

Основные элементы языка программирования

Python

Программирование и языки программирования

- Программирование – набор общих, универсальных, абстрактных понятий, представляющих некий **процесс**
- Языки программирования описывают эти понятия для реализации процесса
 - У каждого языка свои особенности, достоинства и недостатки

Как программировать?

Машинный код

- Команды, поступающие в процессор по шинам, представляют собой последовательности нулей и единиц, то есть числа.
- Поэтому программа, с которой работает процессор, представляет собой последовательность чисел, называемую машинным кодом
- Программа «Hello, world!» для процессора архитектуры x86 выглядит следующим образом (в шестнадцатеричном представлении):

```
BB 11 01 B 9 0 D 00 B 4 0 E 8 A 07 43 CD 10 E 2 F 9 CD  
20 48 65 6 C 6 C 6 F 2 C 20 57 6 F 72 6 C 64 21 3
```

Как программировать?

Язык программирования

- Язык программирования – формализованный язык
- Язык программирования определяет «слова», понятные транслятору, и правила записи команд (операторов)

Адреса памяти

7C90104A 00648B0D
7C90104E 1800
7C901050 0000
7C901052 8B542404
7C901056 EBC4

Коды команд

```
add    [ebx+ecx*4+$0d],ah
sbb    [eax],al
add    [eax],al
mov    edx,[esp+$04]
jmp    -$3c
```

Уровни языков программирования

- Если язык программирования ориентирован на конкретный тип процессора и учитывает его особенности, то он называется языком программирования низкого уровня.
 - Пример: язык ассемблера. Представляет каждую команду машинного кода не в виде чисел, а с помощью символьных условных обозначений, называемых мнемониками.
 - Преобразование одной машинной инструкции в одну команду ассемблера называется транслитерацией
- Язык ассемблера применяется там, где требуется компактность, быстродействие и возможность прямого доступа к аппаратным ресурсам.
 - Применяется для написания небольших системных приложений, драйверов, модулей стыковки с нестандартным оборудованием

Уровни языков программирования

- Языки программирования высокого уровня не учитывают конкретных компьютерных архитектур.
 - Создаваемые программы легко переносятся на другие платформы, для которых создан транслятор этого языка

Трансляция программы

Транслятор – это программа, на вход которой подается текст алгоритма на языке программирования – **исходный модуль**, а на выходе (после трансляции) получается программа на машинном языке – **объектный модуль**.

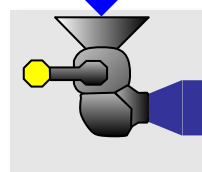


- Транслятор действует по строго формальным правилам: если транслируемая программа содержит хотя бы одну формальную (*синтаксическую*) ошибку, то трансляция не может завершиться!

Компилятор

- **Компилятор** автоматически переводит текст программы в машинный код, который затем можно использовать отдельно от текста исходной программы.
 - Компиляторы полностью обрабатывают весь текст программы.
 - Они просматривают его в поисках ошибок, а затем транслируют в машинный код.

```
main()  
{  
  write('Привет!')  
}
```



1010010100



privet.exe

- В результате законченная программа получается компактной и эффективной и может быть перенесена на другие компьютеры с процессором, поддерживающим этот код.

Интерпретатор

- **Интерпретатор** сразу выполняет команды языка, указанные в тексте программы.
- **Интерпретатор** берет очередной оператор, анализирует его структуру и исполняет, затем происходит переход к следующему оператору и т. д.
 - Программы с большим объемом повторяющихся вычислений могут работать медленно
 - Для работы на другом компьютере требуется наличие интерпретатора.



Python – интерпретатор!

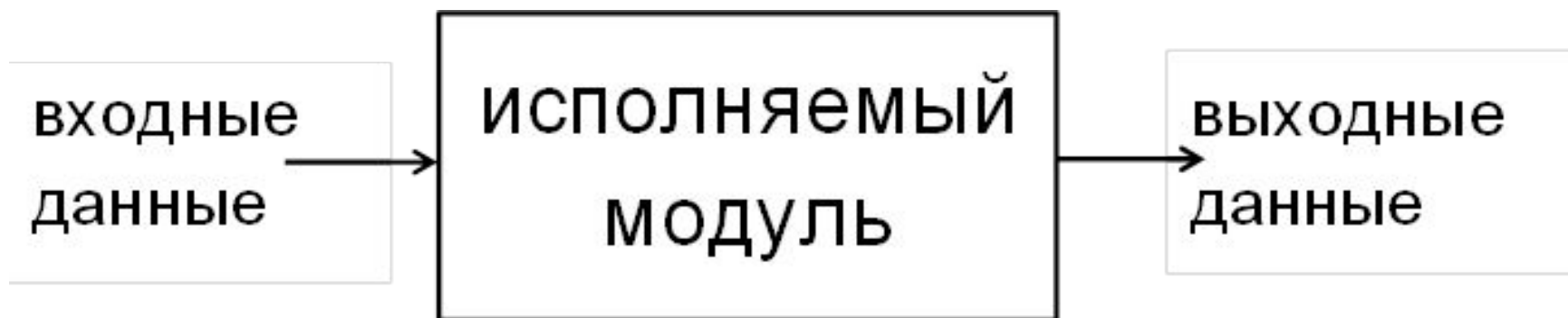
Компоновка программы

Компоновщик (редактор связей) – это программа, которая объединяет объектный модуль и необходимые для выполнения дополнительные модули (поддержка ввода/вывода, стандартные функции и т.д.) в единый **исполняемый модуль**



- В некоторых системах программирования трансляция и компоновка выполняются как один шаг

Схема работы с программой

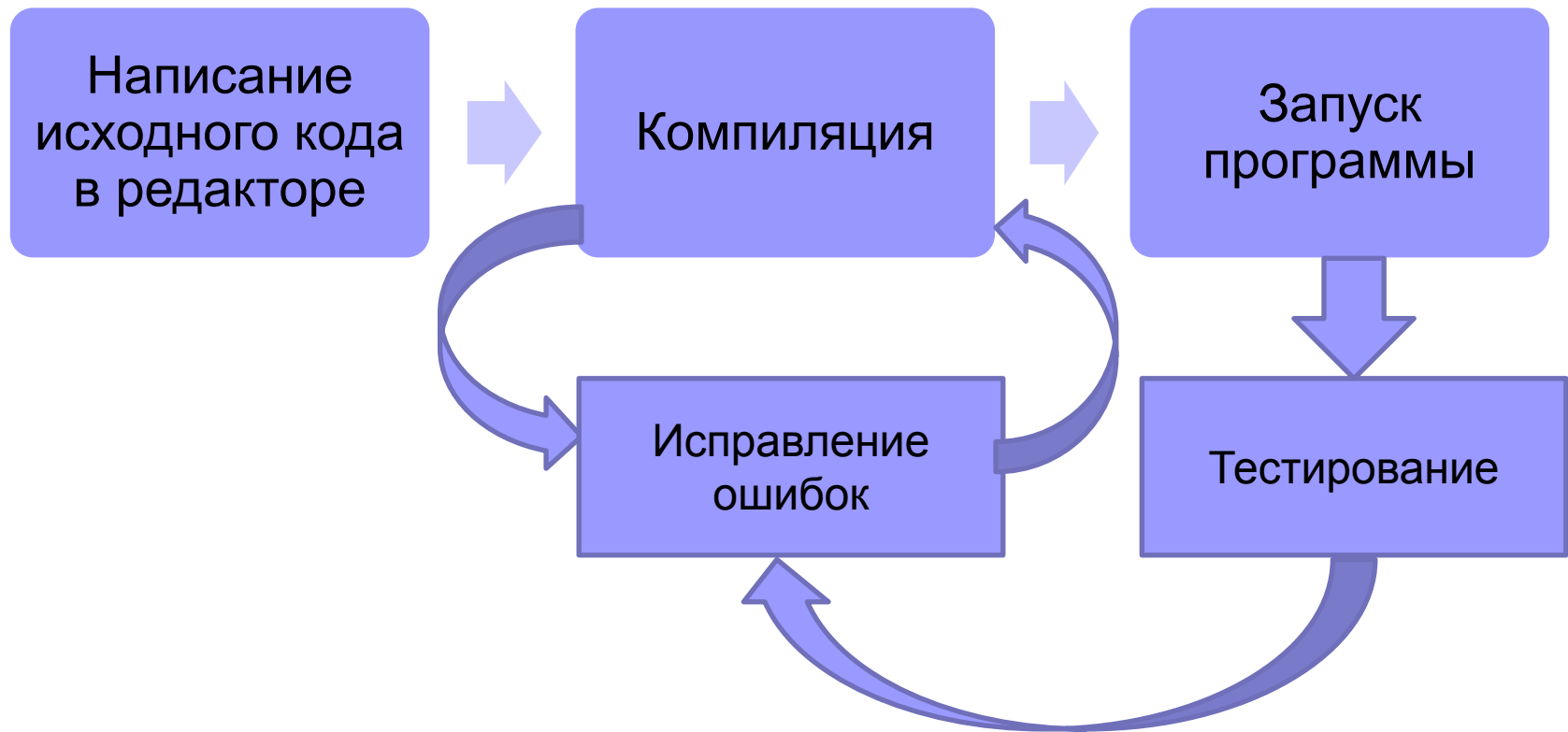


Создание программы

Для создания программы нужны:

- Текстовый редактор
- Транслятор (Компилятор)
- компоновщик (редактор связей)
- Библиотеки функций
- Отладчик

Последовательность разработки программы



Среда разработки

В настоящее время для разработки программ созданы специальные интегрированные системы разработки (**IDE**).

