

**КАЛУЖСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО
ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ Н.Э. БАУМАНА
(национальный исследовательский университет)»**



Факультет «Информатика и управление»

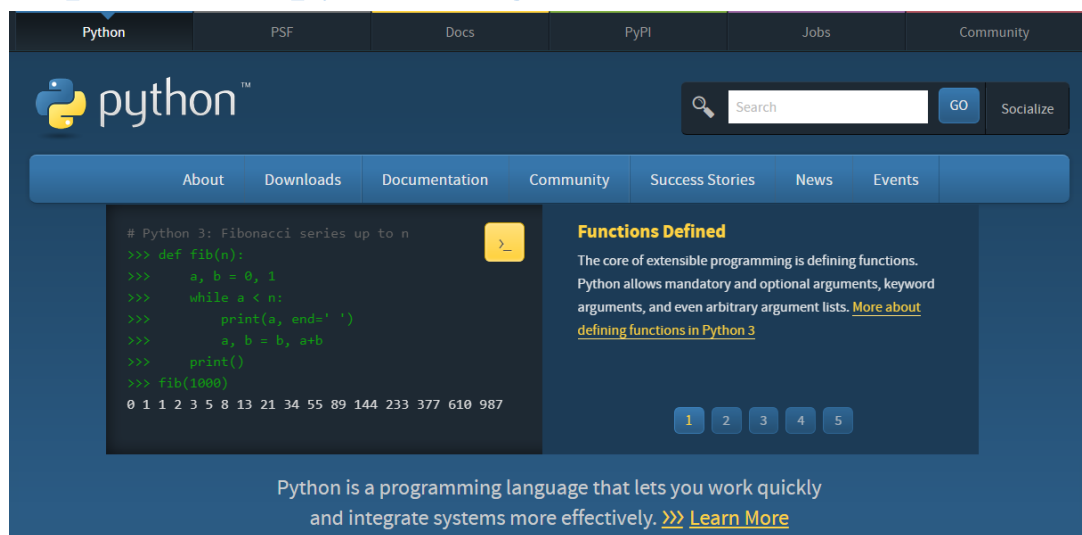
Кафедра «Программное обеспечение ЭВМ, информационные технологии»

Высокоуровневое программирование

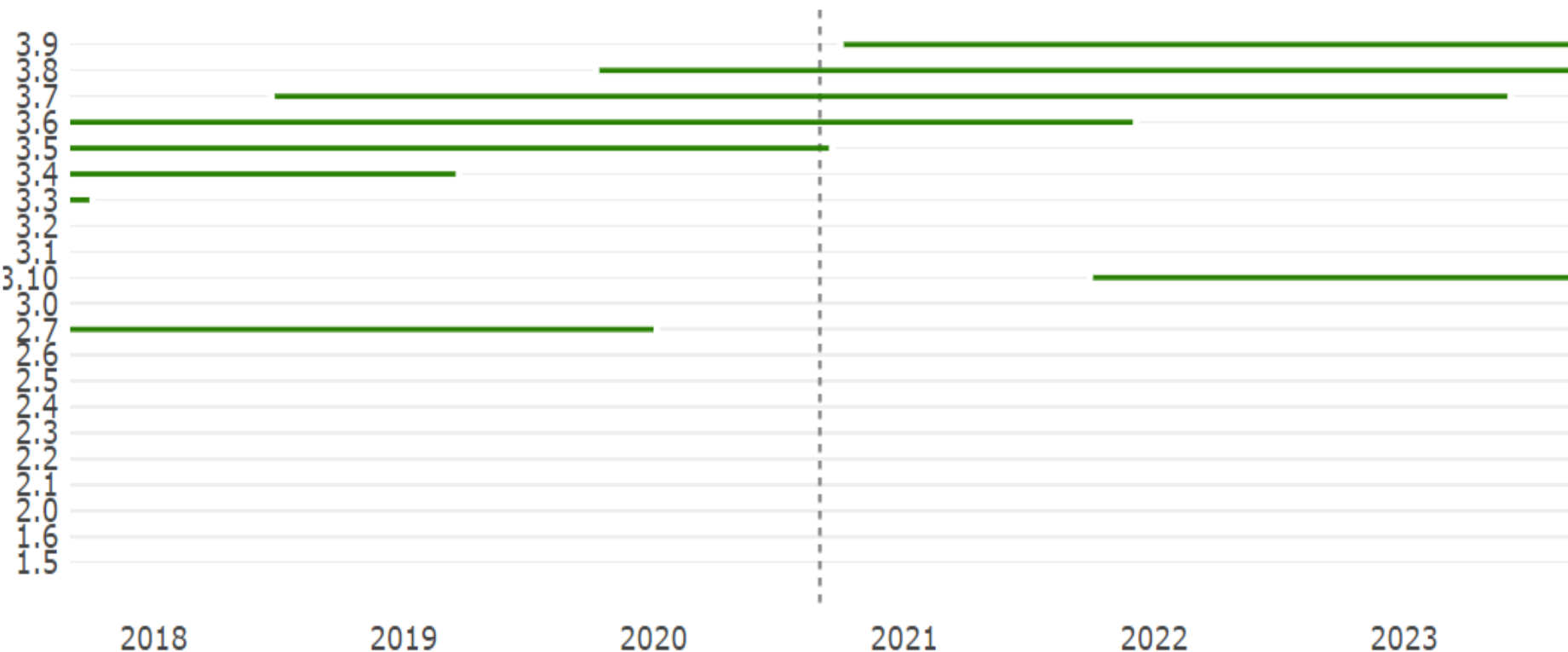
Лекция №1. «Программирование на Python: Введение»

История создания

Впервые язык Python был анонсирован в **1991** году голландским разработчиком Гвидо Ван Россумом. С тех пор данный язык проделал большой путь развития. В 2000 году была издана версия 2.0, а в 2008 году - версия 3.0. Несмотря на вроде такие большие промежутки между версиями постоянно выходят подверсии. Так, текущей актуальной версией на момент написания данного материала является **3.8**. Более подробную информацию о всех релизах, версиях и изменения языка, а также собственно интерпретаторы и необходимые утилиты для работы и прочую полезную информацию можно найти на официальном сайте <https://www.python.org/>.



