



**МИЭТ**

Национальный исследовательский университет «МИЭТ»

Кафедра ПКИМС

# Компьютерные технологии в научных исследованиях

Семинар №7

**Язык Python. Основные сведения**

# Запуск кода в Python

```
C:\windows\system32\cmd.exe - python
Microsoft Windows [Version 10.0.19044.2486]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Дмитрий Булах>python --version
Python 3.10.2

C:\Users\Дмитрий Булах>python
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

```
C:\Users\Дмитрий Булах\code.py - Notepad++
Файл  Правка  Поиск  Вид  Кодировки  Синтаксисы  Опции
Инструменты  Макросы  Запуск  Плагины  Вкладки  ?  +  ▾  X

code.py x
1  x = 42
2  print(f'x = {x}')
```

```
C:\windows\system32\cmd.exe - python
C:\Users\Дмитрий Булах>python
Python 3.10.2 (tags/v3.10.2:a58ebcc, Jan 17 2022, 14:12:15) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> x = 42
>>> print(f'x = {x}')
x = 42
>>> _
```

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>python ./code.py
x = 42
C:\Users\Дмитрий Булах>_
```

# Написание кода в Python: Visual Studio Code



Visual Studio Code

```
main.py x
main.py > ...
1 #!/usr/bin/python
2
3 from graphviz import Digraph
4
5 graph = Digraph("Test", format='png')
6
7 graph.edge('a', 'b')
8
9 print(graph)
10 graph.render('example.dot', view=True)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Python

Windows PowerShell  
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.  
Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/powershell)  
PS C:\Users\Дмитрий Булах\source\repos\_vscode\graphviz\_python> & D:/Python/Python310/python.exe "c:/Users/Дмитрий Булах/source/repos\_vscode/graphviz\_python/main.py"  
digraph Test {  
 a -> b  
}

```
main.py x
main.py > ...
1 #!/usr/bin/python
2
3 from graphviz import Digraph
4
5 graph = Digraph("Graph example", format='png')
6
7 graph.edge('a', 'b')
8
9 print(graph)
10
11 graph.render('example.dot', view=False)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Python

raise ExecutableNotFoundError from e  
graphviz.backends.ExecutableNotFoundError: failed to execute PosixPath('dot'), make sure the Graphviz executables are on your systems' PATH  
digraph "Graph example" {  
 a -> b  
}

```
main.py x
main.py > ...
1 #!/usr/bin/python
2
3 from graphviz import Digraph
4
5 graph = Digraph("Test graph", format='png')
6
7 graph.edge('A', 'B')
8
9 print(graph)
10 graph.render('example.dot', view=True)
11
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Python

/usr/bin/python /home/topgun/Programming/vscode/python\_graphviz/main.py  
[topgun@manjaro python\_graphviz]\$ /usr/bin/python /home/topgun/Programming/vscode/python\_graphviz/main.py  
digraph "Test graph" {  
 A -> B  
}

[topgun@manjaro python\_graphviz]\$ kf.service.services: KApplicationTrader: mimeType "x-scheme-handler/file" not found  
org.kde.kdegraphics.gwenview.lib: Unresolved mimeType "image/x-mng"  
org.kde.kdegraphics.gwenview.lib: Unresolved raw mimeType "image/x-samsung-srw"

# Package Installer for Python (pip)

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>pip

Usage:
  pip <command> [options]

Commands:
  install          Install packages.
  download         Download packages.
  uninstall       Uninstall packages.
  freeze          Output installed packages in requirements format.
  inspect         Inspect the python environment.
  list            List installed packages.
  show            Show information about installed packages.
  check           Verify installed packages have compatible dependencies.
  config          Manage local and global configuration.
  search          Search PyPI for packages.
  cache           Inspect and manage pip's wheel cache.
  index           Inspect information available from package indexes.
  wheel           Build wheels from your requirements.
  hash           Compute hashes of package archives.
  completion      A helper command used for command completion.
  debug           Show information useful for debugging.
  help            Show help for commands.

General Options:
  -h, --help      Show help.
  --debug         Let unhandled exceptions propagate outside the main subroutine, instead of logging them to stderr.
  --isolated      Run pip in an isolated mode, ignoring environment variables and user configuration.
  --require-virtualenv
                  Allow pip to only run in a virtual environment; exit with an error otherwise.
```