Они предоставляют:

- текстовые редакторы для работы с исходным кодом
- трансляторы и средства сборки программ
- библиотеки функций, классов и визуальных компонент
- инструменты для тестирования

Синтаксис языка

Синтаксис языка программирования:

- Система обозначений правильных последовательностей синтаксических элементов программы
- Формальные правила, которым должна соответствовать программа на некотором языке программирования
- Определяет «слова», понятные транслятору, и правила записи команд (операторов).

Семантика языка

- Семантика языка программирования – это смысл, который закладывается в каждую конструкцию языка.
- Семантический анализ – это проверка смысловой правильности конструкции.
 - Например, если в выражении используется переменная, то она должна быть определена ранее по тексту программы, а из этого определения может быть получен ее тип.
 - Исходя из типа переменной, можно говорить о допустимости операции с данной переменной

Синтаксические элементы языка программирования

- Набор символов – буквы и цифры, специальные символы.
- Идентификаторы – имена объектов языка (состоит из последовательности букв и цифр, которая всегда начинается с буквы)
- Чувствительность идентификаторов к регистру (заглавные-строчные)
- Символы (знаки) операций, ограничители и скобки
- Ключевые слова

Синтаксические элементы языка программирования

- Выражения – конструкции языка для вычисления и изменения значений (включают символы операций и операнды)
 - Окончание выражения
- Операторы – конструкции языка, необходимые для управления порядком действий по обработке информации
- Комментарии – пояснения, включаемые в текст программы, но не обрабатываемые компилятором

Обобщенная структура программы

- Конкретная структура программы определяется языком программирования.
- В общем виде программа включает следующие компоненты:
 - список модулей, используемых в программе
 - раздел описания констант
 - раздел описания типов
 - раздел описания переменных
 - описание функций
 - основной блок программы, включающий операторы

Области применения Python

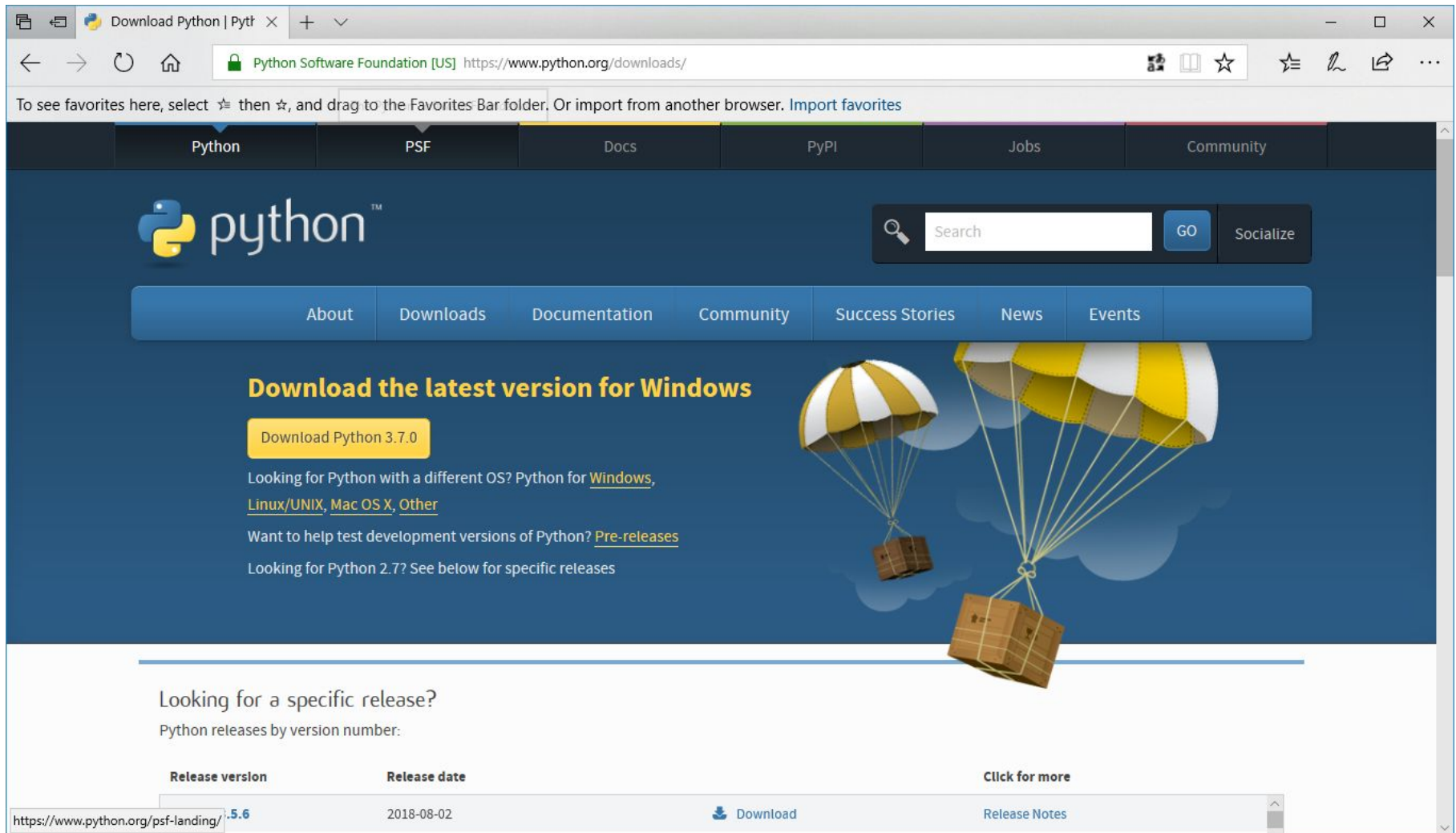


Источники информации

- <https://docs.python.org/3/index.html>
- <https://pythoner.name/documentation> (перевод документации)
- <https://ipython.org/ipython-doc/3/index.html>
- [Справочник по стандартной библиотеке языка Python](#)
- <https://pythonworld.ru/samouchitel-python>
- https://www.bestprog.net/ru/sitemap_ru/python-ru/
- [Python Module of the Week](#)
- <https://python-scripts.com/>

Установка Python

<http://www.python.org>



The screenshot shows the Python.org website's download page. The browser's address bar displays the URL <https://www.python.org/downloads/>. The page features a dark blue header with the Python logo and a navigation menu. Below the header, there's a section titled "Download the latest version for Windows" with a prominent yellow button labeled "Download Python 3.7.0". To the right of this section, there's an illustration of two parachutes carrying boxes. Below the main download section, there's a link to "Download Python 3.7.0" and a link to "Python for Windows, Linux/UNIX, Mac OS X, Other". Further down, there's a section for "Pre-releases" and a link to "Python 2.7?". At the bottom, there's a table of Python releases by version number.

Python Software Foundation [US] <https://www.python.org/downloads/>

To see favorites here, select ☆ then ☆, and drag to the Favorites Bar folder. Or import from another browser. [Import favorites](#)

Python PSF Docs PyPI Jobs Community

python™

Search GO Socialize

About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

Download Python 3.7.0

Looking for Python with a different OS? Python for [Windows](#), [Linux/UNIX](#), [Mac OS X](#), [Other](#)

Want to help test development versions of Python? [Pre-releases](#)

Looking for Python 2.7? See below for specific releases

Looking for a specific release?
Python releases by version number:

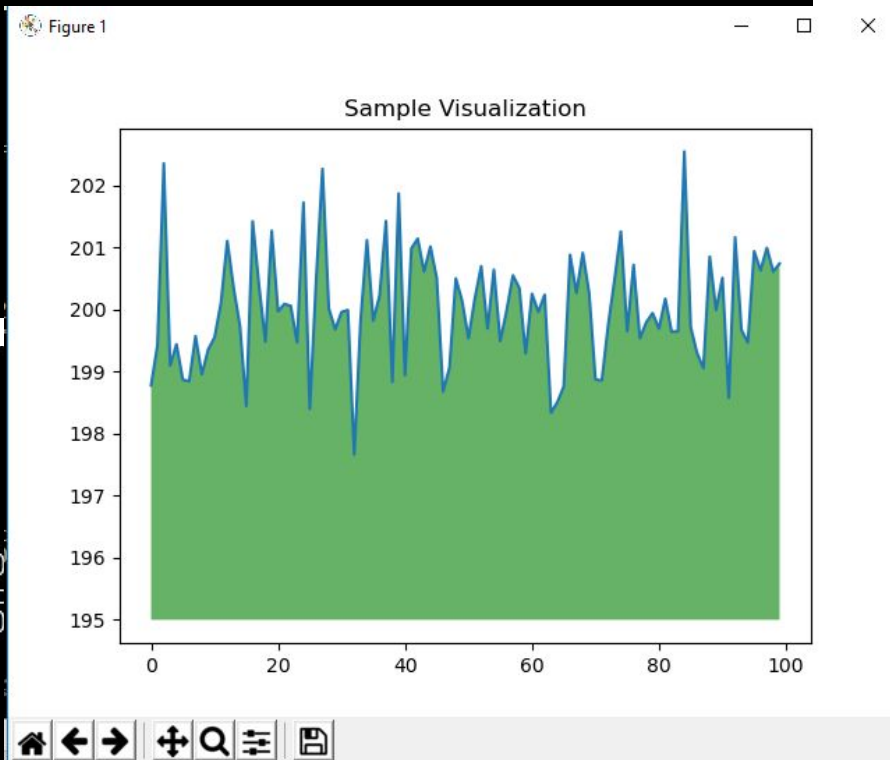
Release version	Release date	Click for more
https://www.python.org/psf-landing/ 3.6	2018-08-02	Download Release Notes

