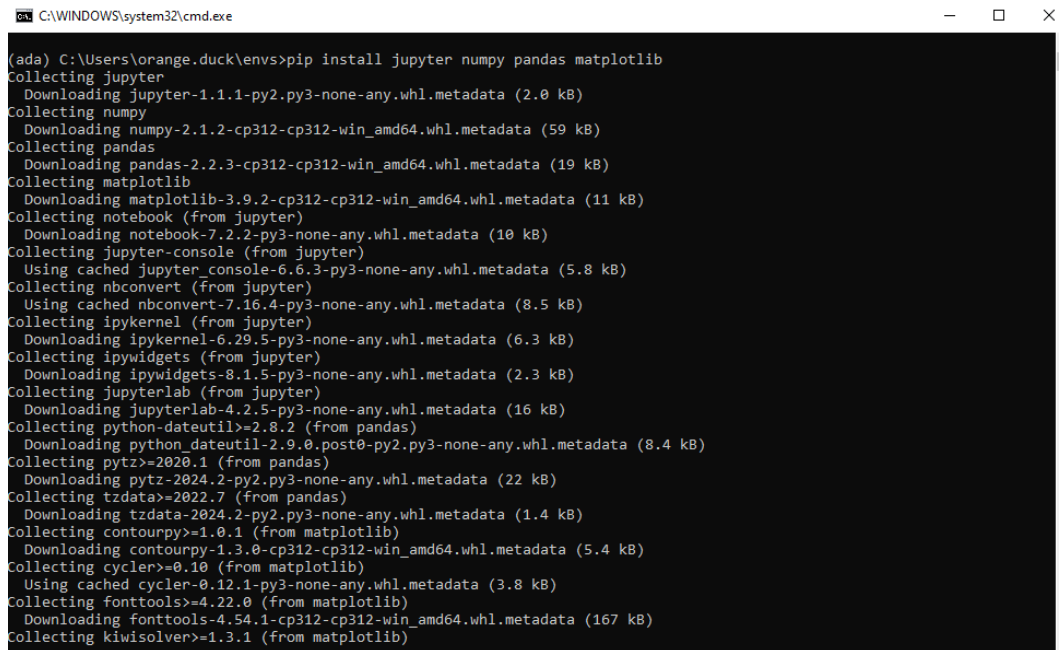
```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\orange.duck>mkdir envs
C:\Users\orange.duck>cd envs
C:\Users\orange.duck\envs>python -m venv ada
C:\Users\orange.duck\envs>ada\Scripts\activate
(ada) C:\Users\orange.duck\envs>
```

Наконец, чтобы «зайти» в виртуальную среду, нужно запустить ее скрипт активации – он лежит по адресу «<путь_к_вашей_виртуальной_среде>\Scripts\activate» и запускается просто

вбиванием этого пути в командную строку. Если вы все сделали правильно, то имя виртуальной среды появится в скобках перед следующим приглашением командной строки. Поздравляю, вы только что создали и активировали свою виртуальную среду!

Дальше вбейте вот эту команду, она установит необходимые библиотеки (это может быть немного долго) **`pip install jupyter numpy pandas matplotlib`**



```
C:\WINDOWS\system32\cmd.exe
(ada) C:\Users\orange.duck\envs>pip install jupyter numpy pandas matplotlib
Collecting jupyter
  Downloading jupyter-1.1.1-py2.py3-none-any.whl.metadata (2.0 kB)
Collecting numpy
  Downloading numpy-2.1.2-cp312-cp312-win_amd64.whl.metadata (59 kB)
Collecting pandas
  Downloading pandas-2.2.3-cp312-cp312-win_amd64.whl.metadata (19 kB)
Collecting matplotlib
  Downloading matplotlib-3.9.2-cp312-cp312-win_amd64.whl.metadata (11 kB)
Collecting notebook (from jupyter)
  Downloading notebook-7.2.2-py3-none-any.whl.metadata (10 kB)
Collecting jupyter-console (from jupyter)
  Using cached jupyter_console-6.6.3-py3-none-any.whl.metadata (5.8 kB)
Collecting nbconvert (from jupyter)
  Using cached nbconvert-7.16.4-py3-none-any.whl.metadata (8.5 kB)
Collecting ipykernel (from jupyter)
  Downloading ipykernel-6.29.5-py3-none-any.whl.metadata (6.3 kB)
Collecting ipywidgets (from jupyter)
  Downloading ipywidgets-8.1.5-py3-none-any.whl.metadata (2.3 kB)
Collecting jupyterlab (from jupyter)
  Downloading jupyterlab-4.2.5-py3-none-any.whl.metadata (16 kB)
Collecting python-dateutil>=2.8.2 (from pandas)
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl.metadata (8.4 kB)
Collecting pytz>=2020.1 (from pandas)
  Downloading pytz-2024.2-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.7 (from pandas)
  Downloading tzdata-2024.2-py2.py3-none-any.whl.metadata (1.4 kB)
Collecting contourpy>=1.0.1 (from matplotlib)
  Downloading contourpy-1.3.0-cp312-cp312-win_amd64.whl.metadata (5.4 kB)
Collecting cycler>=0.10 (from matplotlib)
  Using cached cycler-0.12.1-py3-none-any.whl.metadata (3.8 kB)
Collecting fonttools>=4.22.0 (from matplotlib)
  Downloading fonttools-4.54.1-cp312-cp312-win_amd64.whl.metadata (167 kB)
Collecting kiwisolver>=1.3.1 (from matplotlib)
```