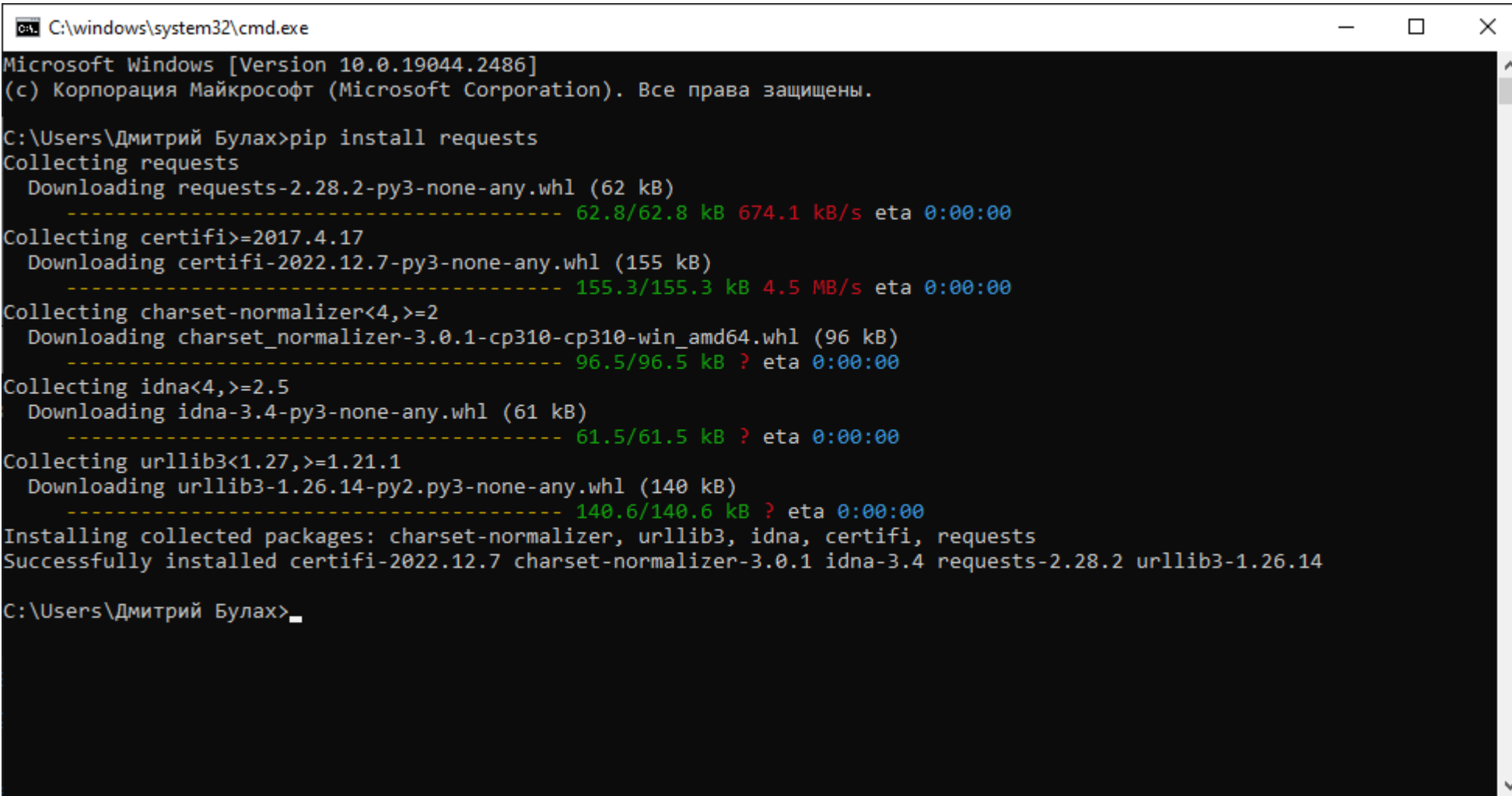
# pip list: получение списка пакетов

```
C:\Users\Дмитрий Булах>pip list
```

```
Package            Version
-----
bashplotlib        0.6.5
certifi            2022.12.7
charset-normalizer 3.0.1
click              8.1.3
colorama           0.4.6
cyclor             0.11.0
fonttools          4.29.1
graphviz           0.19.1
idna               3.4
imgui              1.4.1
kiwisolver         1.3.2
matplotlib         3.5.1
mypy               0.931
mypy-extensions    0.4.3
numpy              1.22.2
opencv-python      4.5.5.64
packaging          21.3
Pillow            9.0.1
pip               22.3.1
progress          1.6
PTable            0.9.2
PyOpenGL          3.1.6
pyparsing         3.0.7
PyQt5             5.15.7
PyQt5-Qt5         5.15.2
PyQt5-sip         12.11.0
PySDL2            0.9.11
pysdl2-dll        2.0.20
PySimpleGUI        4.60.4
python-dateutil   2.8.2
requests          2.28.2
setuptools         58.1.0
six               1.16.0
tk                0.1.0
tomli             2.0.1
typing_extensions 4.1.1
urllib3           1.26.14
```

# pip install: установка пакетов

```
>pip install <пакет1> [<пакет2>, ...]
```



```
C:\windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19044.2486]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Дмитрий Булах>pip install requests
Collecting requests
  Downloading requests-2.28.2-py3-none-any.whl (62 kB)
----- 62.8/62.8 kB 674.1 kB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.12.7-py3-none-any.whl (155 kB)
----- 155.3/155.3 kB 4.5 MB/s eta 0:00:00
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.0.1-cp310-cp310-win_amd64.whl (96 kB)
----- 96.5/96.5 kB ? eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.4-py3-none-any.whl (61 kB)
----- 61.5/61.5 kB ? eta 0:00:00
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.14-py2.py3-none-any.whl (140 kB)
----- 140.6/140.6 kB ? eta 0:00:00
Installing collected packages: charset-normalizer, urllib3, idna, certifi, requests
Successfully installed certifi-2022.12.7 charset-normalizer-3.0.1 idna-3.4 requests-2.28.2 urllib3-1.26.14

C:\Users\Дмитрий Булах>_
```

# Откуда берётся информация о пакетах?

URL: <https://pypi.org/>

The image shows a browser window with three tabs. The active tab is titled "PySimpleGUI · PyPI" and shows the project page for PySimpleGUI on the PyPI website. The URL in the address bar is [pypi.org/project/PySimpleGUI/](https://pypi.org/project/PySimpleGUI/). The page content includes:

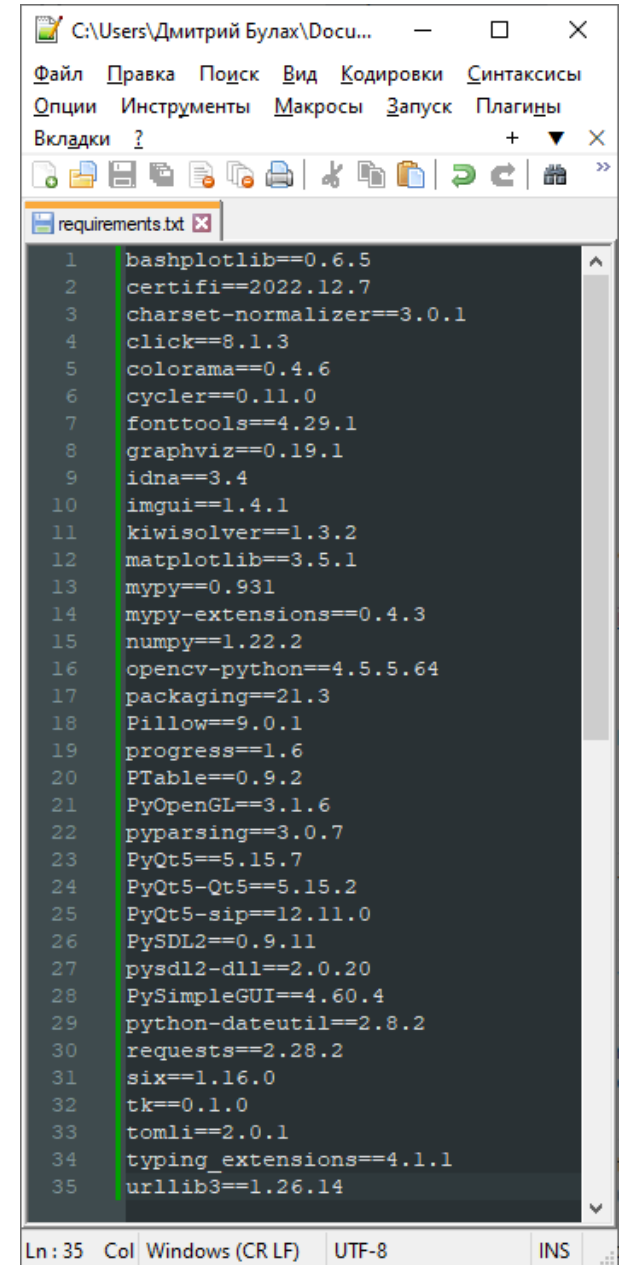
- Навигация** (Navigation):
  - Описание проекта (Project description)
  - История выпусков (Release history)
  - Загрузка файлов (Download files)
- Ссылки проекта** (Project links):
  - Homepage
- Статистика** (Statistics):
  - Статистика GitHub (GitHub statistics):
    - Звезд: 11232 (Stars)
    - Форков: 1669 (Forks)
    - Открытых замечаний/запросов на вытягивание: 685 (Open issues/pull requests)
  - Смотрите статистику этого проекта на [Libraries.io](#) или в [нашем общедоступном наборе данных на Google BigQuery](#)
- Описание проекта** (Project description):
  - PySimpleGUI™ logo
  - Python GUIs for Humans
- Advertisement**:
  - Want to master PySimpleGUI? Sign up to the official course on [udemy](#)
  - APPLY COUPON FOR DISCOUNT: 4FD91A459D56B1029FF8
  - [click here to visit course page](#)
- Text at the bottom**:

Transforms the tkinter, Qt, WxPython, and Remi (browser-based) GUI frameworks into a simpler interface. The window definition is simplified by using Python core data types understood by beginners (lists and dictionaries).

# pip install: установка заданного перечня пакетов

C:\Users\Дмитрий Булах>pip freeze > requirements.txt

C:\Users\Дмитрий Булах>pip install -r requirements.txt

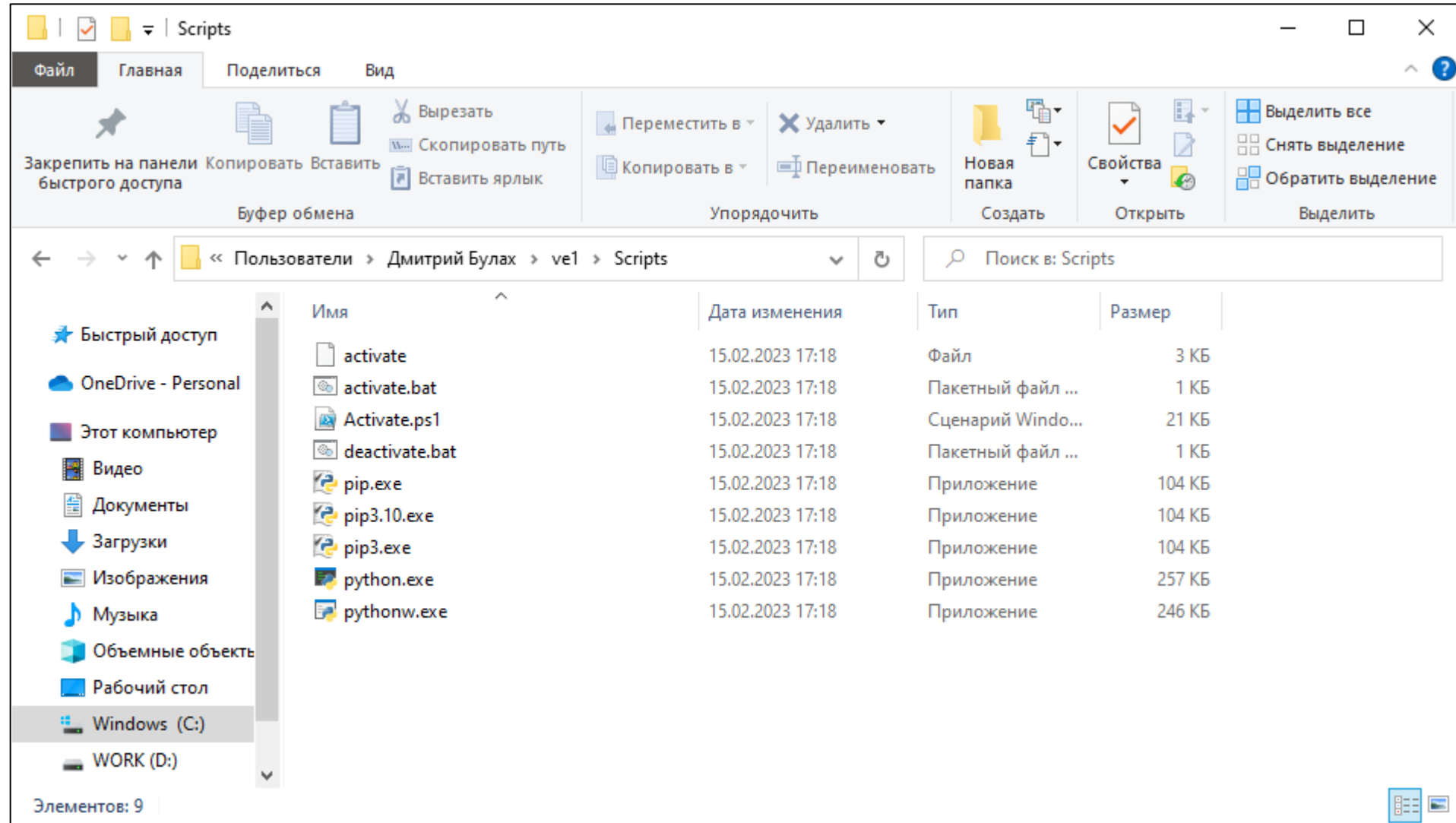


```
1 bashplotlib==0.6.5
2 certifi==2022.12.7
3 charset-normalizer==3.0.1
4 click==8.1.3
5 colorama==0.4.6
6 cyclер==0.11.0
7 fonttools==4.29.1
8 graphviz==0.19.1
9 idna==3.4
10 imgui==1.4.1
11 kiwisolver==1.3.2
12 matplotlib==3.5.1
13 mypy==0.931
14 mypy-extensions==0.4.3
15 numpy==1.22.2
16 opencv-python==4.5.5.64
17 packaging==21.3
18 Pillow==9.0.1
19 progress==1.6
20 PTable==0.9.2
21 PyOpenGL==3.1.6
22 pyparsing==3.0.7
23 PyQt5==5.15.7
24 PyQt5-Qt5==5.15.2
25 PyQt5-sip==12.11.0
26 PySDL2==0.9.11
27 pysdl2-dll==2.0.20
28 PySimpleGUI==4.60.4
29 python-dateutil==2.8.2
30 requests==2.28.2
31 six==1.16.0
32 tk==0.1.0
33 toml==2.0.1
34 typing_extensions==4.1.1
35 urllib3==1.26.14
```