Запуск и выполнение инструкций Python

C:\WINDOWS\system32\cmd.exe - python

```
Microsoft Windows [Version 10.0.14393]
(c) Корпорация Майкрософт (Microsoft Corporation), 2016. Все права
C:\Users\niko>python
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1
Type "help", "copyright", "credits" or "license" for more informat
>>> a = 4
>>> b = 7
>>> c = a + b
>>> print("c = ", c)
c = 11
>>> c
11
>>>
```

```
>>> import numpy as np
>>> from matplotlib import pyplot as plt
>>>
>>> ys = 200 + np.random.randn(100)
>>> x = [x for x in range(len(ys))]
>>>
>>> plt.plot(x, ys, '-')
[<matplotlib.lines.Line2D object at 0x0000001E73DADA070>]
>>> plt.fill_between(x, ys, 195, where=(ys > 195), fac
<matplotlib.collections.PolyCollection object at 0x000
>>>
>>> plt.title("Sample Visualization")
Text(0.5, 1.0, 'Sample Visualization')
>>> plt.show()
```



IPython

IP[y]: IPython
Interactive Computing

- **IPython** – это интерактивная оболочка для с широким набором возможностей и ядро для Jupyter.
- **Jupyter notebook** является графической веб-оболочкой для IPython, которая расширяет идею консольного подхода к интерактивным вычислениям.

C:\> IPython: C:\Users\niko

```
Microsoft Windows [Version 10.0.14393]
(c) Корпорация Майкрософт (Microsoft Corporation), 2016.

C:\Users\niko>pip install ipython
Collecting ipython
  Downloading ipython-7.15.0-py3-none-any.whl (783 kB)
    | 783 kB 1.1 MB/s
```


IPython

IP[y]: IPython Interactive Computing

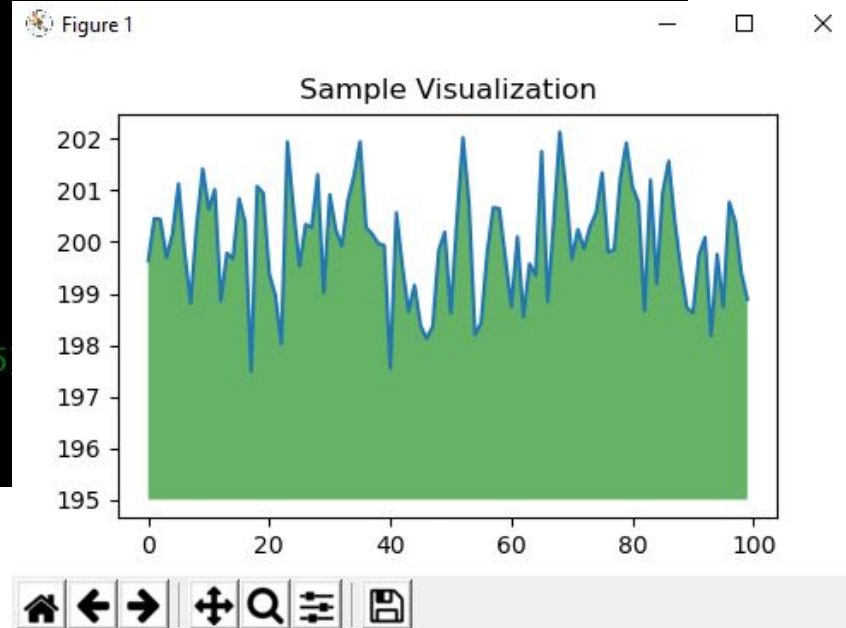
```
C:\Users\niko>ipython
Python 3.8.3 (tags/v3.8.3:6f8c832, May 13 2020, 22:37:02) [MSC v.1924 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 7.15.0 -- An enhanced Interactive Python. Type '?' for help.
```

```
In [1]: # Using Integers
...: int1 = 5
...: int2 = 5+5
...: int3 = int1 + int2
...: print("Value of int3 is: ", int3)
...: int3
```

Value of int3 is: 15

Out[1]: 15

```
In [2]: import numpy as np
...: from matplotlib import pyplot as plt
...:
...: ys = 200 + np.random.randn(100)
...: x = [x for x in range(len(ys))]
...:
...: plt.plot(x, ys, '-')
...: plt.fill_between(x, ys, 195, where=(ys > 195))
...:
...: plt.title("Sample Visualization")
...: plt.show()
```



Anaconda

[Products](#)[Why Anaconda?](#)[Solutions](#)[Resources](#)[Company](#)[Download](#)

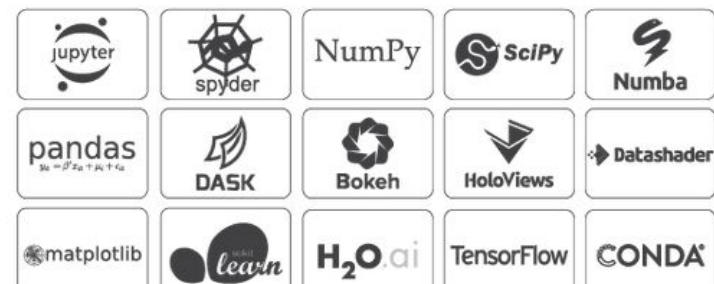
Anaconda Distribution

The World's Most Popular Python/R Data Science Platform

[Download](#)

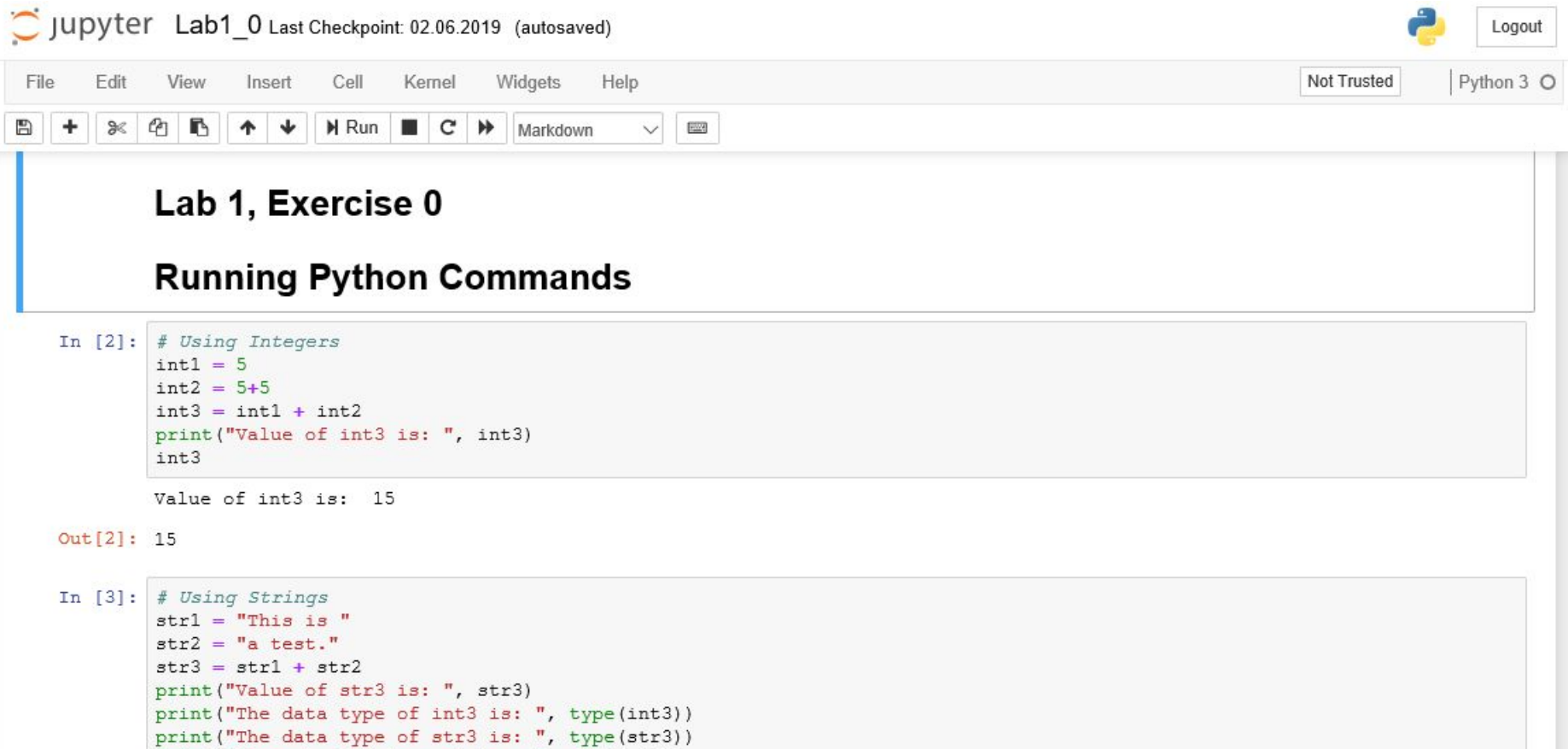
The open-source [Anaconda Distribution](#) is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 11 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