Время жизни версий



Область применения

- Веб-приложения
- Игры
- Настольные программы
- Работа с базами данных
- Анализ данных
- Область машинного обучения
- Исследования искусственного интеллекта

Отличия языка Python

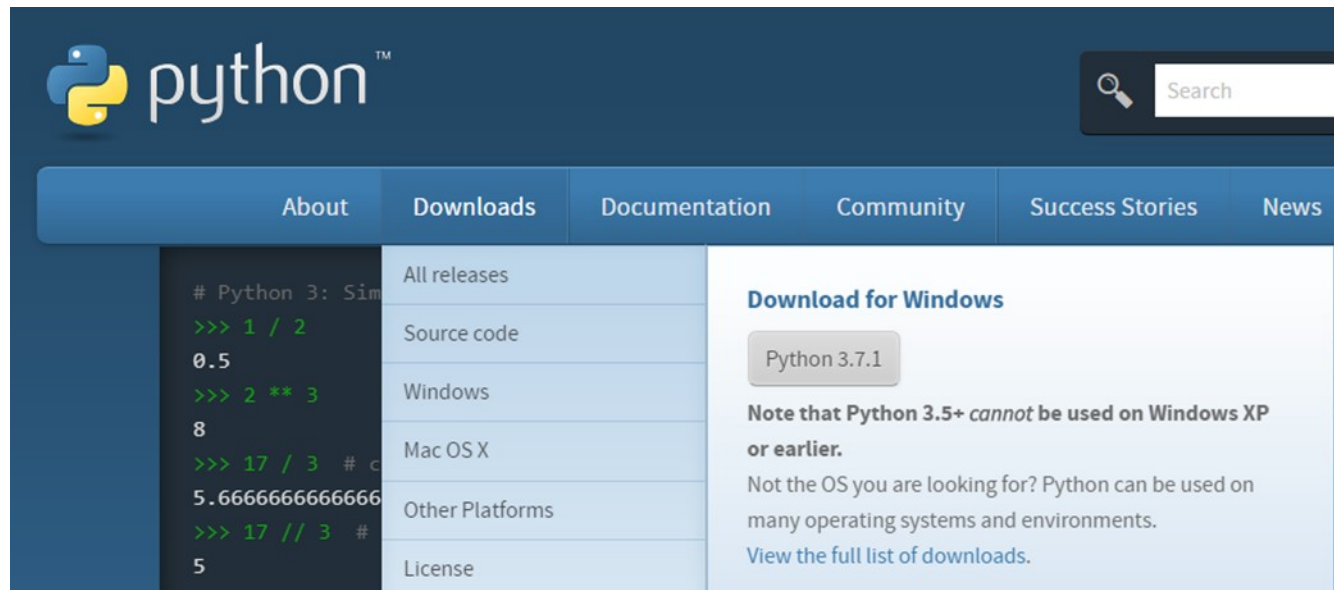
- **Интерпретируемый язык** (портативность и платформонезависимость)
- Динамический язык
- Огромное количество сервисов и сторонних библиотек
- Большое количество открытой информации по языку

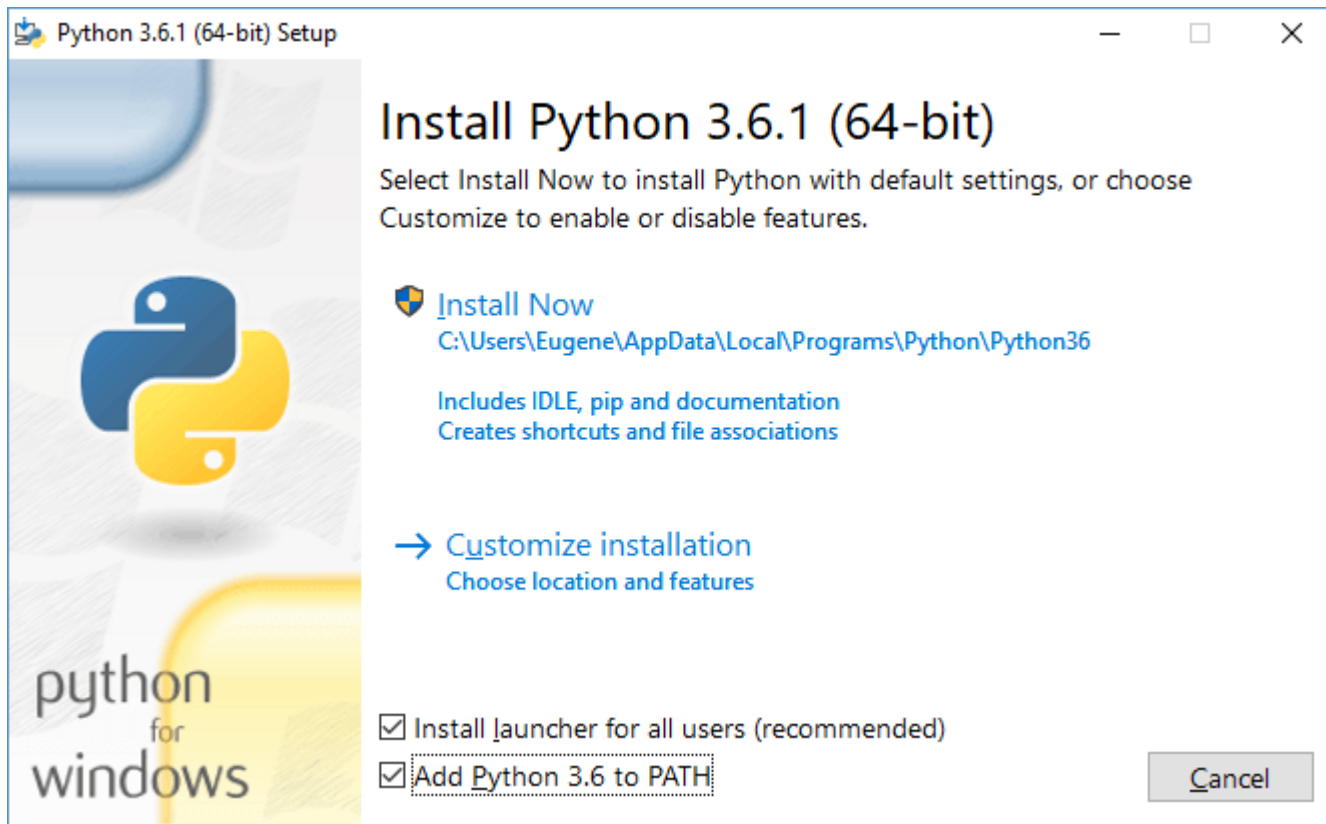
Выполнение программы на Python



Установка Python

Для создания программ на Python нам потребуется интерпретатор. Для его установки перейдем на сайт <https://www.python.org/> и на главной странице перейдем к последней версии языка.