# Виртуальное окружение в Python: создание

```
C:\Users\Дмитрий Булах>python -m venv ve1
```



# Виртуальное окружение в Python: активация

```
C:\Users\Дмитрий Булах>ve1\Scripts\activate.bat
```

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>python -m venv ve1
C:\Users\Дмитрий Булах>ve1\Scripts\activate.bat
(ve1) C:\Users\Дмитрий Булах>
```

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>python -m venv ve1
C:\Users\Дмитрий Булах>ve1\Scripts\activate.bat
(ve1) C:\Users\Дмитрий Булах>pip list
Package      Version
-----
pip          21.2.4
setuptools   58.1.0
WARNING: You are using pip version 21.2.4; however, version 23.0 is available.
You should consider upgrading via the 'C:\Users\Дмитрий Булах\ve1\Scripts\python.exe -m pip install --upgrade pip' command.
(ve1) C:\Users\Дмитрий Булах>
```

# Виртуальное окружение в Python: работа с venv

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>ve1\Scripts\activate.bat

(ve1) C:\Users\Дмитрий Булах>pip list
Package      Version
-----
pip          21.2.4
setuptools  58.1.0
WARNING: You are using pip version 21.2.4; however, version 23.0 is available.
You should consider upgrading via the 'C:\Users\Дмитрий Булах\ve1\Scripts\python.exe -m pip install --upgrade pip' command.

(ve1) C:\Users\Дмитрий Булах>pip install wxpython
Collecting wxpython
  Downloading wxPython-4.2.0-cp310-cp310-win_amd64.whl (18.0 MB)
    |#####| 18.0 MB 2.2 MB/s
Collecting pillow
  Downloading Pillow-9.4.0-cp310-cp310-win_amd64.whl (2.5 MB)
    |#####| 2.5 MB ...
Collecting numpy
  Downloading numpy-1.24.2-cp310-cp310-win_amd64.whl (14.8 MB)
    |#####| 14.8 MB ...
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: six, pillow, numpy, wxpython
Successfully installed numpy-1.24.2 pillow-9.4.0 six-1.16.0 wxpython-4.2.0
WARNING: You are using pip version 21.2.4; however, version 23.0 is available.
You should consider upgrading via the 'C:\Users\Дмитрий Булах\ve1\Scripts\python.exe -m pip install --upgrade pip' command.

(ve1) C:\Users\Дмитрий Булах>
```

```
C:\windows\system32\cmd.exe
  Downloading wxPython-4.2.0-cp310-cp310-win_amd64.whl (18.0 MB)
    |#####| 18.0 MB 2.2 MB/s
Collecting pillow
  Downloading Pillow-9.4.0-cp310-cp310-win_amd64.whl (2.5 MB)
    |#####| 2.5 MB ...
Collecting numpy
  Downloading numpy-1.24.2-cp310-cp310-win_amd64.whl (14.8 MB)
    |#####| 14.8 MB ...
Collecting six
  Using cached six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: six, pillow, numpy, wxpython
Successfully installed numpy-1.24.2 pillow-9.4.0 six-1.16.0 wxpython-4.2.0
WARNING: You are using pip version 21.2.4; however, version 23.0 is available.
You should consider upgrading via the 'C:\Users\Дмитрий Булах\ve1\Scripts\python.exe -m pip install --upgrade pip' command.

(ve1) C:\Users\Дмитрий Булах>pip list
```

```
C:\windows\system32\cmd.exe
C:\Users\Дмитрий Булах>pip list
Package      Version
-----
bashplotlib  0.6.5
certifi      2022.12.7
charset-normalizer 3.0.1
click        8.1.3
colorama     0.4.6
cyclor       0.11.0
fonttools   4.29.1
graphviz     0.19.1
idna         3.4
imgui        1.4.1
kiwisolver   1.3.2
matplotlib  3.5.1
mypy        0.931
mypy-extensions 0.4.3
numpy        1.22.2
opencv-python 4.5.5.64
packaging    21.3
Pillow       9.0.1
pip          23.0
progress     1.6
PTable       0.9.2
PyOpenGL     3.1.6
pyparsing   3.0.7
PyQt5        5.15.7
PyQt5-Qt5    5.15.2
PyQt5-sip    12.11.0
PySDL2       0.9.11
```

# Синтаксис языка Python: вывод данных

```
print(*items, sep=' ', end='\n', file=sys.stdout, flush=False)
```

```
x = 4
```

```
y = 2
```

```
print('x = {}, y = {}'.format(x, y))
```

```
print(f'x = {x}, y = {y}')
```

```
print(f'{{{x}}} = {x}, {{{y}}} = {y}')
```

# Python2 и Python3

```
print "Hello, world!"
```

```
print 'Hello, world!'
```

```
s1 = 'A'
```

```
print "Value: %s" % s1
```

```
s1 = 'A'
```

```
s2 = 'B'
```

```
print "Values are: %(b)s and %(a)s" % {'a' : s1, 'b' : s2}
```

```
x = 4
```

```
y = 2
```

```
print('x = {}, y = {}'.format(x, y))
```

```
print(f'x = {x}, y = {y}')
```

```
print("Values: {1} {0}".format(x, y))
```

---

Строки : ASCII

Строки : Unicode

---

Целочисленное деление:

7/2 => 3

Целочисленное деление:

7/2 => 3.5

# Типы данных в Python

В Python есть несколько стандартных типов данных:

- числа (Numbers)
  - целые (Integers)
  - вещественные (Real)
  - комплексные (Complex)
- строки (Strings)
- списки (Lists)
- множества (Sets)
- кортежи (Tuples)
- словари (Dictionaries)
- логический тип данных (Boolean)

Эти типы данных можно, в свою очередь,

классифицировать по нескольким признакам:

- изменяемые (списки, словари и множества)
- неизменяемые (числа, строки и кортежи)
- упорядоченные (списки, кортежи, строки и словари)
- неупорядоченные (множества)