- Quickly download 1,500+ Python/R data science packages
- Manage libraries, dependencies, and environments with [Conda](#)
- Develop and train machine learning and deep learning models with [scikit-learn](#), [TensorFlow](#), and [Theano](#)




Jupyter Notebook

Как настроить и работать с Jupyter Notebook



The screenshot displays the Jupyter Notebook web interface. At the top, the header shows 'jupyter Lab1_0' and 'Last Checkpoint: 02.06.2019 (autosaved)'. On the right, there is a Python logo and a 'Logout' button. Below the header is a menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. To the right of the menu bar are 'Not Trusted' and 'Python 3' indicators. A toolbar with various icons for file operations and execution is located below the menu bar. The main content area has a title 'Lab 1, Exercise 0' and a subtitle 'Running Python Commands'. It contains two code cells. The first cell, labeled 'In [2]:', contains Python code for using integers and prints the value of 'int3' as 15. The second cell, labeled 'In [3]:', contains Python code for using strings and prints the data types of 'int3' and 'str3'.

jupyter Lab1_0 Last Checkpoint: 02.06.2019 (autosaved)  Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Save Add Split Cell Run Stop Restart Markdown

Lab 1, Exercise 0

Running Python Commands

```
In [2]: # Using Integers
int1 = 5
int2 = 5+5
int3 = int1 + int2
print("Value of int3 is: ", int3)
int3

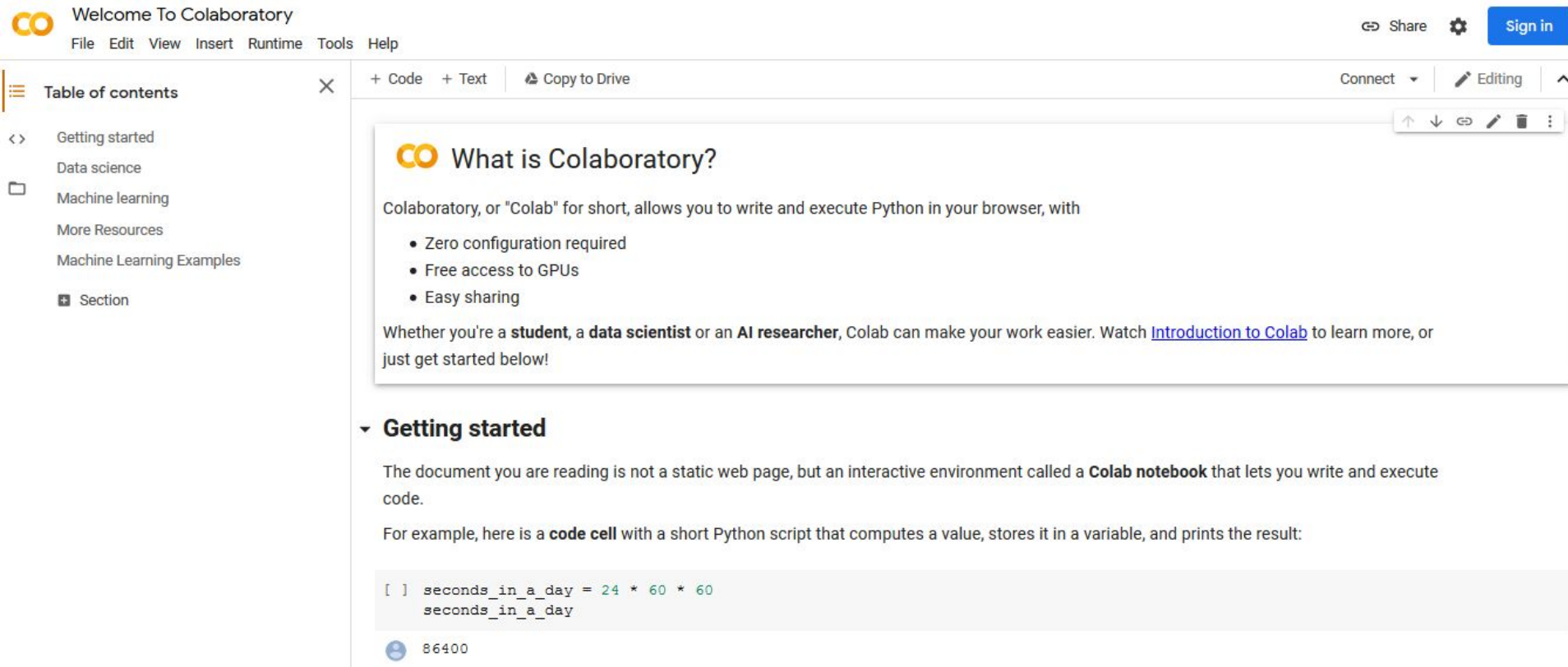
Value of int3 is:  15

Out[2]: 15

In [3]: # Using Strings
str1 = "This is "
str2 = "a test."
str3 = str1 + str2
print("Value of str3 is: ", str3)
print("The data type of int3 is: ", type(int3))
print("The data type of str3 is: ", type(str3))
```

Jupyter Notebook онлайн (Internet)

Google Colaboratory



The screenshot displays the Google Colaboratory web interface. At the top, there's a header with the Colab logo, 'Welcome To Colaboratory', and navigation links: File, Edit, View, Insert, Runtime, Tools, Help. On the right, there are 'Share', 'Settings', and 'Sign in' buttons. A left sidebar contains a 'Table of contents' with links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Machine Learning Examples'. The main content area is titled 'What is Colaboratory?' and explains that Colab allows writing and executing Python in a browser. It lists three benefits: zero configuration required, free access to GPUs, and easy sharing. Below this, it mentions that Colab is useful for students, data scientists, and AI researchers, and provides a link to 'Introduction to Colab'. A section titled 'Getting started' follows, stating that the document is an interactive 'Colab notebook' and provides an example of a code cell with a Python script to calculate the number of seconds in a day. The code is:

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

 The output of the code is 86400.

Welcome To Colaboratory

File Edit View Insert Runtime Tools Help

Share Settings Sign in

Table of contents

- Getting started
- Data science
- Machine learning
- More Resources
- Machine Learning Examples
- Section

+ Code + Text Copy to Drive

Connect Editing

What is Colaboratory?

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

- Zero configuration required
- Free access to GPUs
- 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

Стартовое ОКНО

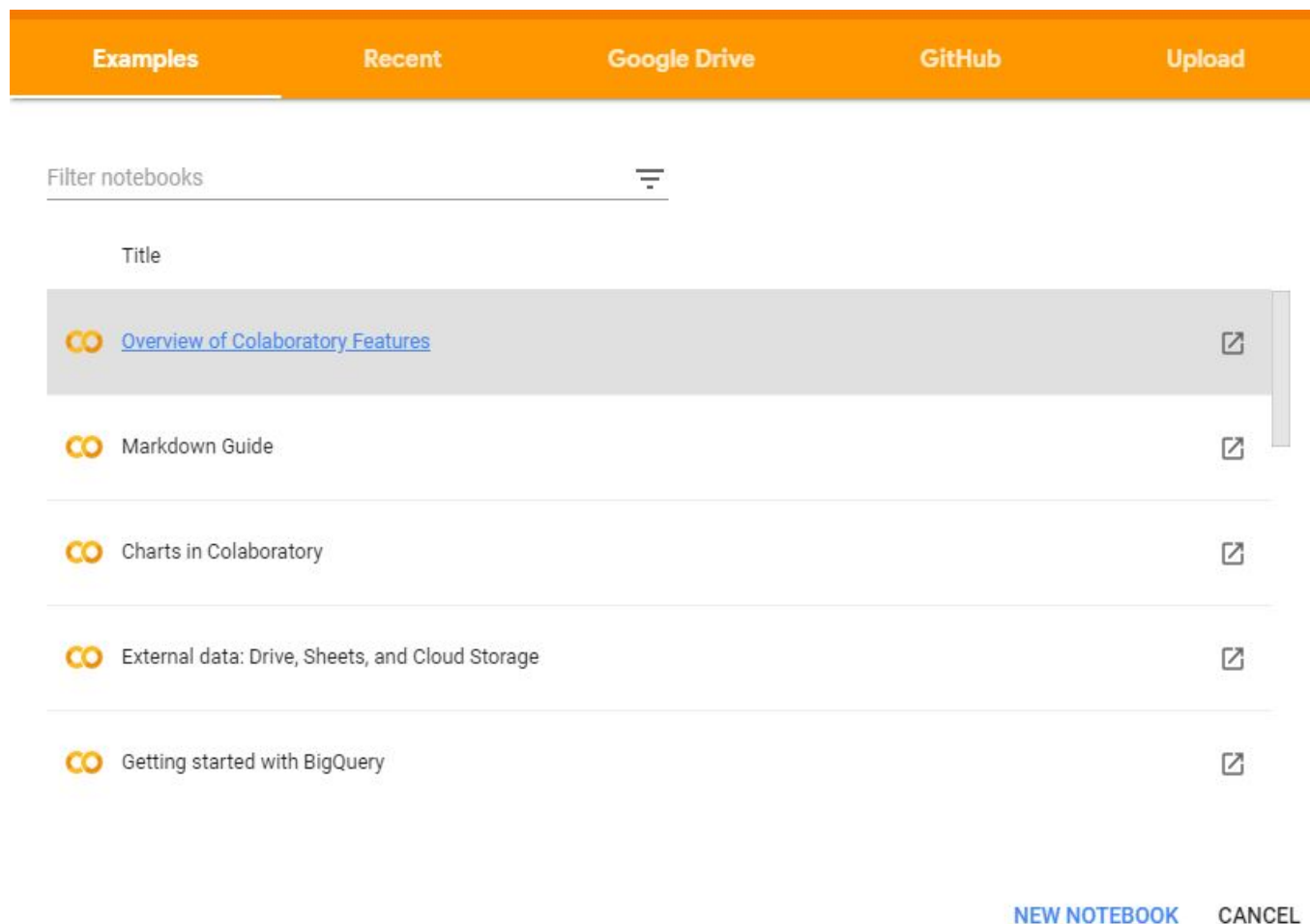
EXAMPLES: Contain a number of Jupyter notebooks of various examples.