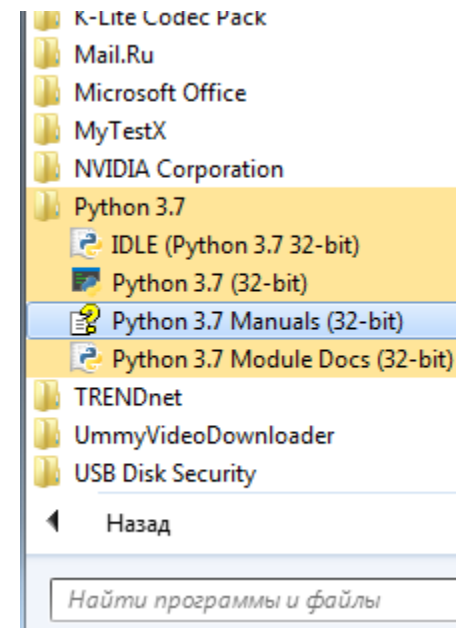




Здесь мы можем задать путь, по которому будет устанавливаться интерпретатор. Оставим его по умолчанию, то есть `C:\Users\[имя_пользователя]\AppData\Local\Programs\Python\Python36\`.

Кроме того, в самом низу отметим флажок "Add Python 3.6 to PATH", чтобы добавить путь к интерпретатору в переменные среды.

После установки в меню Пуск на ОС Windows мы сможем найти иконки для доступа к разным утилитам питона:



Первая программа в командной строке Windows или в IDLE Python

```
c:\Python\Scripts>cd c:\python
```

```
c:\Python>py
```

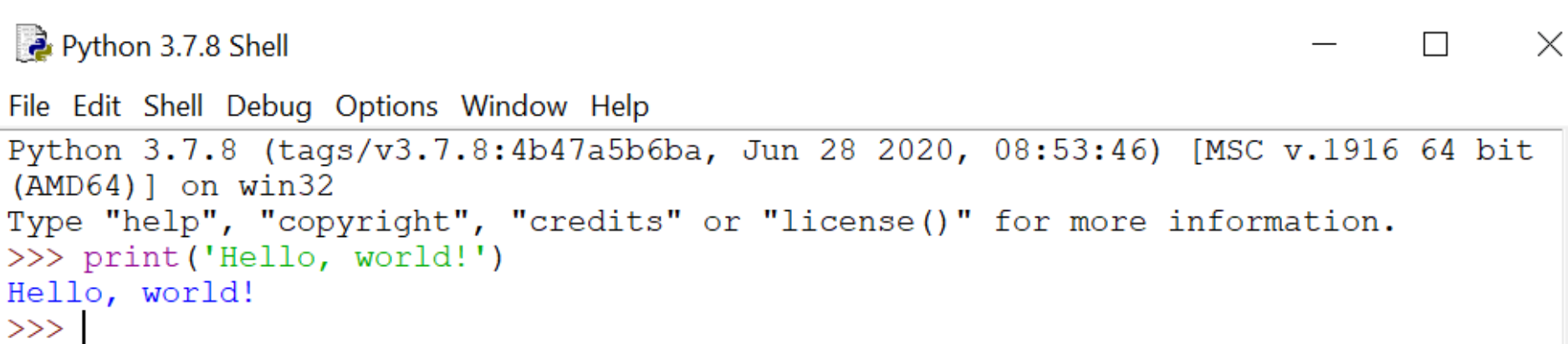
```
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:43:08) [MSC v.1926 32 bit (Intel)] on win32
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> print('Hello, world!')
```

```
Hello, world!
```

```
>>>
```

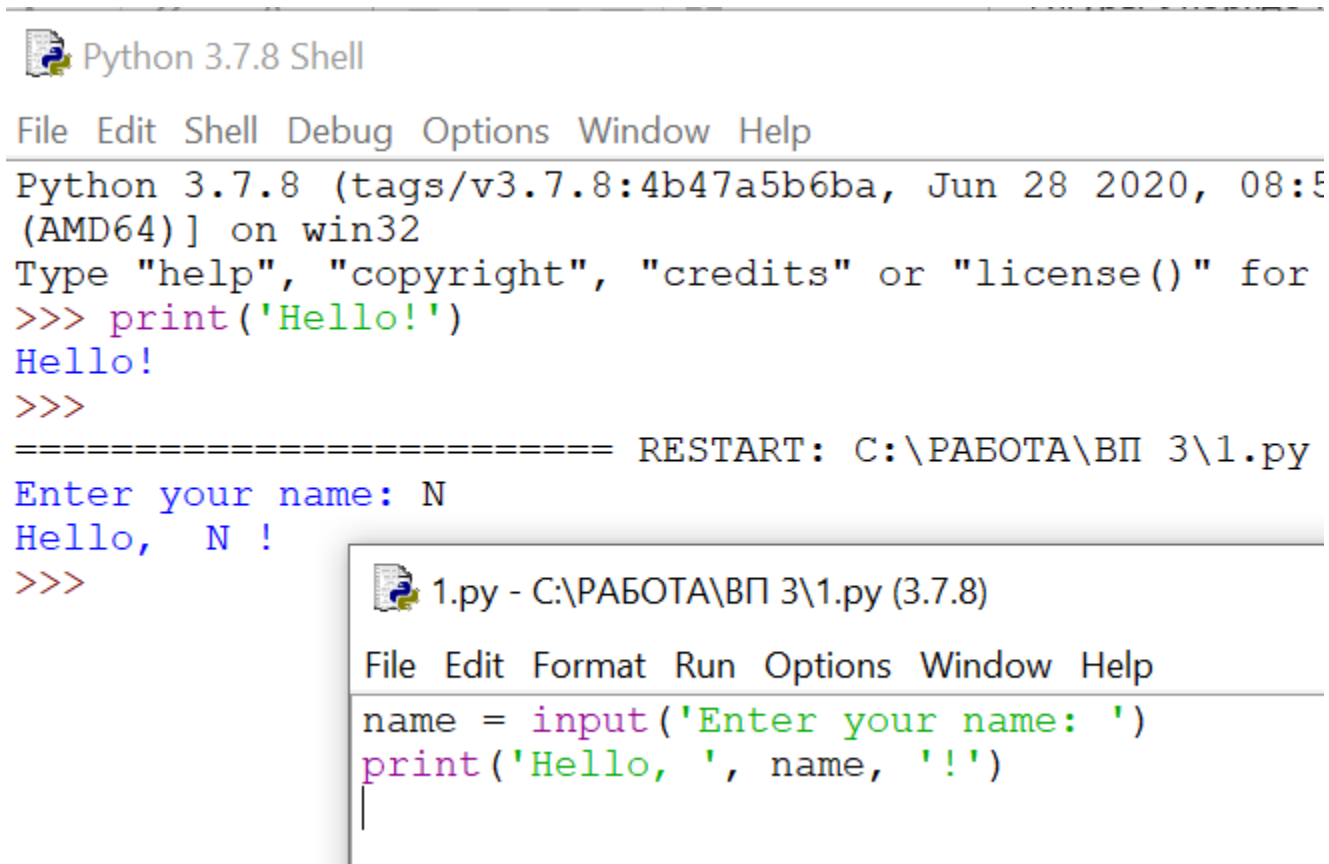


The screenshot shows a window titled "Python 3.7.8 Shell" with standard Windows window controls (minimize, maximize, close). The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following content:

```
Python 3.7.8 (tags/v3.7.8:4b47a5b6ba, Jun 28 2020, 08:53:46) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> print('Hello, world!')
Hello, world!
>>> |
```