```
x = 42
print(f' value: {x} of type {type(x)}')
```

```
value: 42 of type <class 'int'>
```

# Операции с целыми числами

```
#!/usr/bin/python
```

```
x = 5
```

```
y = 3
```

```
print(5 / 3)
```

```
print(5 // 3)
```

```
print(5 % 3)
```

```
print(5 ** 3)
```

```
1.6666666666666667
```

```
1
```

```
2
```

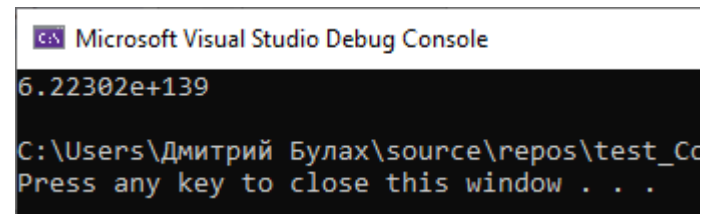
```
125
```

Операция	Результат
$-x$	Смена знака
$x + y$	Сумма двух чисел
$x - y$	Разность двух чисел
$x * y$	Произведение двух чисел
$x / y$	Частное чисел
$x // y$	Целая часть от деления
$x \% y$	Остаток от деления
$x ** y$	Возведение $x$ в степень $y$ , $x^y$

# Длинная арифметика в Python

```
#include <iostream>
#include <cmath>

int main() {
    int x = 5;
    int y = 200;
    std::cout << pow(x, y) << std::endl;
    return EXIT_SUCCESS;
}
```



```
Microsoft Visual Studio Debug Console
6.22302e+139
C:\Users\Дмитрий Булах\source\repos\test_Cpp\test_Cpp.exe
Press any key to close this window . . .
```

```
#!/usr/bin/python
```

```
x = 5
y = 200
```

```
print(x ** y)
```

```
6223015277861141707144064053780124240590252168721167133101116614
7896988340353834411839448231257136169569665895551224821247160434
722900390625
```



# Работа с комплексными числами

```
#!/usr/bin/python
```

```
x = complex(4, 2)
```

```
y = 2 + 4j
```

```
print(x)
```

```
print(y)
```

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

```
print(type(x))
```

```
print(type(y))
```

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

```
PS C:\Users\...\numpy> & D:/Python/python.exe "c:/Users/.../numpy/main.py"
```

```
(4+2j)
```

```
(2+4j)
```

```
(6+6j)
```

```
<class 'complex'>
```

```
<class 'complex'>
```

```
<class 'complex'>
```

```
PS C:\Users\...\numpy>
```

# Списки (Lists)

```
lst = [1, 2, 3, 4, 5]
print(lst)
```

```
[1, 2, 3, 4, 5]
```

```
print(len(lst))
```

```
5
```

```
lst = list('ПКИМС')
print(lst)
```

```
['П', 'К', 'И', 'М', 'С']
```

```
lst = [1, 2, 3, 'ПКИМС', 4]
print(lst)
```

```
[1, 2, 3, 'ПКИМС', 4]
```

```
print(type(lst))
```

```
<class 'list'>
```

```
lst = []
lst.append(1)
lst.append(2)
lst.append(3)
```

```
lst.remove(1)
```

```
lst.reverse()
```

```
lst.count(1)
```

```
lst.clear()
```

# Множества (Sets) (1)

```
s = {1, 2, 3, 4, 5}
print(s)
```

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

```
print(len(s))
```

```
5
```

```
s = {1, 2, 1, 2, 3}
print(s)
```

```
{1, 2, 3}
```

```
l = [1, 2, 1, 2, 3]
print(l)
```

```
s = set(l)
print(s)
```

```
[1, 2, 1, 2, 3]
{1, 2, 3}
```

```
s.add(4)
s.add(8)
s.add(7)
```

```
{1, 2, 3, 4, 7, 8}
```

```
s.discard(2)
```

```
{1, 3, 4, 7, 8}
```

# Множества (Sets) (2)

```
s1 = {1, 2, 3, 4, 5}
s2 = {4, 5, 6, 7, 8}
s3 = {4, 5}
print(s1.intersection(s2))
```

{4, 5}

```
print(s1.difference(s2))
```

{1, 2, 3}

```
print(s1.issubset(s3))
```

False

```
print(s3.issubset(s1))
```

True

```
print(s1.union(s2))
```

{1, 2, 3, 4, 5, 6, 7, 8}

```
print(s1.issuperset(s3))
```

True

# Кортежи (Tuples)

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

```
lst = [1, 2, 3]
t = tuple(lst)
print(t)
(1, 2, 3)
```

```
print(len(t))
3
```

# Словари (Dictionaries)

```
dict = {'name' : 'v1', 'type': 'source', 'value': 5}
print(dict)
```

```
{'name': 'v1', 'type': 'source', 'value': 5}
```

```
dict = {
    'v1' : { 'type': 'source', 'value': 5 },
    'r1' : { 'type': 'resistor', 'value': 1000 }
}
```

```
{'v1': {'type': 'source', 'value': 5}, 'r1': {'type': 'resistor', 'value': 1000}}
```

```
dict['c1'] = {'type': 'capacitor', 'value': 10**-12}
```

# Упорядоченные и неупорядоченные типы данных

Список:

```
l = list('Hello, world')
```

```
print(l)
```

```
['H', 'e', 'l', 'l', 'o', ',', ' ', 'w', 'o', 'r', 'l', 'd']
```

Множество:

```
s = set('Hello, world')
```

```
print(s)
```

```
{'d', 'e', 'H', 'r', 'o', ' ', 'l', 'w', ','}
```

# Синтаксис языка Python: условия

```
var1 = 4
var2 = 6

if var1 == var2:
    print("var1 = var2")
else:
    print("var1 != var2")
```

```
var1 = 4
var2 = 6

if var1 == var2:
    print("var1 = var2")
elif var1 < var2:
    print("var1 < var2")
else:
    print("var1 > var2")
```

Трёхместное выражение:

```
X = input()

if X == 'Y':
    A = True
else:
    A = False
```



```
A = True if X == 'Y' else False
```



# Синтаксис языка Python: циклы

```
for i in range(10):  
    print(i)
```

```
for i in range(1, 10):  
    print(i)
```

```
for i in range(1, 10, 2):  
    print(i)
```

```
lst = [1, 'two', 3, 'four', 5]
```

```
for i in lst:  
    print(f'i={i} of type {type(i)}')
```

```
i=1 of type <class 'int'>  
i=two of type <class 'str'>  
i=3 of type <class 'int'>  
i=four of type <class 'str'>  
i=5 of type <class 'int'>
```

# Python: работа с файлами

```
f = open("myfile.txt", 'r')  
  
text = f.read()  
  
print (text)  
  
f.close()
```

```
f = open("myfile.txt", 'r')  
  
for line in f:  
    print (line)  
  
f.close()
```

```
f = open("myfile.txt", 'r')  
  
while True:  
    line = f.readline()  
    if line == "":  
        break  
    print (line)  
  
f.close()
```

# Синтаксис языка Python: функции (1)

```
def func1(a, b, strval):  
    c = a+b  
    print (strval)  
    return c
```

```
sum = func1(4, 5, "String value!")  
print (sum)
```

```
sum = func1(strval ="String", a=4, b=6)  
print(sum)
```

# Документирование программного кода в Python

```
def RequestAirHT(sensorId : int) -> Tuple[float, float]:  
    """  
    Функция посылает запрос типа GET для получения значений температуры и  
    влажности воздуха  
    URL для запроса: https://dt.miet.ru/ppo_it/api/temp_hum/<number>  
    Вход : (int) номер датчика в диапазоне [1..4]  
    Выход: (int, int) значения температуры и влажности воздуха  
    """  
    ret = requests.get(f'https://dt.miet.ru/ppo_it/api/temp_hum/{sensorId}')  
  
    print(f'Status: {ret.status_code} Answer: {ret.text}')  
  
    json_code = json.loads(ret.text)  
  
    return (float(json_code['temperature']), float(json_code['humidity']))
```

```
print(help(RequestAirHT))
```

```
АН1, АТ1 = RequestAirHT(1)
```

Help on function RequestAirHT in module `__main__`:

RequestAirHT(sensorId: int)

Функция посылает запрос типа GET для получения значений температуры и  
влажности воздуха

URL для запроса: `https://dt.miet.ru/ppo_it/api/temp_hum/<number>`

Вход : (int) номер датчика в диапазоне [1..4]

Выход: (int, int) значения температуры и влажности воздуха

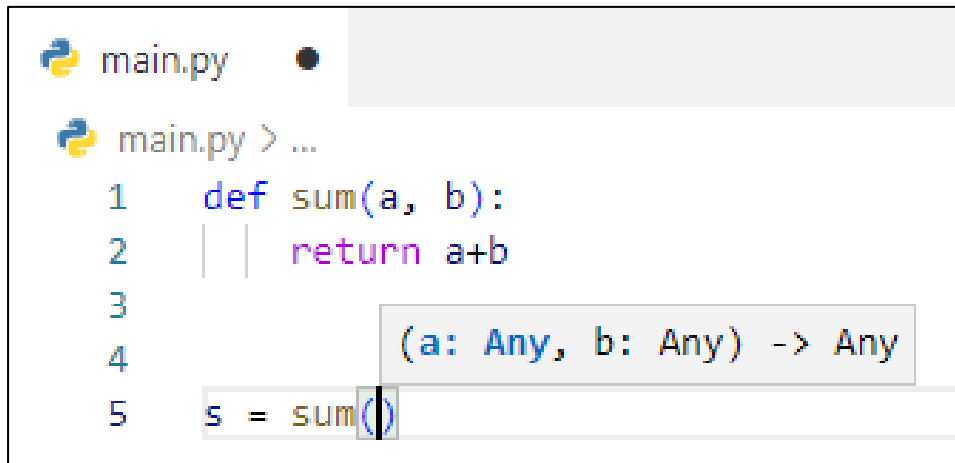
None

# Синтаксис языка Python: аннотирование типов данных

```
def RequestGroundH(sensorId : int) -> float:  
    """
```

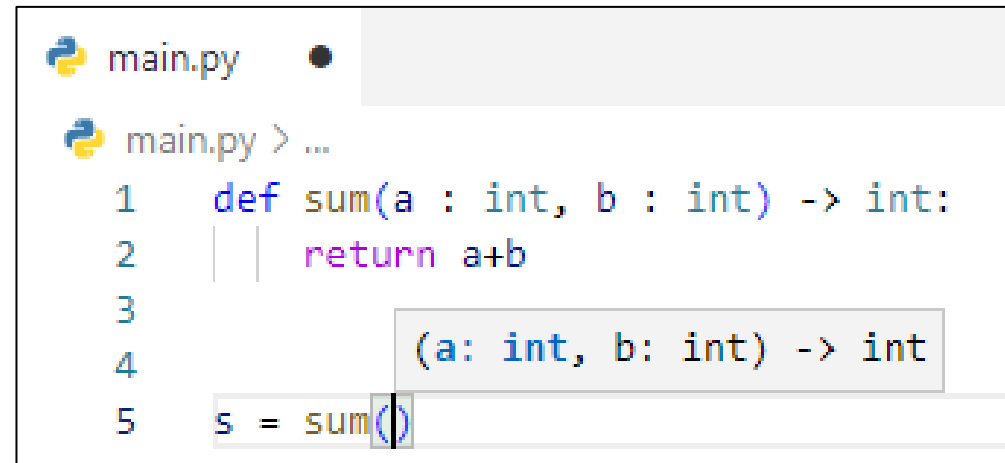
Функция посылает запрос типа GET для получения значения влажности почвы  
URL для запроса: `https://dt.miet.ru/ppo_it/api/hum/<number>`  
Вход : (int) номер датчика в диапазоне [1..6]  
Выход: (float, float) значения температуры и влажности воздуха  
"""

```
ret = requests.get(f'https://dt.miet.ru/ppo_it/api/hum/{sensorId}')  
print(f'Status: {ret.status_code} Answer: {ret.text}')  
json_code = json.loads(ret.text)  
return (float(json_code['humidity']))
```