RECENT: Jupyter notebook you have recently worked with.

GOOGLE DRIVE: Jupyter notebook in your google drive.

GITHUB: You can add Jupyter notebook from your GitHub but you first need to connect Colab with GitHub.

UPLOAD: Upload from your local directory.



The screenshot displays the Google Colaboratory interface. At the top, there is a navigation bar with five tabs: 'Examples', 'Recent', 'Google Drive', 'GitHub', and 'Upload'. The 'Examples' tab is currently selected. Below the navigation bar, there is a search bar labeled 'Filter notebooks' and a hamburger menu icon. A list of notebooks is shown below, each with a Colab icon, a title, and an external link icon. The notebooks listed are:

Title
Overview of Colaboratory Features
Markdown Guide
Charts in Colaboratory
External data: Drive, Sheets, and Cloud Storage
Getting started with BigQuery

At the bottom right of the interface, there are two buttons: 'NEW NOTEBOOK' and 'CANCEL'.

Пример



Untitled0.ipynb ☆

File Edit View Insert Runtime Tools Help [All changes saved](#)

Comment

Share



+ Code + Text



RAM
Disk



Editing



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

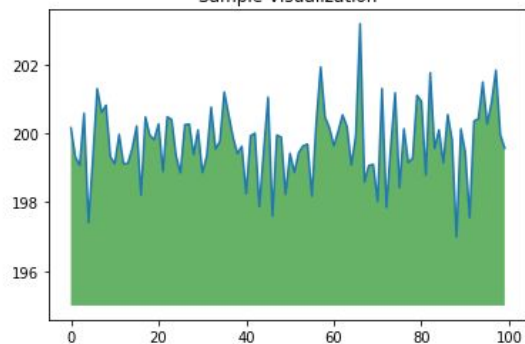
ys = 200 + np.random.randn(100)
x = [x for x in range(len(ys))]

plt.plot(x, ys, '-')
plt.fill_between(x, ys, 195, where=(ys > 195), facecolor='g', alpha=0.6)

