Python.
Введение.

Язык Python разработал голландский программист Гвидо Ван Россум (Guido van Rossum) в 1991 году. Не подумайте, что язык назван в честь змеи-питона: Гвидо был большим фанатом британского комедийного сериала «Летающий цирк Монти Пайтона» (англ. Monty Python's Flying Circus), и именно оттуда пришло название языка. С тех пор данный язык проделал большой путь развития. В 2000 году была издана версия 2.0, а в 2008 году - версия 3.0.

В настоящее время в русском языке для обозначения используют два варианта — «Питон» и «Пайтон».

Python относится к интерпретируемым языкам программирования: чтобы запустить написанную на Python программу, нужен интерпретатор Python (его можно скачать с официального сайта.)

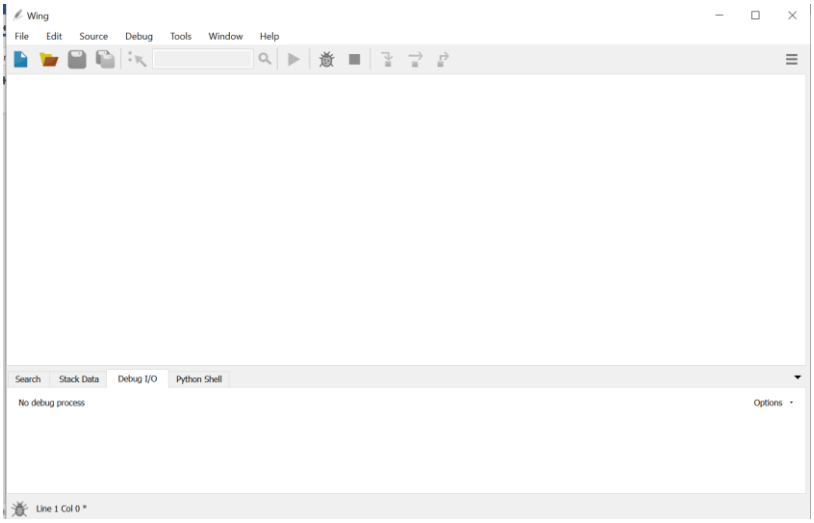
*При установке обязательно поставьте галочку **Add Python 3.8 to PATH**, что позволит вам пользоваться командой python в командной строке.*

Среда разработки.

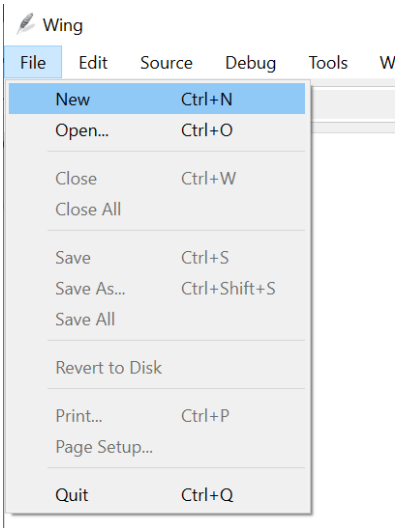
Команды для интерпретатора можно писать в обычном текстовом редакторе (например, в «Блокноте»). Но чаще для этого пользуются специальной программой, которая называется средой разработки (англ.IDE, Integrated Development Environment). Среда разработки — тоже текстовый редактор, но с дополнительными возможностями. Например, она умеет сама находить на компьютере программу-интерпретатор и запускать одной кнопкой. Среда разработки, кроме того, форматирует написанный вами код, чтобы его удобно было читать, а иногда даже подсказывает, где вы допустили ошибку.

Мы будем использовать среду разработки ***Wing 101 Python IDE***. Ее можно скачать с сайта разработчика, фирмы Wingware.