```
main.py  
main.py > ...  
1 def sum(a, b):  
2     return a+b  
3  
4  
5 s = sum()
```

(a: Any, b: Any) -> Any



```
main.py  
main.py > ...  
1 def sum(a : int, b : int) -> int:  
2     return a+b  
3  
4  
5 s = sum()
```

(a: int, b: int) -> int

# Регулярные выражения в Python

```
import re
```

```
ret = re.search(r'^\s*([Rr]\w+)\s+(\w+)\s+(\w+)', line)
```

```
import re
```

```
line = 'R1 1 2 1K'
```

```
ret = re.search(r'^\s*([Rr]\w+)\s+(\w+)\s+(\w+)', line)
```

```
if ret:
```

```
    print(ret.group(0))
```

```
    print(ret.group(1))
```

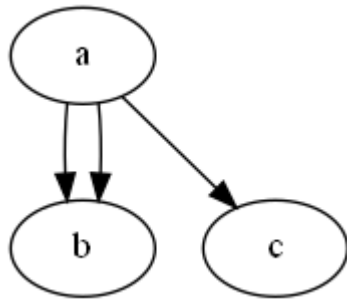
```
    print(ret.group(2))
```

```
    print(ret.group(3))
```

# Модули в Python: graphviz

Язык DOT

```
digraph {  
    a -> b  
    a -> b  
    a -> c  
}
```



Язык Python и библиотека graphviz

```
from graphviz import Digraph
```

```
graph = Digraph()
```

```
graph.edge('a', 'b')
```

```
graph.edge('a', 'b')
```

```
graph.edge('a', 'c')
```

```
graph.render('result.dot', format='png', view=True)
```

# Модули в Python: numpy



```
import numpy as np
```

```
a = np.array([1, 2, 3])
```

Создание массива по диапазону:

```
a = np.arange(0, 10, 1)  
a = np.arange(0, 1, 0.1)
```

Создание массива по диапазону и числу разбиений:

```
a = np.linspace(0, 2, 7)
```



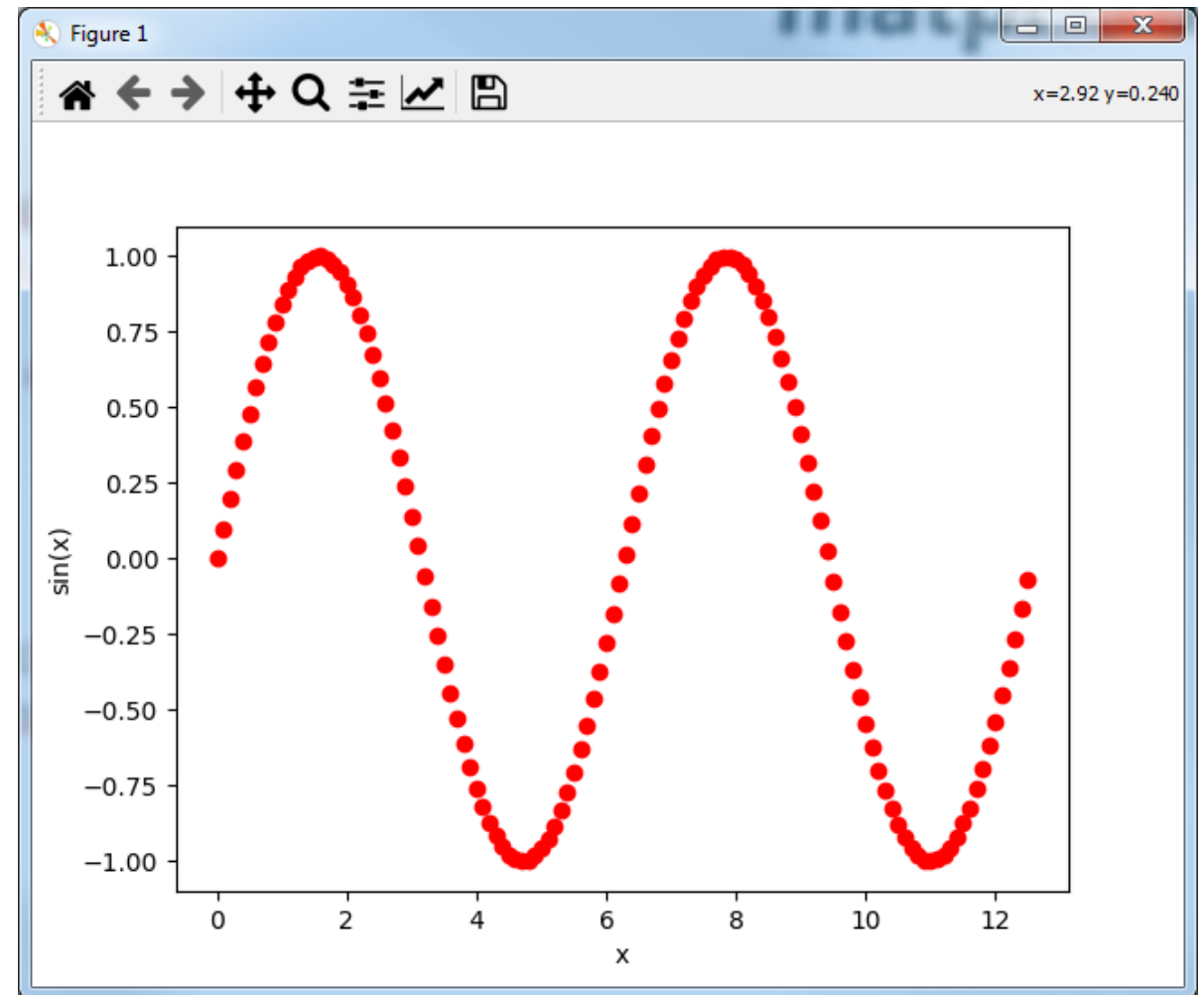
# Модули в Python: matplotlib