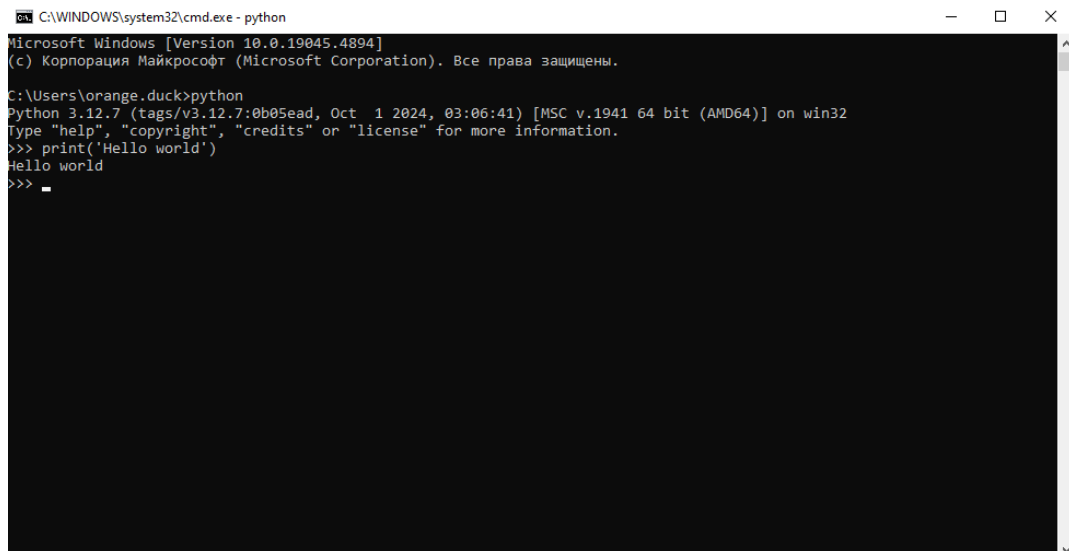
Редакторы кода и IDE.

Итак, вы установили Python и даже некоторые библиотеки, но как с этим всем работать? Опций на этот счет великое множество, и некоторые из них мы рассмотрим.

Во-первых, с чем вообще вам предстоит работать. Интерпретатор Python выполняет программы в файлах с расширением `.py`. Но помимо этого вам также предстоит работать с файлами с расширением `.ipynb` – «ноутбуками» (Jupyter Notebook). Работать с ними позволяет установленная выше библиотека `jupyter`, без нее выполнять код в этих файлах не выйдет. Однако если программы `.py` – это просто исходный код Python, который можно набрать в любом текстовом редакторе (можно и в блокноте), то файлы `.ipynb` в «сыром» виде читать достаточно проблематично. Поэтому рассмотрим инструменты для обоих типов.

0. Командная строка (Python – да, Jupyter – нет)

В целом, код на Python можно выполнять прямо в командной строке – для этого достаточно просто запустить интерпретатор Python, и все, можно писать код, который тут же будет выполняться. Этот способ едва ли подходит для написания программ, однако в документации многих библиотек встречаются примеры, написанные именно таким образом, поэтому о его существовании полезно знать (P. S. Чтобы выйти, напишите `exit()`).