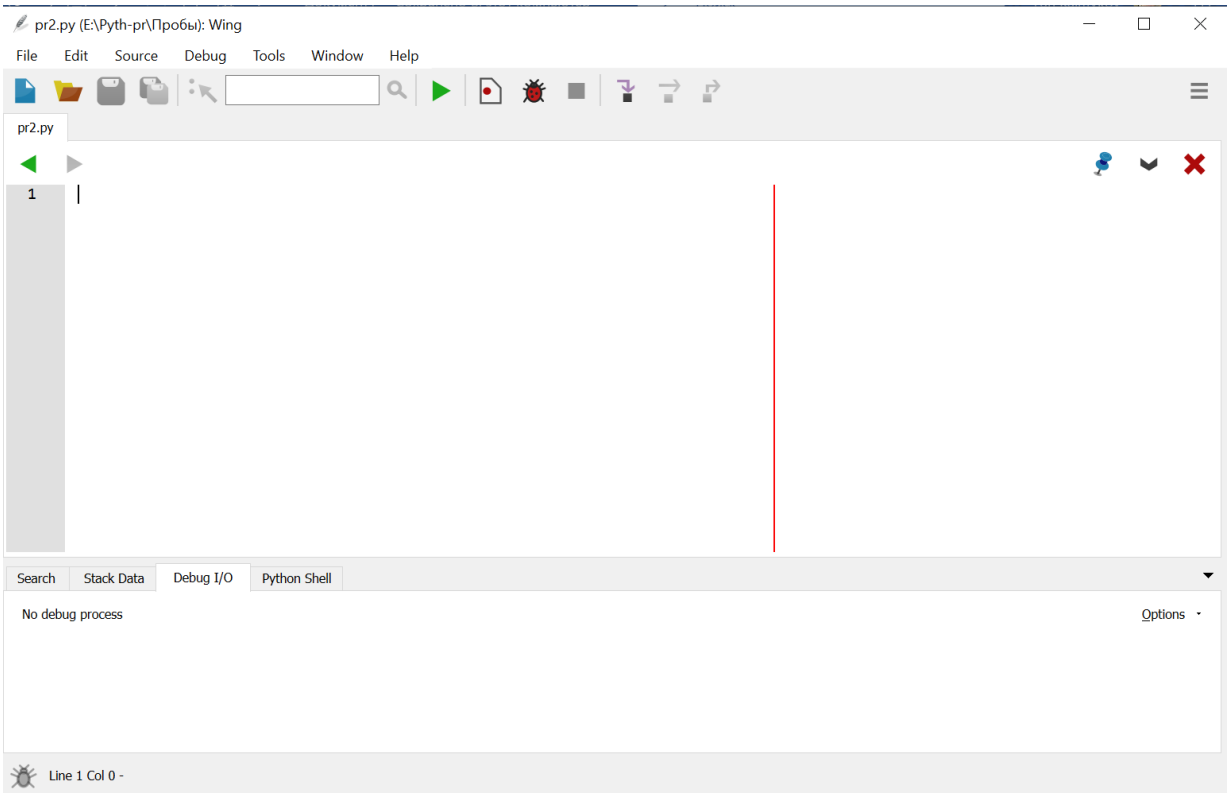
Давайте разберемся, как с ней работать. После запуска программы мы увидим следующую картинку



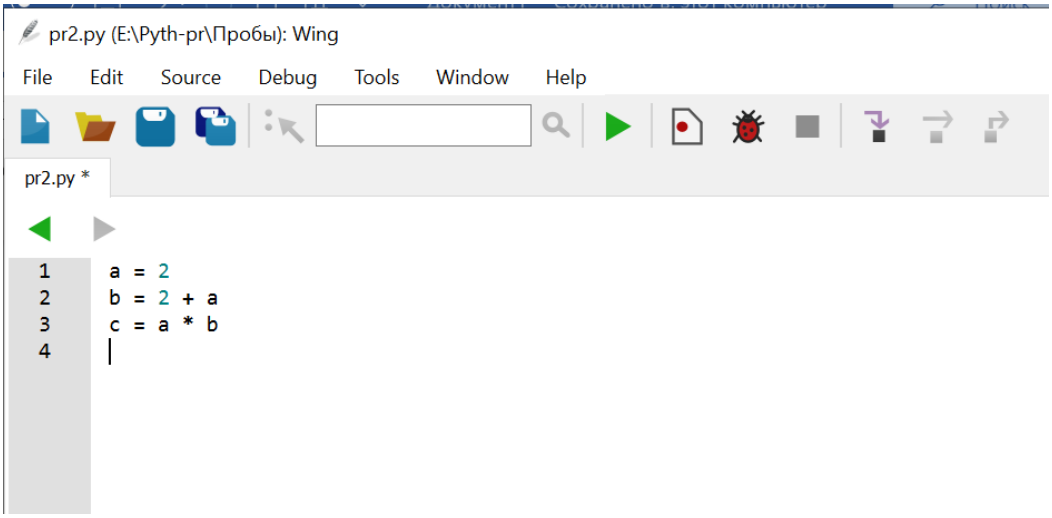
Чтобы начать писать программу, необходимо создать новый файл File - New