```
import numpy as np
import matplotlib.pyplot as plt
```

```
x = np.arange(0, 4*np.pi, 0.1)
y = np.sin(x)
```

```
plt.xlabel('x')
plt.ylabel('sin(x)')
plt.plot(x, y, 'ro')
plt.show()
```



# Модули в Python: PyQt5



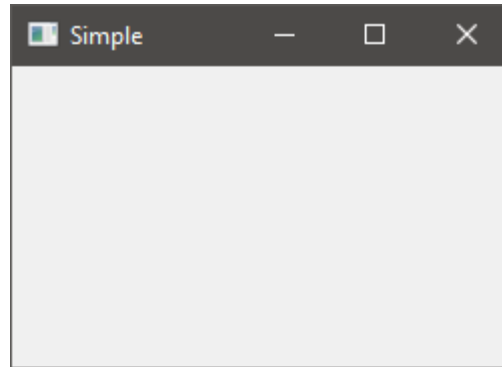
```
import sys
from PyQt5.QtWidgets import QApplication, QWidget

if __name__ == '__main__':

    app = QApplication(sys.argv)

    win = QWidget()
    win.resize(250, 150)
    win.move(300, 300)
    win.setWindowTitle('Simple')
    win.show()

    sys.exit(app.exec_())
```



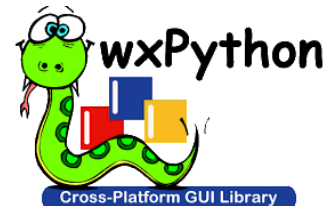
```
#include <QApplication>
#include <QWidget>

int main(int argc, char *argv[]) {
    QApplication app(argc, argv);

    QWidget *win = new QWidget;
    win->resize(250, 150);
    win->move(300, 300);
    win->setWindowTitle("Simple");
    win->show();

    return app.exec();
}
```

# Модули в Python: wxPython

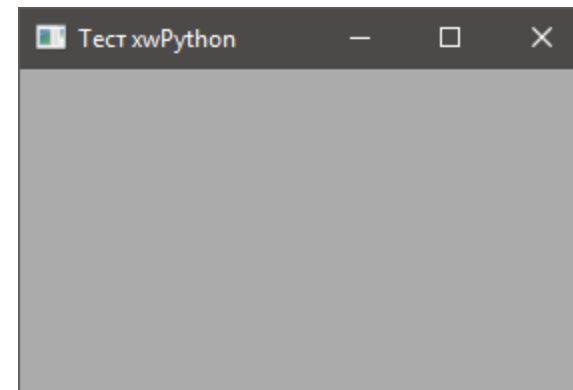


```
import wx
```

```
class Example(wx.Frame):  
    def __init__(self, title):  
        super(Example, self).__init__(None, title=title,  
                                       size=(300, 200))  
        self.Move((800, 250))
```

```
def main():  
    app = wx.App()  
    ex = Example(title='Тест wxPython')  
    ex.Show()  
    app.MainLoop()
```

```
if __name__ == '__main__':  
    main()
```



# Модули в Python: MyHDL (1)

```
module inv(x, y);          entity inv is                from myhdl import *
    input  x;              port(x: in bit;                @block
    output y;              y: out bit);          def inv(x, y):
                            end inv;
                            architecture BEH of inv is    @always(x)
                            begin                          def logic():
                                process(x)                  y.next = not x
                                begin
                                    y <= not x;
                                end process;
                            end RTL;                        return logic
```

# Модули в Python: MyHDL (2)

```
@block
```

```
def test_inv():
```

```
    x, y = [Signal(1)]
```

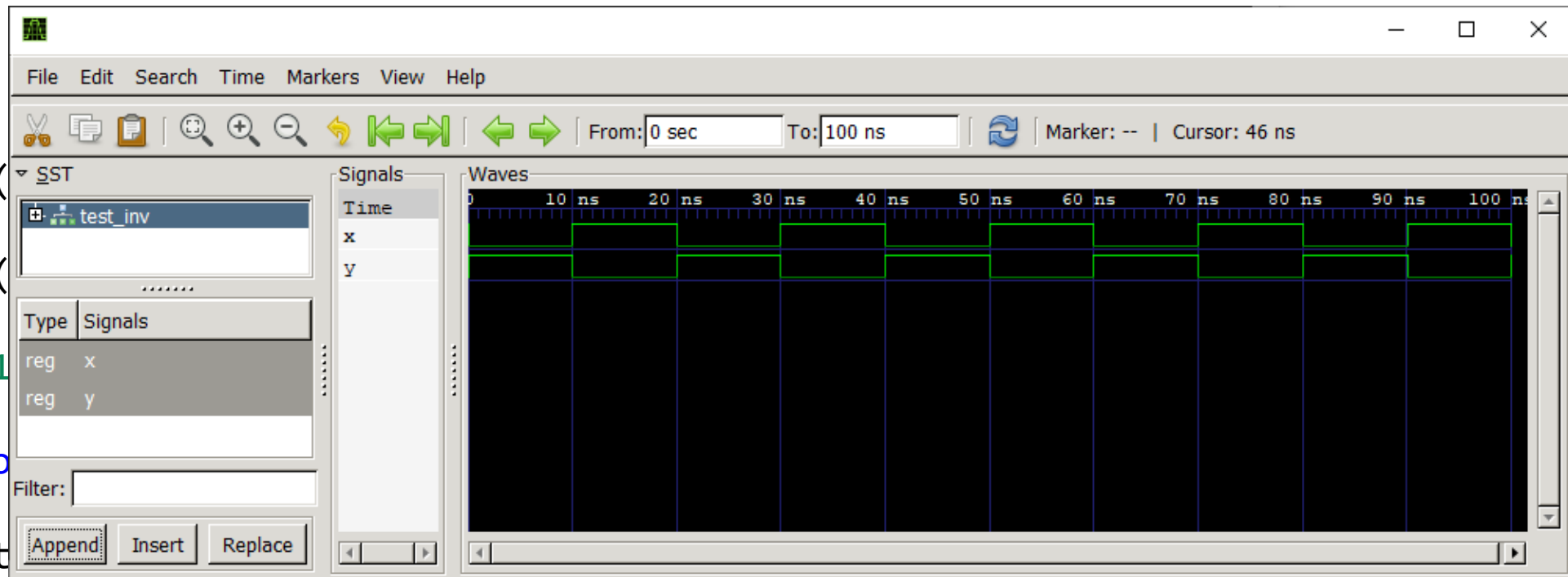
```
    inv_inst = inv(x)
```

```
    @always(delay(10))
```

```
    def gen_x():
```

```
        x.next = not x
```

```
    return inv_inst
```



```
def simulate(timesteps):
```

```
    simInst = test_inv()
```

```
    simInst.config_sim(trace=True, traceback=False)
```

```
    simInst.run_sim(timesteps, quiet=0)
```

```
simulate(100)
```