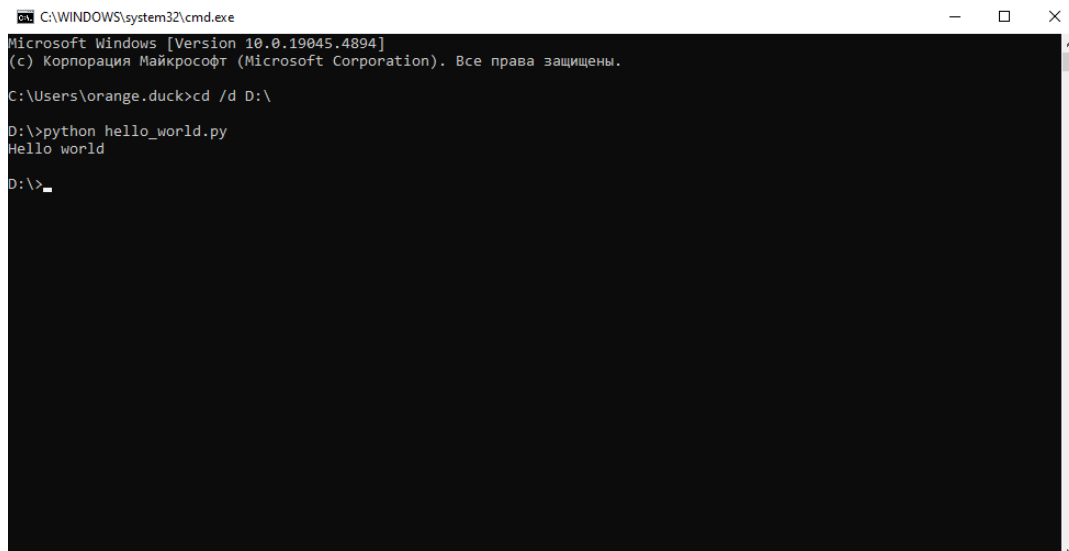
A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe - python". The window shows the output of running the 'python' command. It displays the Python version (3.12.7), build information, and the copyright notice. Then, the user enters a Python shell prompt '>>>' and runs the command 'print('Hello world')', which outputs 'Hello world'. The prompt '>>>' is shown again on the next line.

```
C:\WINDOWS\system32\cmd.exe - python
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\orange.duck>python
Python 3.12.7 (tags/v3.12.7:0b05ead, Oct  1 2024, 03:06:41) [MSC v.1941 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello world')
Hello world
>>> _
```

1. Любой текстовый редактор (Python – да, Jupyter – скорее нет)

Подойдет хоть блокнот – можно написать текст программы, сохранить в файле с расширением `.py` и затем запустить из командной строки командой **python <путь_к_вашей_программе>**

A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The user has navigated to the directory 'D:\' using the 'cd /d D:\' command. Then, they run 'python hello_world.py', which outputs 'Hello world'. The prompt 'D:\>' is shown at the bottom.

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\orange.duck>cd /d D:\
D:\>python hello_world.py
Hello world
D:\>_
```

2. Jupyter в браузере (Python – только редактировать, Jupyter – да)

Библиотека jupyter позволяет запускать свой интерфейс прямо в браузере. Интерфейс достаточно функциональный – в нем можно редактировать и `.py`, и `.ipynb` файлы, да и любые другие, просматривать файлы (изображения, например). Но исполнять –

только .ipynb. Запустить этот интерфейс можно командой (не забудьте активировать виртуальную среду) **jupyter notebook**

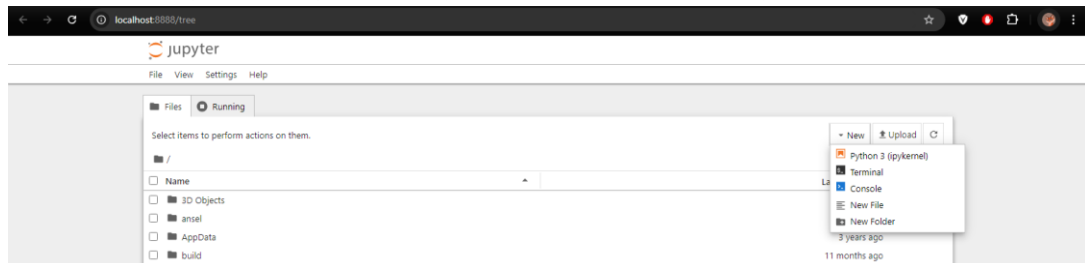
```
npm prefix
Microsoft Windows [Version 10.0.19045.4894]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\orange.duck>envs\ada\Scripts\activate