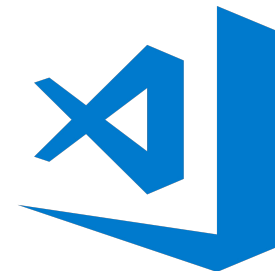
plt.title("Sample Visualization")
plt.show()
```



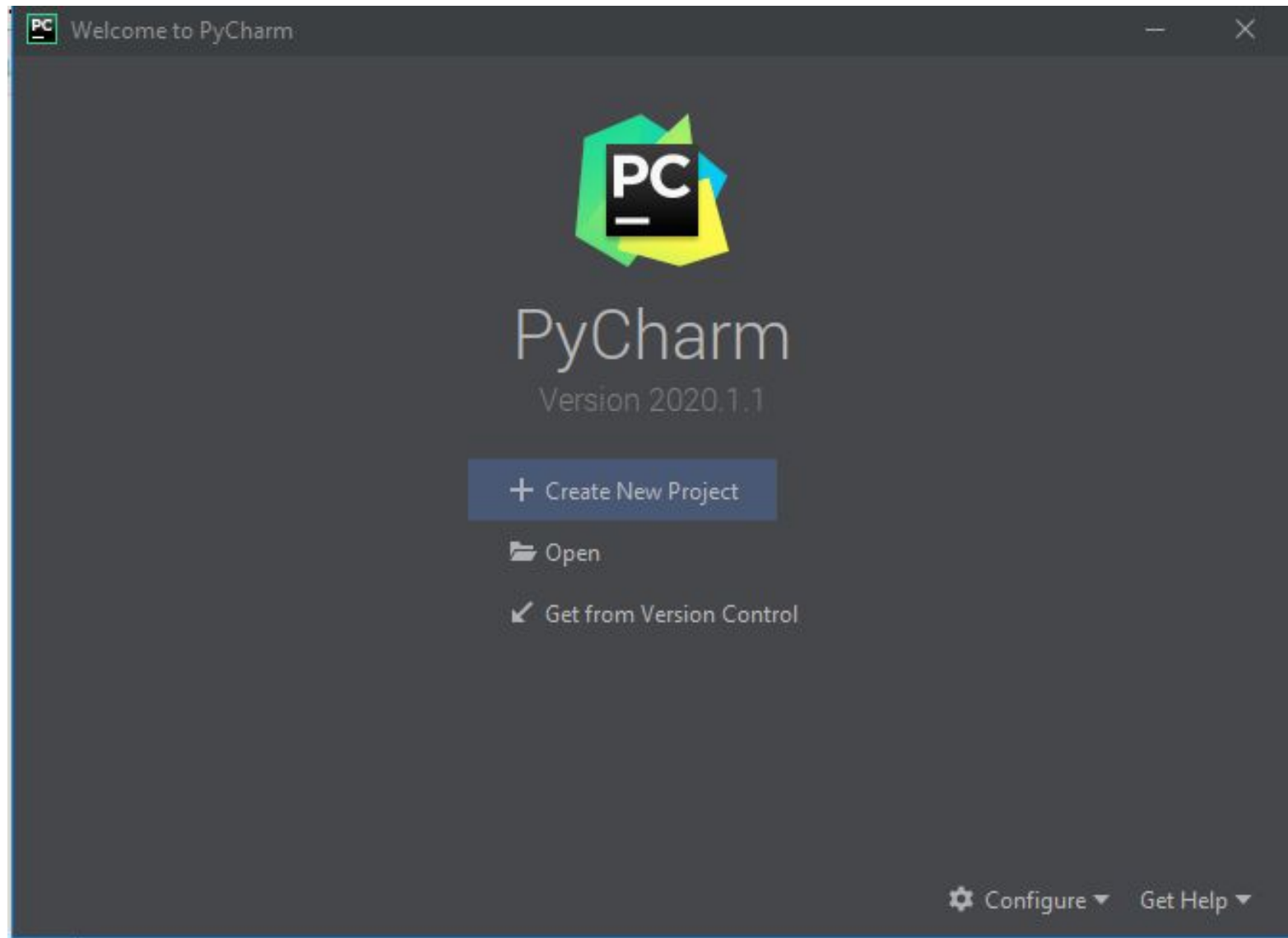
Sample Visualization

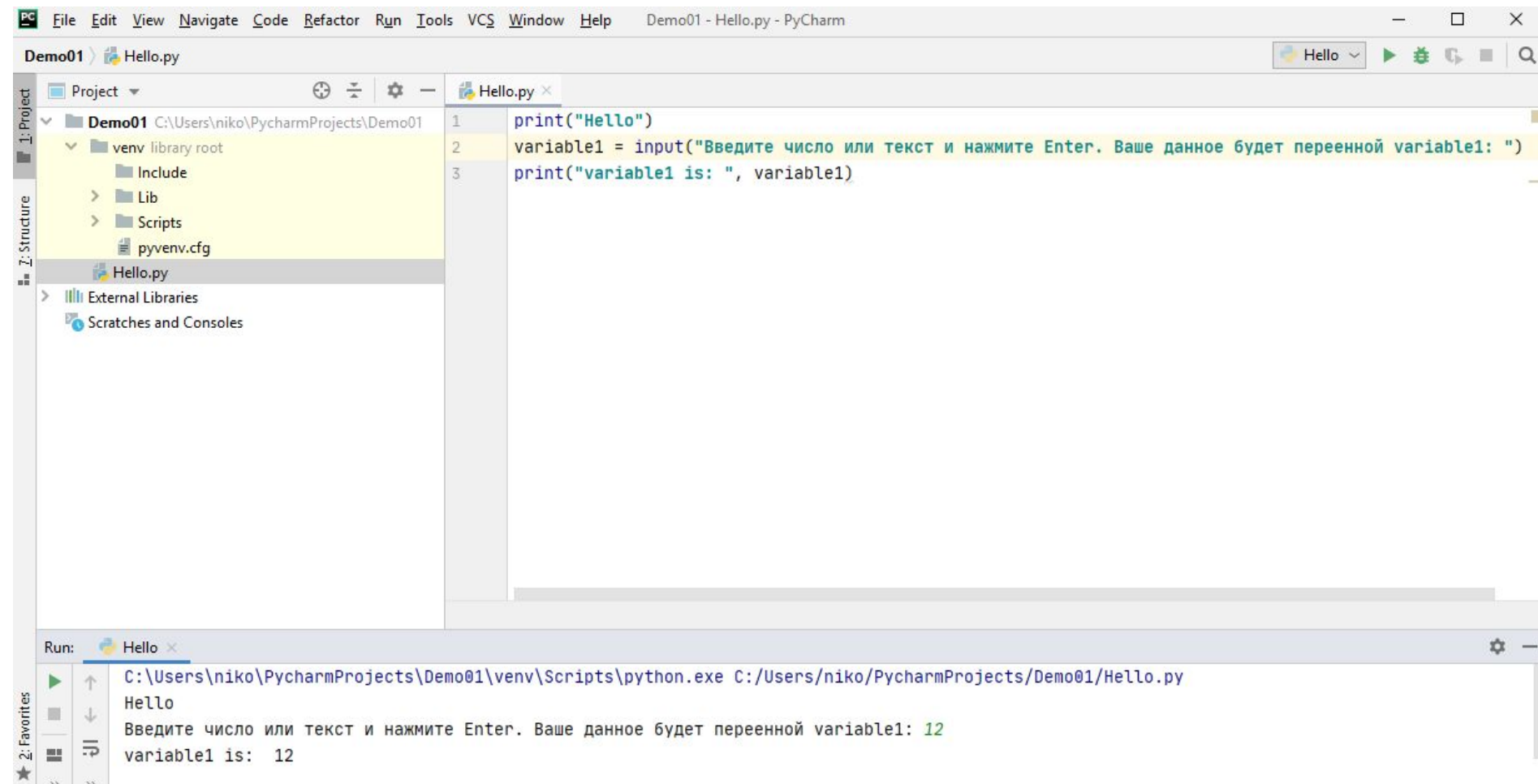


Выбор IDE



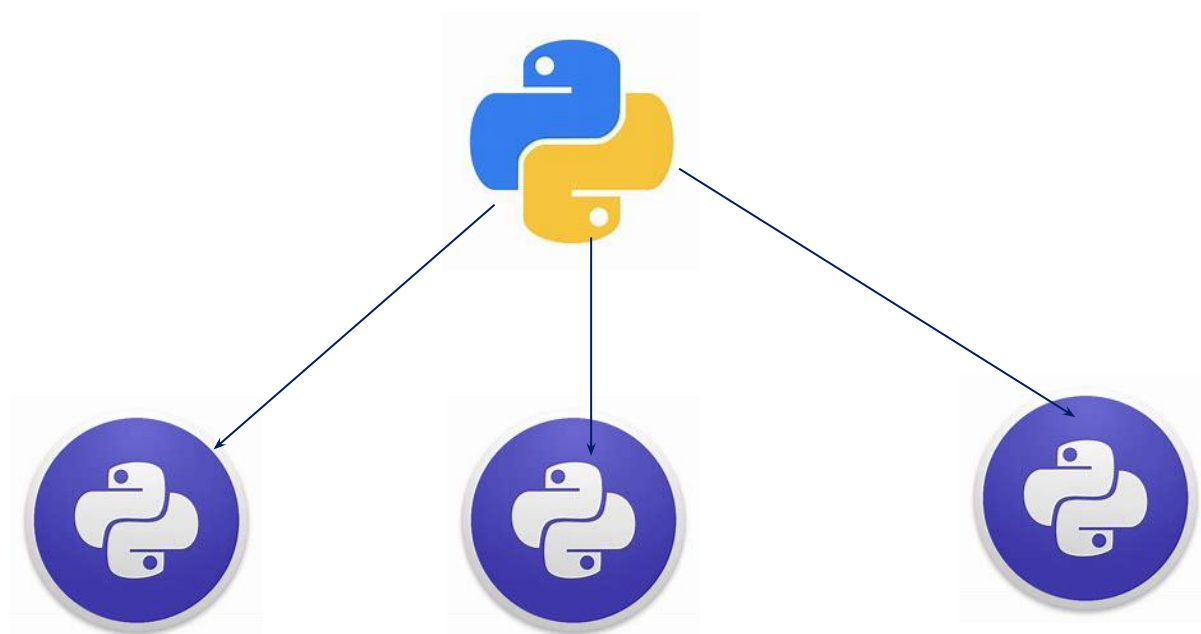
JetBrains PyCharm





Virtualizing The Python Environment

Виртуальное окружение Python

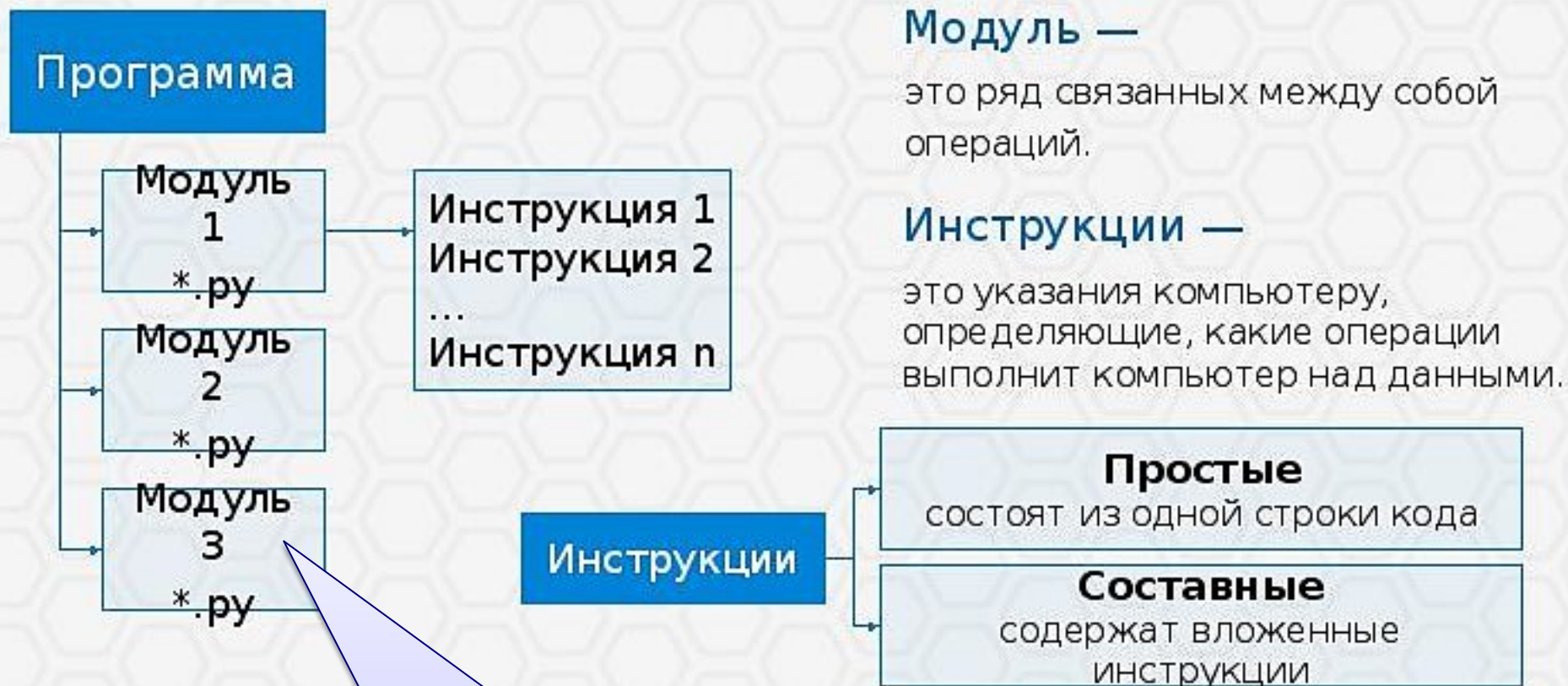


```
D:\PythonProject\my-venv\Scripts\python.exe C:/Users/niko/PycharmProjects/Demo02/dchart.py
```