Создание файла программы

- File -> New



The image shows two overlapping windows from a Python 3.7.8 environment. The background window is titled 'Python 3.7.8 Shell' and displays the following text:

```
File Edit Shell Debug Options Window Help
Python 3.7.8 (tags/v3.7.8:4b47a5b6ba, Jun 28 2020, 08:5
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for
>>> print('Hello!')
Hello!
>>>
===== RESTART: C:\РАБОТА\ВП 3\1.py
Enter your name: N
Hello, N !
>>>
```

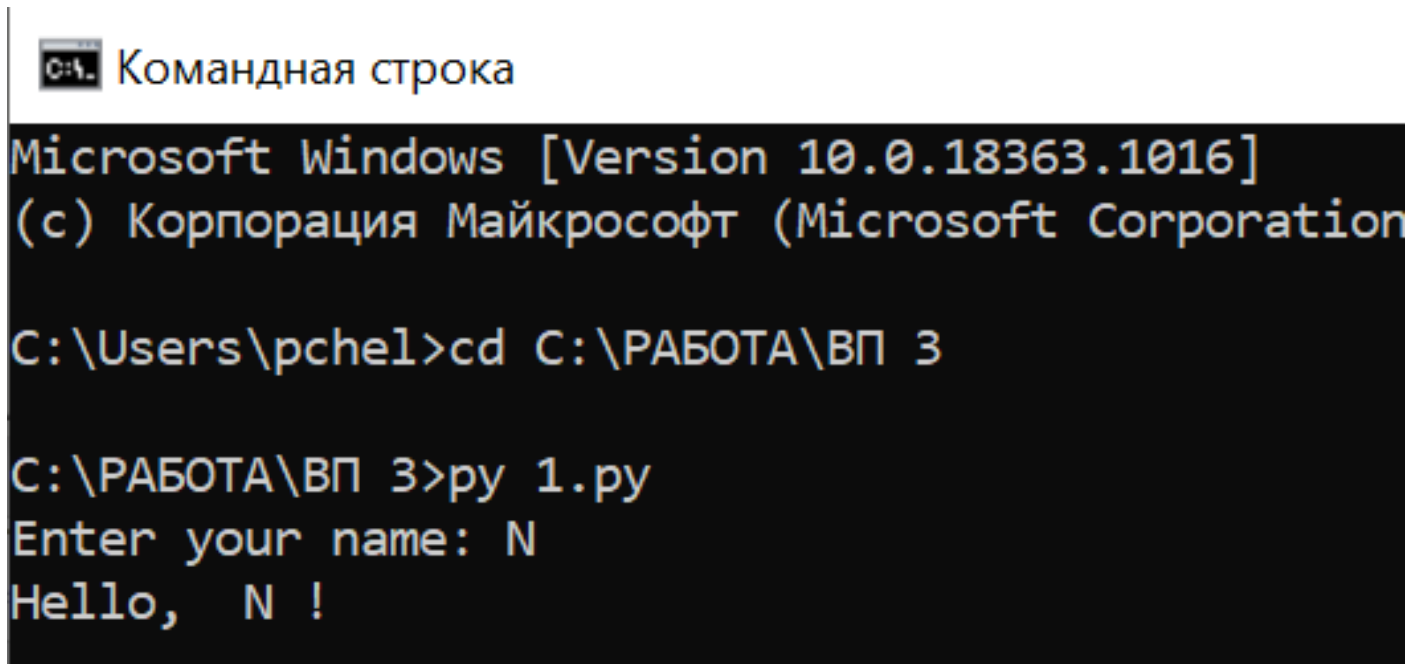
The foreground window is titled '1.py - C:\РАБОТА\ВП 3\1.py (3.7.8)' and displays the following code:

```
File Edit Format Run Options Window Help
name = input('Enter your name: ')
print('Hello, ', name, '!')
```

Запуск программы из командной строки

Скрипт состоит из двух строк. Первая строка с помощью метода **input()** ожидает ввода пользователем своего имени. Введенное имя затем попадает в переменную **name**.

Вторая строка с помощью метода **print()** выводит приветствие вместе с введенным именем.



```
C:\> Командная строка

Microsoft Windows [Version 10.0.18363.1016]
(c) Корпорация Майкрософт (Microsoft Corporation)

C:\Users\pchel>cd C:\РАБОТА\ВП 3

C:\РАБОТА\ВП 3>py 1.py
Enter your name: N
Hello,  N !
```