(ada) C:\Users\orange.duck>jupyter notebook
[I 2024-10-07 18:50:12.254 ServerApp] jupyter_lsp | extension was successfully linked.
[I 2024-10-07 18:50:12.262 ServerApp] jupyter_server_terminals | extension was successfully linked.
[I 2024-10-07 18:50:12.272 ServerApp] jupyterlab | extension was successfully linked.
[I 2024-10-07 18:50:12.281 ServerApp] notebook | extension was successfully linked.
[I 2024-10-07 18:50:12.304 ServerApp] Writing Jupyter server cookie secret to C:\Users\orange.duck\AppData\Roaming\jupyter\runtime\jupyter_cookie_secret
[I 2024-10-07 18:50:13.194 ServerApp] notebook_shim | extension was successfully linked.
[I 2024-10-07 18:50:13.241 ServerApp] notebook_shim | extension was successfully loaded.
[I 2024-10-07 18:50:13.244 ServerApp] jupyter_lsp | extension was successfully loaded.
[I 2024-10-07 18:50:13.245 ServerApp] jupyter_server_terminals | extension was successfully loaded.
[I 2024-10-07 18:50:13.250 LabApp] JupyterLab extension loaded from C:\Users\orange.duck\envs\ada\Lib\site-packages\jupyterlab
[I 2024-10-07 18:50:13.250 LabApp] JupyterLab application directory is C:\Users\orange.duck\envs\ada\share\jupyter\lab
[I 2024-10-07 18:50:13.251 LabApp] Extension Manager is 'pypi'.
[I 2024-10-07 18:50:13.715 ServerApp] jupyterlab | extension was successfully loaded.
[I 2024-10-07 18:50:13.722 ServerApp] notebook | extension was successfully loaded.
[I 2024-10-07 18:50:13.724 ServerApp] Serving notebooks from local directory: C:\Users\orange.duck
[I 2024-10-07 18:50:13.724 ServerApp] Jupyter Server 2.14.2 is running at:
[I 2024-10-07 18:50:13.724 ServerApp] http://localhost:8888/tree?token=7dcd49055baaa5186b2b3693a249925fdf865351d0bc263d
[I 2024-10-07 18:50:13.724 ServerApp] http://127.0.0.1:8888/tree?token=7dcd49055baaa5186b2b3693a249925fdf865351d0bc263d
[I 2024-10-07 18:50:13.724 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 2024-10-07 18:50:13.808 ServerApp]