```
C:\Users\niko\PycharmProjects\Demo03\venv\Scripts\python.exe C:/Users/niko/PycharmProjects/Demo04/dchart04.py
```

<https://docs.python.org/3/tutorial/venv.html>

Структура программы на Python



В любом модуле могут быть объявлены функции

Структура программы (модуля) на Python

Интерфейс ввода

- Реализация диалога с пользователем – ввод исходных данных

Реализация логики

- Реализация алгоритмов выбора, циклы и т.д.

Интерфейс вывода

- Реализация вывода результирующих данных

Структура программы (модуля) на Python

Объявление функций

- Объявление функций, описывающих реализацию алгоритмов

Интерфейс ввода

- Реализация диалога с пользователем – ввод исходных данных

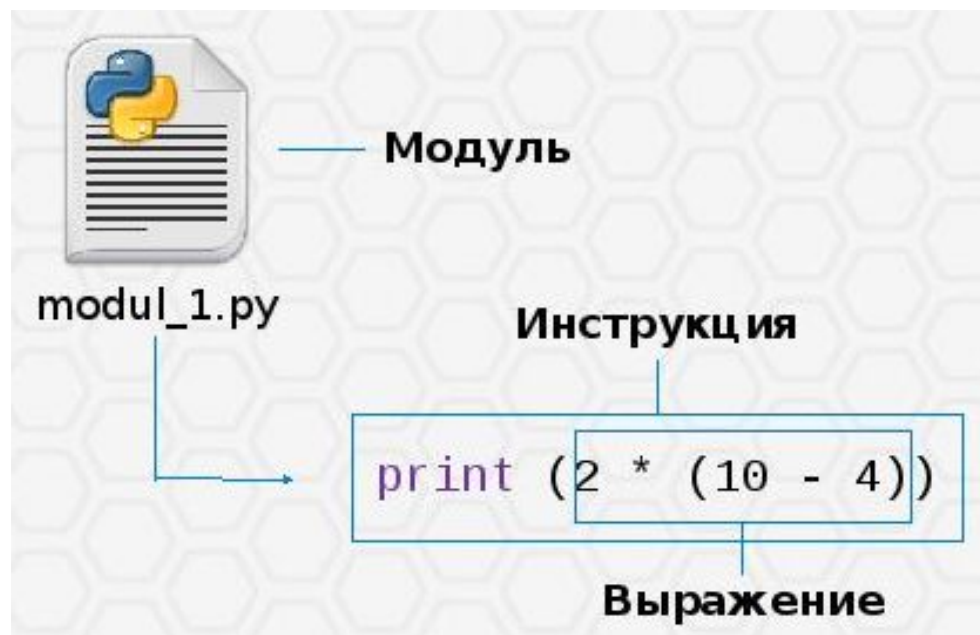
Реализация логики

- **Вызов функций**, реализация алгоритмов выбора, циклы и т.д.

Интерфейс вывода

- Реализация вывода результирующих данных

Структура инструкции на Python



Операции —

это любые действия над операндами.

Операнды —

это некоторые данные.



Пример.

Задача. Вычислить общую стоимость товара

Алгоритм на русском языке:

1. Получить цену одной единицы товара
2. Получить количество товара
3. Вычислить общую стоимость, умножив цену на количество
3. Вывести результат на экран



Как реализовать модель данных?



Как реализовать ввод, вычисление и вывод данных?

Что такое переменная?

Для реализации
модели данных

- **Переменная** – это математическое понятие, математическая величина:
 - под переменной x понимают каждый элемент некоторого множества, состоящего, например, из вещественных чисел. Фиксированный элемент этого множества называется значением переменной.
 - в математике переменной может быть, как реальная измеримая физическая величина, так и некая абстрактная величина, прямо не связанная с процессами реального мира
- В программировании **переменная это область памяти**, имеющая физические характеристики в виде длины, которую нужно
 - выделить
 - и поместить в нее определенным образом закодированные данные, отвечающие представлению объектной области

Имена переменных

МОЖНО использовать

- латинские буквы (A-Z, a-z)

заглавные и строчные буквы **различаются**

- русские буквы (**не рекомендуется!**)
- цифры

имя не может начинаться с цифры

- знак подчеркивания _

НЕЛЬЗЯ использовать

~~• скобки~~

~~• знаки +, =, !, ? и др.~~

Какие имена правильные?

**AXby R&B 4Wheel Вася “PesBarbos”
TU154 [QuQu] _ABBA A+B**

Типы переменных

- `int` # целое
- `float` # вещественное
- `bool` # логический
- `str` # строка

Для реализации
модели данных

Все типы являются
объектами

```
a = 5
print ( type(a) )
a = 4.5
print ( type(a) )
a = True
print ( type(a) )
a = "Вася"
print ( type(a) )
```

`<class 'int'>`

`<class 'float'>`

`<class 'bool'>`

`<class 'str'>`



Зачем нужен тип переменной?

Тип определяет (из лекции про типизацию):

- область допустимых значений
- допустимые операции
- объём памяти
- формат хранения данных

Как записать значение в переменную?

Объявить
переменную!



При записи нового значения
тип данных определяется в
момент присваивания

Оператор
присваивания

`a = 5`

`a` → 5

Переменная `a` ссылается на область памяти, в которой
хранится целое число 5

Оператор – это команда языка
программирования (инструкция)

Оператор присваивания – это команда для
записи нового значения переменной

Размещение переменных в памяти

оператор
присваивания

`a = 5`

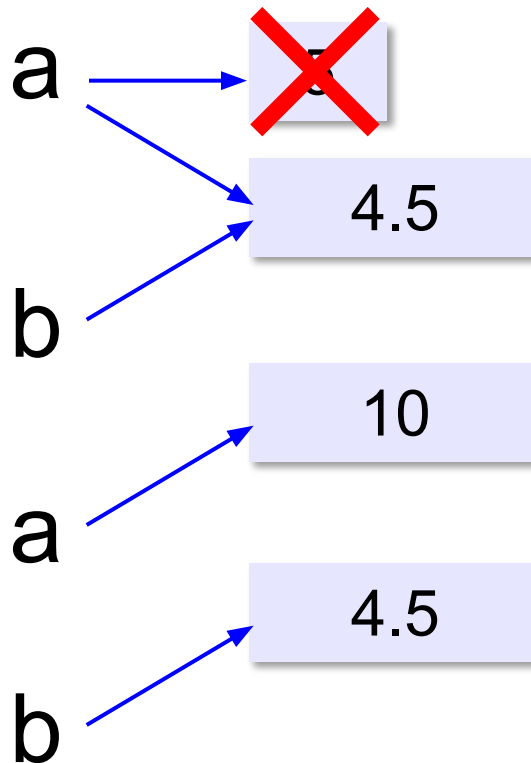
`a = 4.5`

`b = a`

`a = 10`



При записи нового значения
старое удаляется из памяти!



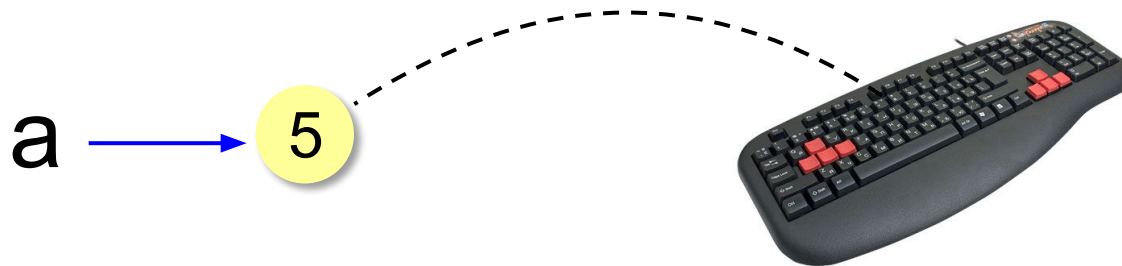
«сборщик
мусора»

Особенности Python – Безопасность доступа к памяти

- Python – язык со строгой динамической типизацией
- Все с чем работает программист – объекты
- Python - язык с встроенным менеджером управления памятью и выполняет операции очистки памяти автоматически за счет наличия сборщика мусора (Garbage Collection, GC)
- Алгоритм, используемый сборщиком мусора называется подсчетом ссылок (Reference Counting).
- Python хранит журнал ссылок на каждый объект и автоматически уничтожает объект, как только на него больше нет ссылок

Ввод исходных значений

❑ С клавиатуры



1. Программа ждет, пока пользователь введет значение и нажмет *Enter*.
2. Введенное значение записывается в переменную **a** (связывается с именем **a**)

❑ Из файла (будет рассматриваться позднее)

Комментарий

```
# Это важная программа
```

? Что делает эта программа?

комментарии после #
не обрабатываются

кодировка utf-8
по умолчанию)

```
# -*- coding: utf-8 -*-
```

```
# Это тоже программа
```

Windows: cp1251

```
"""
```

```
Это комментарий для  
документирования модуля и функции
```

```
"""
```

Ввод значения с клавиатуры

```
a = input()
```

Ввести строку с клавиатуры и связать с переменной `a`

```
b = input()
```

```
c = a + b
```

```
print(c)
```

Результат:

21

33

2133



Почему?



Результат функции `input` – строка символов!

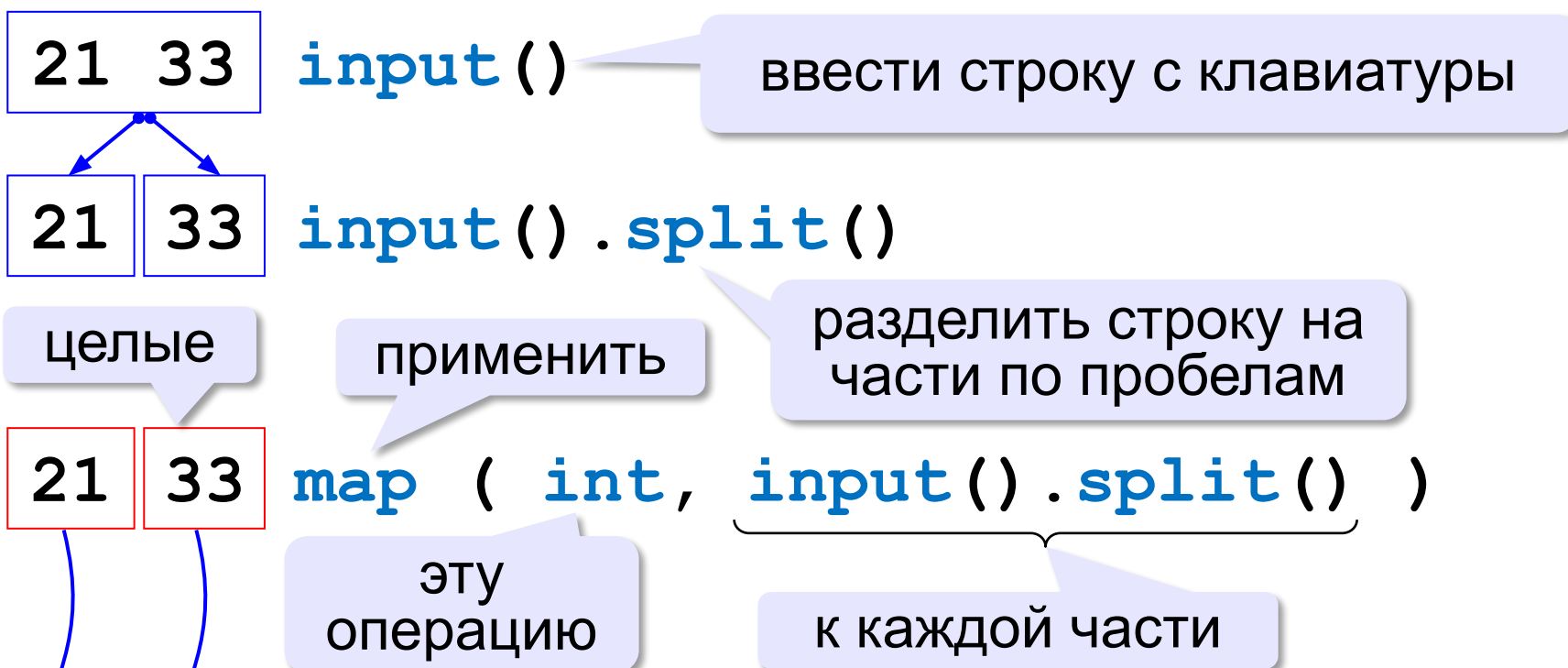
Нужно преобразовать в целое число