И сразу же его сохранить File – Save As… Тогда среда приобретет следующий вид



При наборе текста программы слева высвечиваются номера строк программы



Инструкция присваивания. Типы данных

Пусть величина **degr** задает число градусов, а переменная **rad** — число радиан. Тогда формула перевода градусов в радианы на языке Python будет выглядеть следующим образом:

rad = (degr * 3.14) / 180

Инструкция присваивания - основная в любом языке программирования. Эта инструкция позволяет присвоить переменной значение вычисленного выражения.

Инструкция присваивания записывается следующим образом. Слева стоит имя переменной, справа — выражение, значение которого должно быть вычислено. Разделяет левую и правую части символ **=**. Выполнение инструкции присваивания заключается в следующем: вычисляется результат выражения, стоящего справа, и этот результат становится значением переменной, имя которой стоит слева.

Формат инструкции присваивания:

<имя переменной> = <арифметическое выражение>

Переменная создается в момент присваивания. Тип переменной определяется автоматически в соответствии с теми данными, которые ей присваиваются.

Пример 1.

```
a = 5
b = 3.75
```

В этом примере переменная ***a*** будет целого типа, а переменная ***b*** - вещественного.

Существует возможность множественного присваивания

Пример 2.

```
a = b = c = 10 + 5
```

В этом случае все три переменные приобретут значение ***15***.

При выполнении же следующего оператора

```
a, b, c = 3 + 4, 5 * 7, 2.5
```

переменные ***a, b, c*** приобретут соответственно следующие значения: ***7 35 2.5.***

Пример 3.

```
a = 4
b = 7.3
a, b = b, a
```

В этом примере переменные обменяются значениями. Переменная ***a*** после выполнения этого фрагмента программы примет значение 7.3, а переменная ***b*** – значение 4. При этом изменится и тип переменных. На самом деле, при каждом присваивании создается новая переменная, а старая с таким же именем уничтожается. То есть в нашем примере переменные ***a*** и ***b*** имели значения 4 и 7.3 соответственно. При выполнении последнего

оператора присваивания создались новые переменные ***a*** и ***b*** со значениями 7.3 и 4 соответственно, а старые переменные уничтожились.

Имя переменной

Имя переменной в языке Python представляет собой последовательность символов, составленную по следующим правилам:

- символы, используемые в имени, являются либо латинскими буквами, либо цифрами (от 0 до 9), либо знаком подчеркивания;
- первый символ в имени может быть только буквой или знаком подчеркивания;

В Python заглавные и прописные буквы считаются разными.

Примеры имен переменных: K2, xyz, MaxMin, maxmin, t101z14, NUMBER10.

Обратите внимание, что переменные ***MaxMin*** и ***maxmin*** – это разные переменные.

Название переменной не должно совпадать с названием ключевых слов языка Python. Ключевые слова: **and, as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, in, is, lambda, None, nonlocal, not, or, pass, raise, return, True, try, while, with, yield.**

Арифметическое выражение

Арифметическое выражение — это совокупность констант, переменных, функций, объединенных знаками арифметических действий и круглыми скобками, имеющая математический смысл.