To access the server, open this file in a browser:
file:///C:/Users/orange.duck/AppData/Roaming/jupyter/runtime/jpserver-17744-open.html
Or copy and paste one of these URLs:
http://localhost:8888/tree?token=7dcd49055baaa5186b2b3693a249925fdf865351d0bc263d
http://127.0.0.1:8888/tree?token=7dcd49055baaa5186b2b3693a249925fdf865351d0bc263d
[I 2024-10-07 18:50:21.771 ServerApp] Skipped non-installed server(s): bash-language-server, dockerfile-language-server-nodejs, javascript-typescript-langserver, jedi-language-server, julia-language-server, pyright, python-language-server, python-lsp-server, r-language-server, sql-language-server, texlab, typescript-language-server, unified-language-server, vscode-css-language-server-bin, vscode-html-language-server-bin, vscode-json-language-server-bin, yaml-language-server
```

Интерфейс сам откроется в браузере (если нет, откройте браузер и вбейте туда то, что предлагает вам юпитер после строки **To access the server, open this file in a browser:**). Здесь вы можете открывать, просматривать, редактировать и создавать файлы.



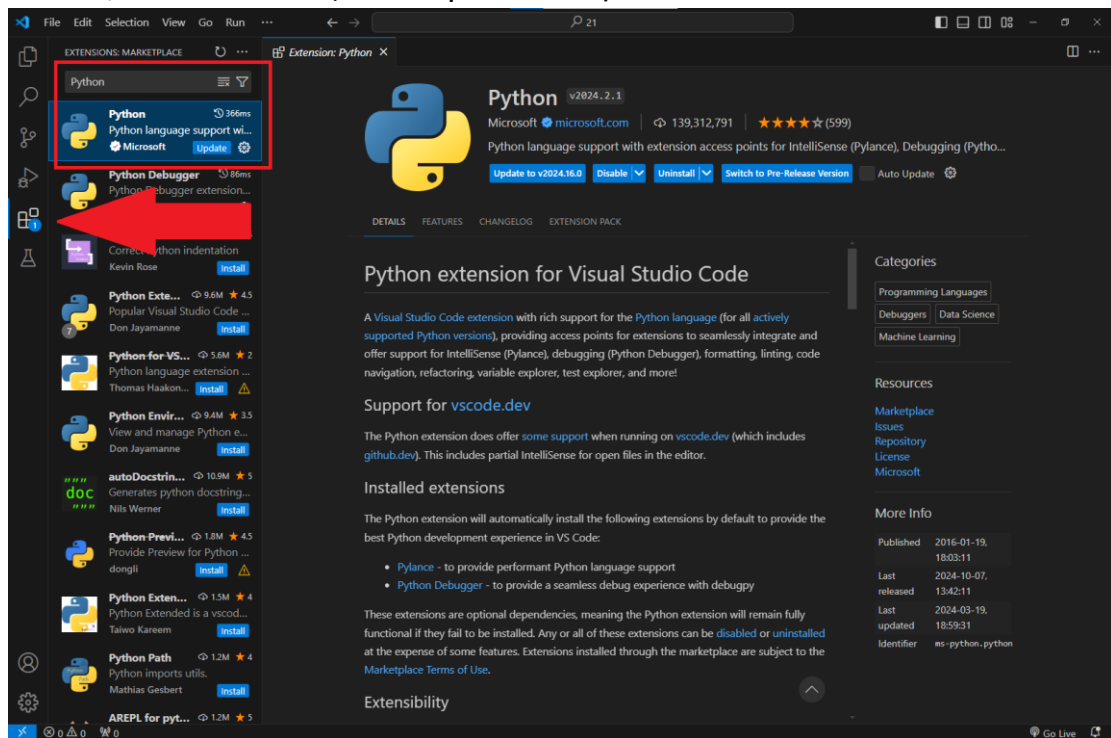
3. Google Colab (Python – скорее нет, Jupyter – да)

То, что вы видели на занятиях (вы же были на первом занятии, правда?). Интерфейс похож на Jupyter, но с некоторыми дополнительными плюсами. Во-первых, вам вообще не надо ничего устанавливать (в том числе большинство библиотек, они уже установлены). Во-вторых, вы легко можете редактировать один и тот же файл с разных устройств и даже совместно с другими людьми. В-третьих, все вычисления выполняются на серверах Google, в том числе Colab выделяет вам GPU time, поэтому он хорошо подходит для дорогих вычислительных задач. Для доступа к Colab вам нужен только гугл-аккаунт и браузер, доступ по ссылке

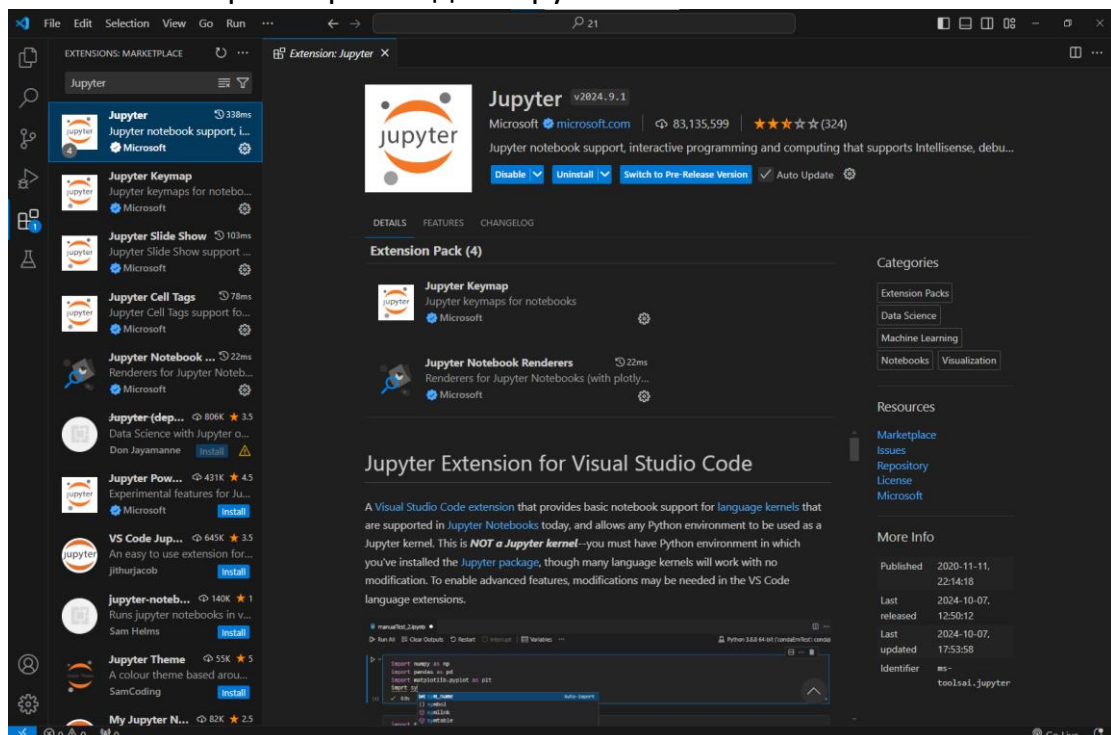
<https://colab.research.google.com/> .

4. VS Code (Python – да, Jupyter – тоже да)