```
a = int(input("Введите первое число: "))
```

```
b = int(input("Введите второе число: "))
```

Ввод нескольких значений в одной строке

```
a, b = map ( int, input().split() )
```



```
a, b = map ( int, input().split() )
```

Ввод вещественных чисел

```
print( "Введите число:" )  
x = float (input())
```

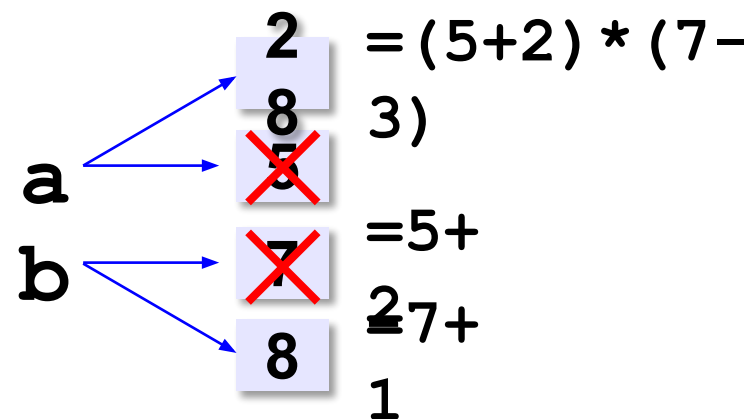
или так:

```
x = float (input("Введите число:"))
```

Изменение значений переменной

```

a = 5
b = a + 2
a = (a + 2) * (b - 3)
b = b + 1
  
```



Сложение чисел

```
print ("Введите два целых числа: ")  
a = int (input ())  
b = int (input ())  
c = a + b  
print ( a, "+", b, "=", c )
```

подсказка

Результат:

компьютер

Введите два целых числа

25

пользователь

30

25 + 30 = 55

компьютер

Арифметические выражения

$$a \leftarrow \frac{c + b - 1}{2} \cdot d$$

Линейная запись (в одну строку):

```
a = (c + b - 1) / 2 * d
```

Операции: + сложение

* умножение

/ деление

** возведение в степень ($x^2 \rightarrow \mathbf{x^{**2}}$)

Вывод данных

```
print ( a )
```

значение
переменной

```
print ( "Ответ: ", a )
```

значение и
текст

перечисление через запятую

```
print ( "Ответ: ", a+b )
```

вычисление
выражения

```
print ( a, "+", b, "=", c )
```

2 + 3 = 5

через пробелы

```
print ( a, "+", b, "=", c, sep = "" )
```

2+3=5

убрать разделители

Форматный вывод

целое

```
print("{:d}+{:d}={:d}".format(a,b,a+b))
```

```
a = 5
```

```
print("{:5d}{:5d}{:5d}".format(a, a*a, a*a*a))
```

```

_ _ _ _ 5 _ _ _ 25 _ _ 125
└─┬─┬─┬┘ └─┬─┬─┬┘ └─┬─┬┘
  5 знаков 5 знаков 5 знаков
  
```

Научный формат чисел

```
x=123456789
```

```
print("x={:e}".format(x))
```



1.234568e+008

1,234568 · 10⁸

```
x=0.0000123456789
```

```
print("x={:e}".format(x))
```



1.234568e-005

1,234568 · 10⁻⁵

Арифметическое выражения

3 1 2 4 5 6

```
a = (c + b**5*3 - 1) / 2 * d
```

Приоритет (*старшинство*):

- 1) скобки
- 2) возведение в степень **
- 3) умножение и деление
- 4) сложение и вычитание

$$a = \frac{c + b^5 \cdot 3 - 1}{2} \cdot d$$

```
a = (c + b*5*3 - 1) \
      / 2 * d
```

перенос на
следующую строку

```
a = (c + b*5*3
      - 1) / 2 * d
```

перенос внутри
скобок разрешён

Деление

Классическое деление:

```
a = 9; b = 6
x = 3 / 4    # = 0.75
x = a / b    # = 1.5
x = -3 / 4   # = -0.75
x = -a / b   # = -1.5
```

Целочисленное деление (округление «вниз»!):

```
a = 9; b = 6
x = 3 // 4    # = 0
x = a // b    # = 1
x = -3 // 4   # = -1
x = -a // b   # = -2
```

Сокращенная запись операций

`a += b # a = a + b`

`a -= b # a = a - b`

`a *= b # a = a * b`

`a /= b # a = a / b`

`a //= b # a = a // b`

`a %= b # a = a % b`

`a += 1`

увеличение на 1

Множественное присваивание:

`a = b = 0 # b = 0, a = b`

`a, b = 1, 2 # a = 1; b = 2`

Остаток от деления

% – остаток от деления

```
d = 85
```

```
b = d // 10
```

```
a = d % 10
```

```
d = a % b
```

```
d = b % a
```

Для отрицательных чисел:

```
a = -7
```

```
b = a // 2 # -4
```

```
d = a % 2 # 1
```



Как в математике!

остаток ≥ 0

$$-7 = (-4) * 2 + 1$$

Пример

// – деление нацело (остаток отбрасывается)

% – остаток от деления

175 сек соответствует -- 2 мин 55 сек




Как получить 2 и 55?

```
t = 175
```

```
m = t // 60      # 2
```

```
s = t % 60       # 55
```

Пример. Частное и остаток

 Что получится?

```
n = 123
```

```
d = n // 10
```

```
k = n % 10
```

При делении на 10 **нацело** отбрасывается последняя цифра числа

Остаток от деления на 10 – это последняя цифра числа

Операторы // и %

```
a = 1234
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10 ;
```

4

3

2

1

Сокращенная запись операций

<code>a += b</code>	<code># a = a + b</code>
<code>a -= b</code>	<code># a = a - b</code>
<code>a *= b</code>	<code># a = a * b</code>
<code>a /= b</code>	<code># a = a / b</code>
<code>a // = b</code>	<code># a = a // b</code>
<code>a %= b</code>	<code># a = a % b</code>

`a += 1`

увеличение на 1

Вещественные числа



Целая и дробная части числа разделяются точкой!

Форматы вывода:

```
x = 123.456
```

```
print( x )
```

```
print( "{:10.2f}".format(x) )
```

123.456

_____ 123.46

всего знаков

в дробной части

```
print( "{:10.2g}".format(x) )
```

_____ 1.2e+02

значащих цифр

$1,2 \cdot 10^2$

Вещественные числа

Экспоненциальный формат:

```
x = 1. / 30000
```

```
print("{:e}".format(x))
```

```
x = 12345678.
```

```
print("{:e}".format(x))
```

$3,333333 \cdot 10^{-5}$

3.333333e-05

$1,234568 \cdot 10^7$

1.234568e+07

```
x = 123.456
```

```
print("{:e}".format(x))
```

```
print("{:10.2e}".format(x))
```

1.234560e+02

__1.23e+02

всего знаков

в дробной части

Операции с вещественными числами

int – целая часть числа

```
x=1.6  
print(int(x))
```



1

round – ближайшее целое число

```
x=-1.2  
print(round(x))
```



-1

Стандартные функции

abs(x) — модуль числа

int(x) — преобразование к целому числу

round(x) — округление

```
x = abs ( -1.6 )      # 1.6
```

```
x = int ( -1.6 )      # -1
```

```
x = round ( -1.6 )    # -2
```

bin(x) — в двоичную систему

oct(x) — в восьмеричную систему

hex(x) — в шестнадцатеричную систему

```
x = bin ( 29 )        # '0b11101'
```

```
x = oct ( 29 )        # '0o35'
```

```
x = hex ( 29 )        # '0x1d'
```

Математические функции

```
import math
```

ПОДКЛЮЧИТЬ
МАТЕМАТИЧЕСКИЙ МОДУЛЬ

<code>math.pi</code>	—	число «пи»
<code>math.sqrt(x)</code>	—	квадратный корень
<code>math.sin(x)</code>	—	синус угла, заданного в радианах
<code>math.cos(x)</code>	—	косинус угла, заданного в радианах
<code>math.exp(x)</code>	—	экспонента e^x
<code>math.ln(x)</code>	—	натуральный логарифм
<code>math.floor(x)</code>	—	округление «вниз»
<code>math.ceil(x)</code>	—	округление «вверх»

```
x = math.floor(1.6) # 1  
x = math.ceil(1.6)  # 2
```

```
x = math.floor(-1.6) #-2  
x = math.ceil(-1.6)  #-1
```



Практическое занятие