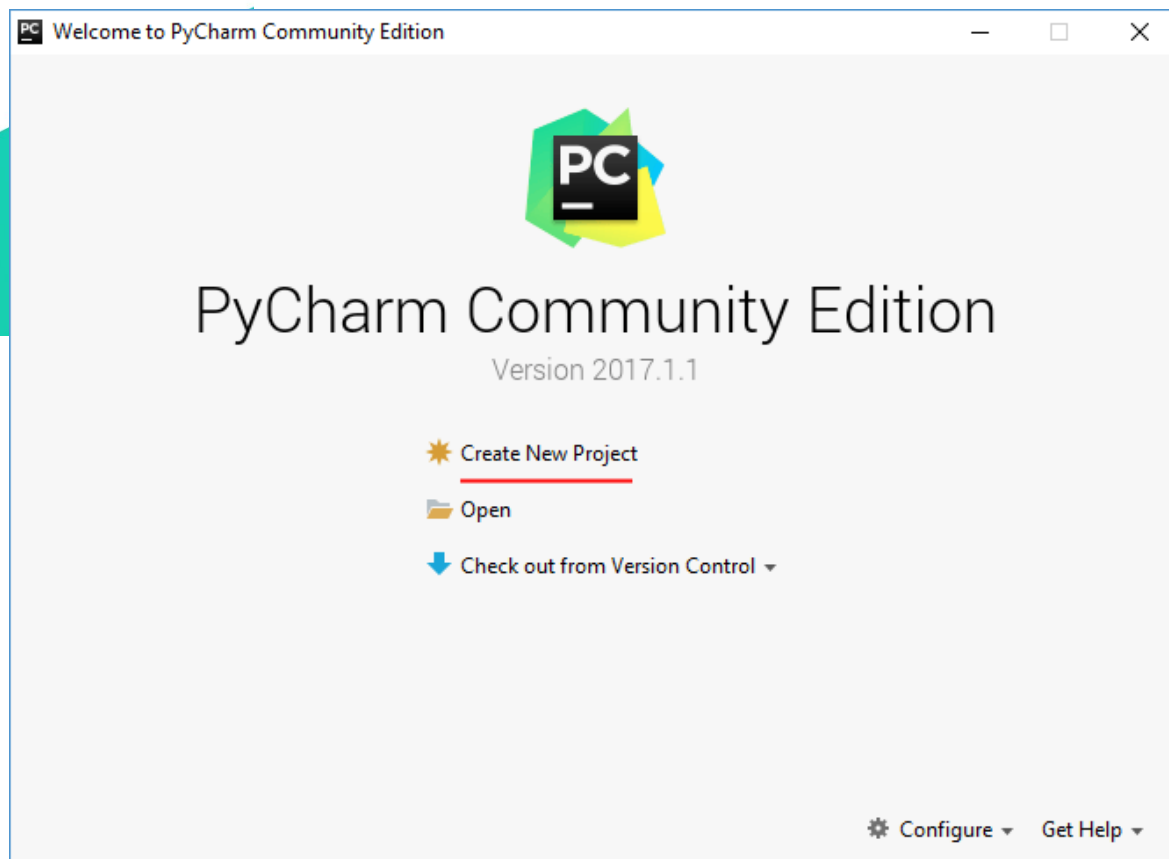
IDE PyCharm

- Ранее было описано создание простейшего скрипта на языке Python. Для создания скрипта использовался текстовый редактор. Но есть и другой способ создания программ, который представляет использование различных **интегрированных сред разработки или IDE**.
- IDE предоставляют нам текстовый редактор для набора кода, но в отличие от стандартных текстовых редакторов, IDE также обеспечивает полноценную подсветку синтаксиса, автодополнение или интеллектуальную подсказку кода, возможность тут же выполнить созданный скрипт, а также многое другое.
- Для Python можно использовать различные среды разработки, но одной из самых популярных из них является среда **PyCharm**, созданная компанией JetBrains. Эта среда динамично развивается, постоянно обновляется и доступна для наиболее распространенных операционных систем - Windows, MacOS, Linux.
- Правда, она имеет одно важное ограничение. А именно она доступна в двух основных вариантах: платный выпуск Professional и бесплатный Community. Многие базовые возможности доступны и в бесплатном выпуске Community. В то же время ряд возможностей, например, веб-разработка, доступны только в платном Professional.

В нашем случае воспользуемся бесплатным выпуском Community. Для этого перейдем на страницу загрузки и загрузим установочный файл PyCharm Community. После загрузки выполним его установку. <https://www.jetbrains.com/pycharm/>

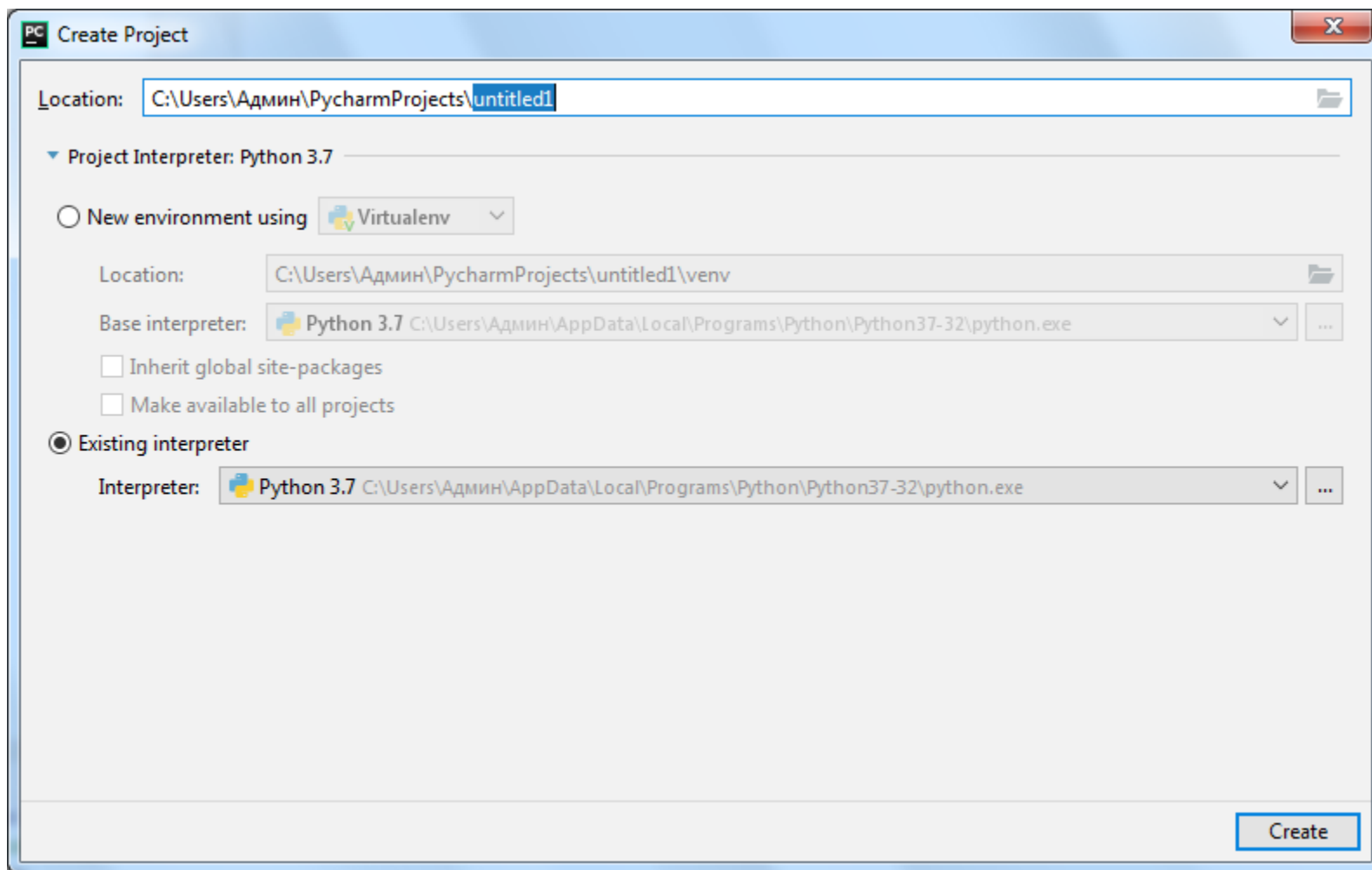


После завершения установки запустим программу. При первом запуске открывается начальное окно:



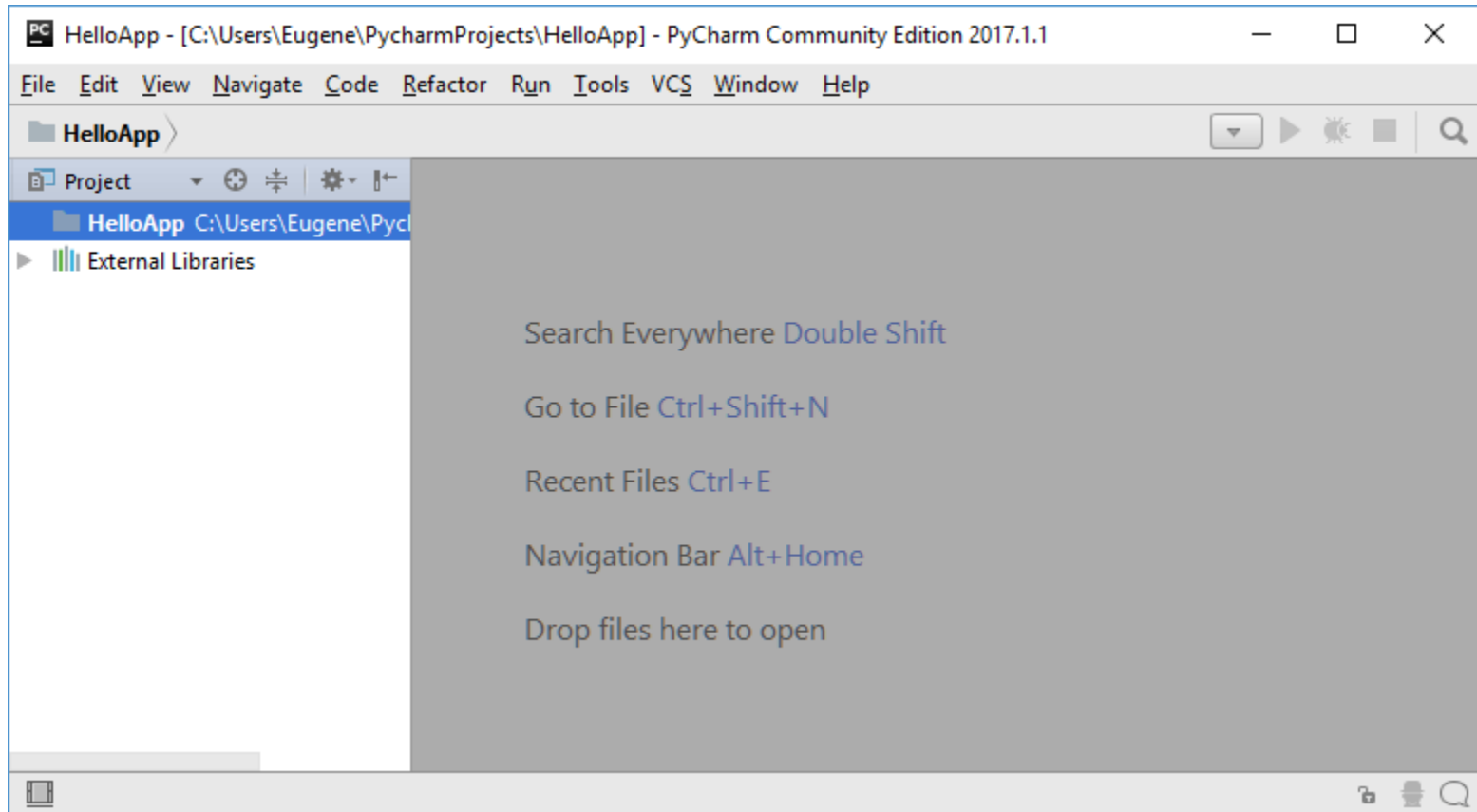
Создадим проект и для этого выберем пункт **Create New Project**.

Далее нам откроется окно для настройки проекта. Здесь можно указать путь, и также необходимо указать путь к файлу интерпретатора.

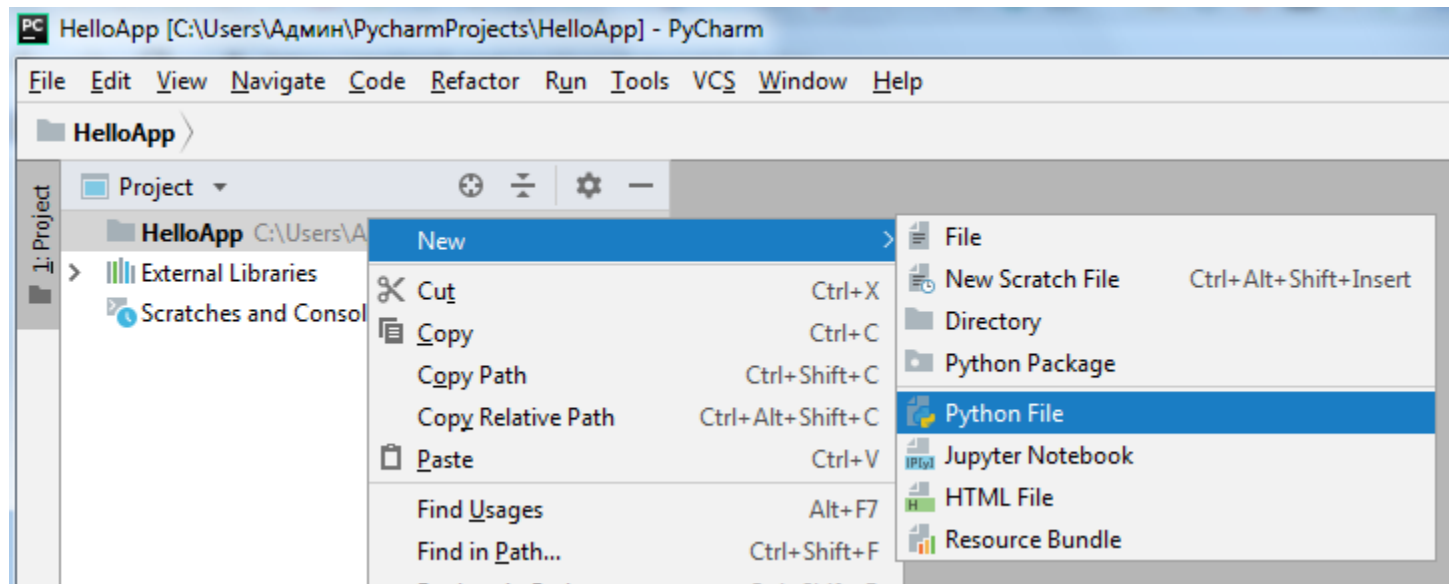


Проект будет помещаться в папку HelloApp. Собственно название папки и будет названием проекта. И после установки всех путей нажмем на кнопку Create для создания проекта.