Visual Studio Code – это редактор кода, не полноценная IDE. Тем не менее он легок в использовании и в нем существуют расширения для работы практически с любым расширением файлов. Установить его можно отсюда – <https://code.visualstudio.com/>. После установки самой программы зайдите в расширения и установить расширение для Python (скорее всего вам сразу предложат его установить, если нет, оно легко ищется через поиск).



И такое же расширение для Jupyter:



После этого вы сможете работать с файлами обоих типов. Если вам понадобится работать с другими типами файлов (например, json, yaml, css), для них тоже существуют различные расширения (однако открывать и редактировать большинство файлов вы можете и без них, они призваны лишь упростить вам жизнь).

5. PyCharm (Python – да, Jupyter – скорее нет)

PyCharm это полноценная IDE, он отлично подходит для больших проектов, написанных на Python, однако работу с .ipynb файлами он поддерживает только в Pro версии, приобрести которую у граждан РФ сейчас возможности нет. Тем не менее, он обладает рядом очень полезных особенностей (например, анализирует ваш код, указывает на потенциально проблемных места и предлагает quick-fixes). Скачать Community версию можно здесь:

<https://www.jetbrains.com/pycharm/download/?section=windows>

Linux

В качестве примера установки будет рассмотрен дистрибутив Ubuntu 18.04. Если у вас другой дистрибутив со своим пакетным менеджером – замените команду apt на свой пакетный менеджер (например, pacman для Arch).

Базовый Python.

*Note: для большинства дистрибутивов Linux есть разница между Python 2 и Python 3. Если вы напишете просто **python** – будет использован Python 2, поэтому вам нужно пользоваться именно **python3**. Вполне вероятно, что у вас уже установлены оба, так как некоторые программы могут их задействовать.*

Если же у вас **python3** не установлен, сделать это можно командой **sudo apt install python3**. Если в вашем дистрибутиве нет опции **sudo**, вам нужно устанавливать его под рутом (впрочем, скорее всего, вы и так это знаете).