Операции, используемые в арифметических выражениях:

Знак	Операция	Тип операндов	Тип результата
+	сложение	хотя бы один из операндов *вещественный*	*вещественный*
		оба операнда *целые*	*целый*
-	вычитание	хотя бы один из операндов *вещественный*	*вещественный*
		оба операнда *целые*	*целый*
*****	умножение	хотя бы один из операндов *вещественный*	*вещественный*
		оба операнда *целые*	*целый*
/	деление	*не зависит от типа операндов*	*вещественный*
******	возведение в степень	хотя бы один из операндов *вещественный*	*вещественный*
		оба операнда *целые*	*целый*

Вывод информации на экран

Для вывода информации на экран служит функция вывода ***print()***.

Формат функции вывода:

print(<список вывода>) ;

Список вывода может содержать: текст, заключенный в апострофы или кавычки, переменные, арифметические выражения, логические выражения, а также специальные параметры. В качестве разделителя в инструкции вывода используется запятая.

При выполнении функции вывода, текст, указанный в апострофах или кавычках, будет выведен на экран без изменений, а вместо переменных или выражений будет выведены их значения. Выводимые значения отделяются друг от друга пробелом.

Примеры

```
a = 4
b = 7.3
print(a, b + 4, "Привет!")
```

На экран выведется

4 11.3 Привет!

```
a = 4
b = 7.3
print(a)
print(b)
```

На экран выведется

4

7.3

Для того, чтобы отделять данные при выводе не пробелом, а другими символами используется параметр ***sep***. Например,

```
a = 4
b = 7.3
print(a, b + 4, "Привет!", sep='---')
```

На экран выведется

4---11.3---Привет!

Для того, чтобы выводимые данные выводились без разделителей вообще, необходимо параметру ***sep*** присвоить пустую строку.

```
a = 4
b = 7.3
print(a, b + 4, "Привет!", sep='')
```

На экран выведется

411.3Привет!

Для того, чтобы в конце строки поставит определенные символы служит параметр ***end***.

```
print(a, b, sep='+', end='=')
print(a + b)
```

На экран будет выведено

4+7.3=11.3

Обратите внимание, что строка не только закончилась знаком равно, но и не произошло перевода на другую строку (после выполнения первой функции **print** курсор остался на той же строке, а следующая функция **print** продолжила вывод там же). Соответственно, если необходимо, чтобы следующая функция выводила информацию на той же строке, что и предыдущая, необходимо использовать параметр ***end***. При этом, если нет необходимости завершать строку особым символом, то параметру ***end*** необходимо присвоить пустую строку.

Комментарии

Для того, чтобы поместить комментарии в программе чаще всего используется символ решетки (#). Любой текст, помещенный после этого знака и до конца строки будет считаться комментарием.

```
# Вывод сообщения на экран
print("Добрый день!")


print("Добрый день!")  # Вывод сообщения на экран
```

Однако если необходимо закомментировать большой фрагмент текста, то удобно использовать другой способ комментирования, а именно три апострофа, размещенные на отдельной строке.

```
1  a = 4
2  b = 7.3
3  '''
4  a, b, c = 3 + 4, 5 * 7, s
5  a = b = c = 10
6  print(a, b, sep='+', end='=')
7  print(a + b)
8  '''
9  print(a, b + 4, "Привет!", sep='---')
```

В данном примере текст, расположенный на строках с 3 по 8, будет считаться комментарием.

Встроенные функции

Как и в других языках программирования, в Python имеется ряд встроенных функций. Рассмотрим некоторые из них.

Функция **int().**

Преобразует данные любого типа к целому.

Пример	Результат
`print(int(23.976))`	23
`print(int(5-7.25))`	-2
`print(int('243'))`	243
`print(int('12'+"17"))`	1217

Функция **float().**

Преобразует данные любого типа к вещественному.

Пример	Результат
print(float(5))	5.0
print(float(24*10))	240.0
print(float("1.745"))	1.745
print(float('12' + "17"))	1217.0

Функция **str().**

Преобразует данные любого типа к строковому.

Пример	Результат
print(str(24 * 10) + '!')	240!
print(str(3 + 2) + str(2 - 7))	5-5
print('4/2=' + str(4 / 2))	4/2=2.0
print(str(12 + 5) * 5)	1717171717

Обратите внимание на последний пример. Если строку умножить на некоторое целое положительное число, то строка повторится указанное число раз. При сложении строк строки склеиваются, так же, как и в других языках программирования.