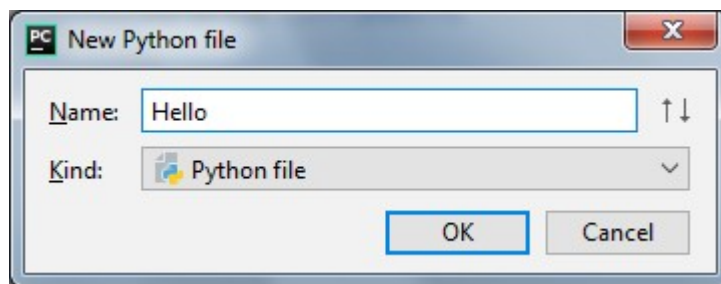
После этого будет создан пустой проект:



Теперь создадим простейшую программу. Для этого нажмем на название проекта правой кнопкой мыши и в появившемся контекстном меню выберем **New -> Python File**.



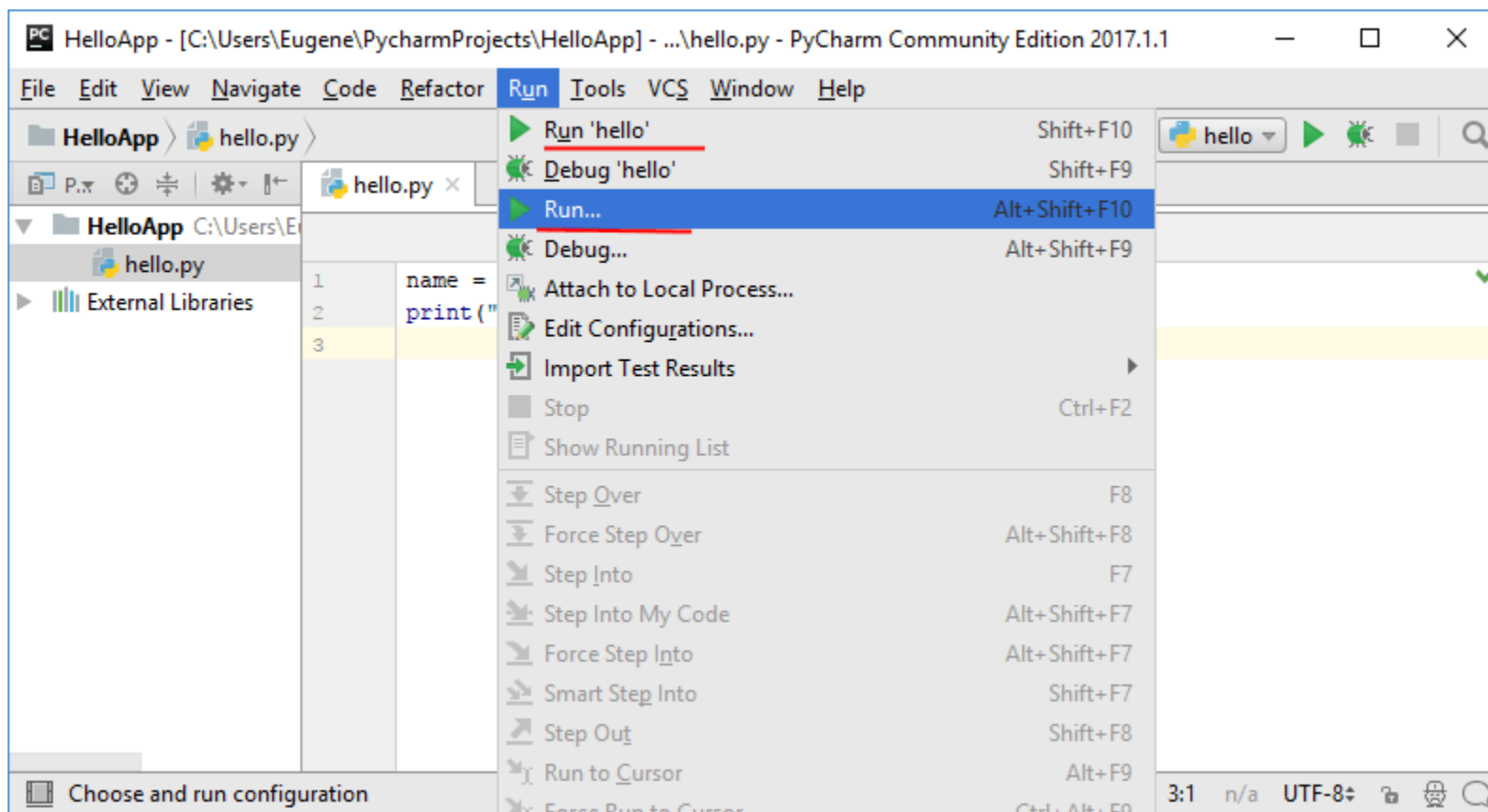
Затем откроется окно, в котором надо будет указать название файла. Пусть файл называется **hello**:



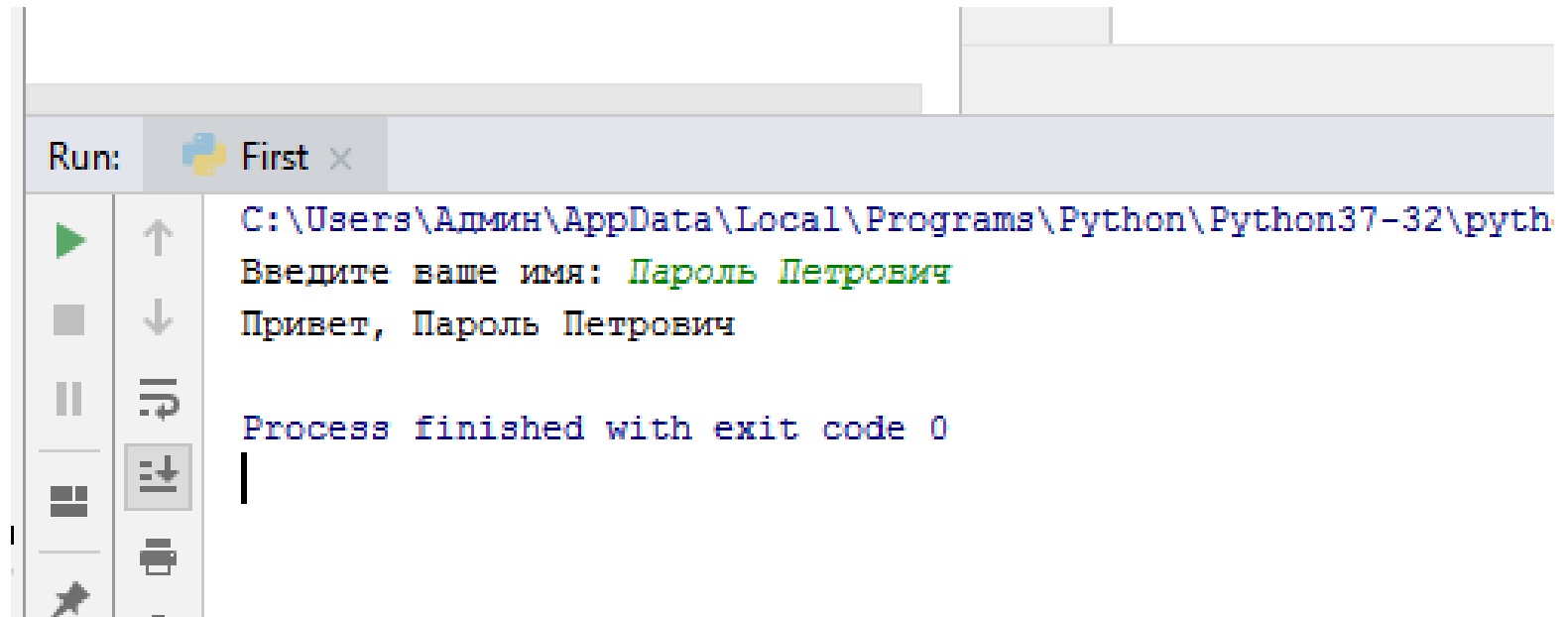
В созданный файл введем следующие строки:

```
name = input("Введите ваше имя: ")  
print("Привет,", name)
```

Для запуска скрипта перейдем в меню **Run -> Run 'Hello'** или **Run...**:



После этого внизу IDE отобразится окно вывода, где надо будет ввести имя и где после этого будет выведено приветствие:



The screenshot shows the 'Run' window of an IDE. The title bar reads 'Run: Python First x'. The window contains a vertical toolbar on the left with icons for running, stepping through, and other debugging actions. The main area displays the following text:

```
C:\Users\Админ\AppData\Local\Programs\Python\Python37-32\python.exe
Введите ваше имя: Пароль Петрович
Привет, Пароль Петрович

Process finished with exit code 0
```

Текстовый процессор Jupyter

```
c:\Python\Scripts>pip install notebook
```

```
c:\Python\Scripts>jupyter notebook
```

localhost:8888/tree

Почта 26 Календарь Дневник Gmail OneDrive Dropbox Кругосвет Python C++ Переводчик Вопросы детей

jupyter Quit Log

Files Running Clusters

lect items to perform actions on them.

Upload New

0 /

	Name
<input type="checkbox"/>	FirstCourse.ipynb
<input type="checkbox"/>	SecondCourse.ipynb
<input type="checkbox"/>	UFO_analysis.ipynb
<input type="checkbox"/>	abc.txt
<input type="checkbox"/>	cipy_install_3.8.exe

Notebook:
Python 3

Other:
Text File
Folder
Terminals Unavailable

localhost:8888 07.1

Первая программа в jupyter

Ноутбук для первой лекции

In [2]:

```
1 name = input('Enter your name: ')\n2 print('Hello, ', name, '!')
```

Enter your name: N
Hello, N !

In [3]:

```
1 print(name)
```

N

In []:

```
1
```

Полезные ссылки

- Официальный сайт Python, где можно скачать интерпретатор (Python 3): <https://www.python.org/>
- Официальная документация по Python: <https://docs.python.org/3/>
- Среда для написания программ PyCharm Community Edition: <https://www.jetbrains.com/pycharm/>
- Текстовый редактор с подсветкой синтаксиса программ Sublime Text3: <http://www.sublimetext.com/3>
- Визуализатор программ <http://pythontutor.com/visualize.html#mode=edit>
- Интерактивный учебник языка Python (на русском языке): <http://pythontutor.ru/>

Особенности синтаксиса программ

- Каждая инструкция с новой строки
- Вложенность регулируется отступами

```
In [5]: 1 n = int(input())
        2 if n > 0:
        3     print('positive')

5
positive
```

```
In [6]: 1 n = 5
        2 N = 10
        3 print(n, N)

5 10
```

- Регистрозависимость
- Комментарии (ctrl + /)

```
In [ ]: 1 # комментарий|

In [ ]: 1 """
        2 многослойный
        3 комментарий|
        4 """
```

- В Python есть специальный документ, в котором описано как лучше писать код Python **PEP 8** - the Style Guide for Python Code

Переменные

Переменные в Python не требуют объявления типа переменной (так как Python – язык с динамической типизацией) и являются ссылками на область памяти.

Правила именования переменных:

- имя переменной может состоять только из букв, цифр и знака подчёркивания;
- имя не может начинаться с цифры;
- имя не может содержать специальных символов @, \$, %.

Переменные

Пример создания переменных в Python:

```
In [1]: a = 3

In [2]: b = 'Hello'

In [3]: c, d = 9, 'Test'

In [4]: print(a,b,c,d)
3 Hello 9 Test
```

Переменные являются ссылками на область памяти. Это можно продемонстрировать с помощью **id()**, которая показывает идентификатор объекта:

```
In [5]: a = b = c = 33

In [6]: id(a)
Out[6]: 31671480

In [7]: id(b)
Out[7]: 31671480

In [8]: id(c)
Out[8]: 31671480
```


Типы данных

В Python есть несколько стандартных типов данных:

- Numbers (числа)
- Strings (строки)
- Lists (списки)
- Dictionaries (словари)
- Tuples (кортежи)
- Sets (множества)
- Boolean (логический тип данных)

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)