```

orangeduck@orangeduck:~$ python --version
Python 2.7.17
orangeduck@orangeduck:~$ python3 --version
Python 3.6.9
orangeduck@orangeduck:~$ sudo apt install python3
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Уже установлен пакет python3 самой новой версии (3.6.7-1~18.04).
python3 помечен как установленный вручную.
Следующие пакеты устанавливались автоматически и больше не требуются:
  efibootmgr gir1.2-geocodeglib-1.0 gir1.2-keybinder-3.0 libfwupd1
  libkeybinder-3.0-0 libllvm9 linux-hwe-5.4-headers-5.4.0-53
  linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-72 python-cairo
  python-dbus python-gi python-gi-cairo python-psutil python3-click
  python3-colorama ubuntu-web-launchers
Для их удаления используйте «sudo apt autoremove».
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов,
и 68 пакетов не обновлено.
orangeduck@orangeduck:~$

```

Jupyter и другие.

Виртуальное окружение создается той же командой, что и в Windows, с поправкой на тройку: **python3 -m venv** **<полный_путь_к_вашей_новой_среде>** . Казалось бы... Но иногда нужно доустановить отдельный пакет **python3-venv** (не забудьте про **sudo**, хоть установщик про него и не говорит).

```

orangeduck@orangeduck:~$ mkdir envs
orangeduck@orangeduck:~$ cd envs
orangeduck@orangeduck:~/envs$ python3 -m venv ada
The virtual environment was not created successfully because ensurepip is not
available. On Debian/Ubuntu systems, you need to install the python3-venv
package using the following command.

    apt-get install python3-venv

You may need to use sudo with that command. After installing the python3-venv
package, recreate your virtual environment.

Failing command: ['/home/orangeduck/envs/ada/bin/python3', '-Im', 'ensurepip', '--upgrade', '--default-pip']

orangeduck@orangeduck:~/envs$ apt-get install python3-venv
E: Не удалось открыть файл блокировки /var/lib/dpkg/lock-frontent - open (13: Отказано в доступе)
E: Невозможно получить блокировку внешнего интерфейса dpkg (/var/lib/dpkg/lock-frontent); у вас есть права суперпользователя?
orangeduck@orangeduck:~/envs$ sudo apt-get install python3-venv
[sudo] пароль для orangeduck:
Чтение списков пакетов... Готово
Построение дерева зависимостей

```

Наконец, активировать вашу виртуальную среду позволяет команда **source** вида **source <полный_путь_к_вашей_новой_среде>/bin/activate** .


```

Обновлено 0 пакетов, установлено 2 новых пакетов, для удаления отмечено 0 пакетов, и 68 пакетов не обновлено.
Необходимо скачать 7 392 В архивов.
После данной операции объем занятого дискового пространства возрастет на 44,0 kB.
Хотите продолжить? [Д/н] у
Пол:1 http://ru.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 python3.6-venv amd64 3.6.9-1~18.04ubuntu1.12 [6 184 B]
Пол:2 http://ru.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 python3-venv amd64 3.6.7-1~18.04 [1 208 B]
Получено 7 392 В за 0с (108 kB/s)
Выбор ранее не выбранного пакета python3.6-venv.
(Чтение базы данных ... на данный момент установлено 287560 файлов и каталогов.)
Подготовка к распаковке .../python3.6-venv_3.6.9-1~18.04ubuntu1.12_amd64.deb ...
Распаковывается python3.6-venv (3.6.9-1~18.04ubuntu1.12) ...
Выбор ранее не выбранного пакета python3-venv.
Подготовка к распаковке .../python3-venv_3.6.7-1~18.04_amd64.deb ...
Распаковывается python3-venv (3.6.7-1~18.04) ...
Настраивается пакет python3.6-venv (3.6.9-1~18.04ubuntu1.12) ...
Настраивается пакет python3-venv (3.6.7-1~18.04) ...
Обрабатываются триггеры для man-db (2.8.3-2ubuntu0.1) ...
orangeduck@orangeduck:~/envs$ python3 -m venv ada
orangeduck@orangeduck:~/envs$ source ada/bin/activate
(ada) orangeduck@orangeduck:~/envs$

```

С `pip` то же самое, что и с `python` – вам нужно использовать `pip3`. Необходимые библиотеки можно установить командой **`pip3 install jupyter numpy pandas matplotlib`**. Так же, если `python3` уже был у вас установлен, вероятно, что не был установлен `pip3`. Его можно установить командой (вероятно, ваш установщик сам вам ее предложит) **`sudo apt install python3-pip`**. После установки `pip3` запустите команду на установку библиотек.

```

orangeduck@orangeduck:~$ pip3 install jupyter numpy pandas matplotlib
Command 'pip3' not found, but can be installed with:
sudo apt install python3-pip
orangeduck@orangeduck:~$ sudo apt install python3-pip
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Следующие пакеты устанавливались автоматически и больше не требуются:
efibootmgr gir1.2-geocodeglib-1.0 gir1.2-keybinder-3.0 libfwup1
libkeybinder-3.0-0 libllvm9 linux-hwe-5.4-headers-5.4.0-53
linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-72 python-cairo
python-dbus python-gi python-gi-cairo python-psutil python3-click
python3-colorama ubuntu-web-launchers
Для их удаления используйте «sudo apt autoremove».
Будут установлены следующие дополнительные пакеты:
dh-python libpython3-dev libpython3.6-dev python-pip-whl python3-dev
python3-setuptools python3-wheel python3.6-dev

```

Редакторы кода и IDE.

Здесь опции у вас те же, что и для Windows – все они доступны и для большинства Linux-дистрибутивов (кроме блокнота, само собой – замените его на любой редактор текстов вроде `vim` или `nano`).