

# Введение в программирование

Лицей № 134

Учитель информатики:

Арыскина Н.А.

# Введение в

## программирование

В курсе информатики 7-8 класса вы уже познакомились с понятием алгоритма и составляли программы на каком-либо языке программирования. Мы продолжим заниматься программированием, используя язык Python 3. Этот язык сейчас применяется во многих областях, в том числе для разработки веб-сайтов и решения задач искусственного интеллекта. Сначала вспомним основные сведения из курса 7-8 класса, которые нам понадобятся.

**Алгоритм – это точное описание порядка действий для некоторого исполнителя.**

Исполнителем называют человека, животное или машину, способных понимать и выполнять некоторые команды.

**Исполнитель – тот, кто выполняет команды.** Формальный исполнитель любую команду всегда выполняет одинаково, не обдумывая её.

Любой алгоритм можно составить с помощью трёх базовых конструкций: следования (последовательного выполнения команд), ветвлений (выбора одного из двух вариантов действий) и циклов (повторения одинаковых действий).

## Способы

### описания алгоритма:

1. **Словесное описание** представляет структуру алгоритма на естественном языке. Например, любой прибор бытовой техники (утюг, электропила, дрель и т.п.) имеет инструкцию по эксплуатации, т.е. словесное описание алгоритма, в соответствии которому данный прибор должен использоваться. Никаких правил составления словесного описания не существует. Запись алгоритма осуществляется в произвольной форме на естественном, например, русском языке. **Приведи свои примеры.**
2. **Псевдокод** - описание структуры алгоритма на естественном, частично формализованном языке, позволяющее выявить основные этапы решения задачи, перед точной его записью на языке программирования. В псевдокоде используются некоторые формальные конструкции и общепринятая математическая символика. Строгих синтаксических правил для записи псевдокода не существует.

**Пример программы, выводящий сообщение «[Здравствуй, Мир!](#)»**

алг ЗДРАВСТВУЙМИР

нач

вывод ('Здравствуй, Мир!')

кон алг ЗДРАВСТВУЙМИР

## Способы

**3. Блок-схема** - описание структуры алгоритма с помощью геометрических фигур с линиями-связями, показывающими порядок выполнения отдельных инструкций. Этот способ имеет ряд преимуществ. Благодаря наглядности, он обеспечивает «читаемость» алгоритма и явно отображает порядок выполнения отдельных команд. В блок-схеме каждой формальной конструкции соответствует определенная геометрическая фигура или связанная линиями совокупность фигур.



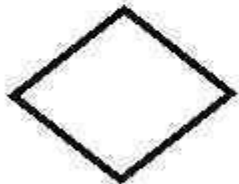
- начало и конец алгоритма



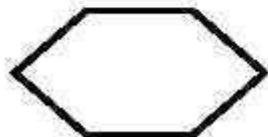
- обработка данных (операция присваивания)



- ввод/вывод данных



- решение (оператор условного перехода).



- цикл (подготовка), организация циклического процесса

## Способы

### описания алгоритма:

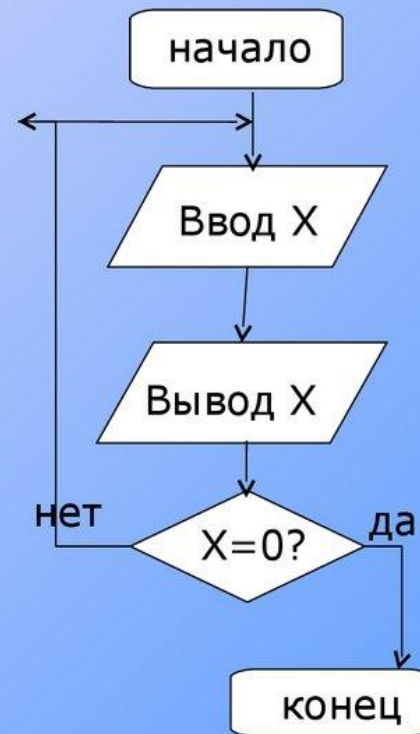
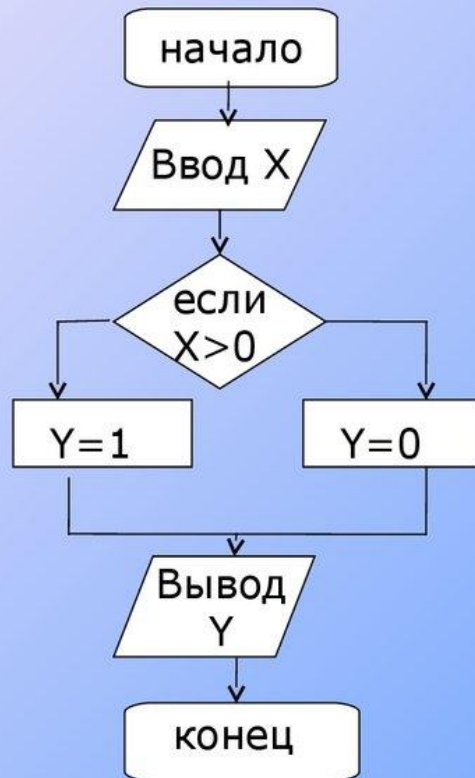
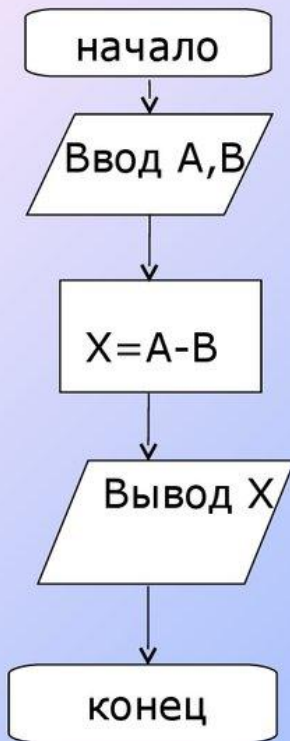
4. Запись алгоритма на языке программирования называется **программой**. **Программирование** – это создание программ для компьютеров. Людей, которые этим занимаются, называют **программистами**. Код программы зависит от языка

Паскаль	C++	Python
<pre>program E3; uses crt; var num1, num2, summa: real; begin clrscr; writeln(' введите значение num1 ='); readln(num1 ); writeln(' введите значение num2='); readln(num2); summa = num1 + num2 writeln ('summa=', summa ) end.</pre>	<pre>#include &lt;iostream&gt; using namespace std; int main() { cout &lt;&lt; "введите значение num1 ="; cin&gt;&gt;num1 ; cout &lt;&lt; "введите значение num2 ="; cin&gt;&gt;num2 ; summa = num1 + num2 cout &lt;&lt;"summa=" &lt;&lt;summa; return 0; }</pre>	<pre>num1 = int(input('введите значение num1 =')) num2 = int(input('введите значение num1 =')) summa = num1 + num2 print(' summa =', summa)</pre>

Различают следующие **виды алгоритмов**:

1. **линейный** – список команд (указаний), выполняемых последовательно друг за другом;
2. **разветвляющийся** – алгоритм, содержащий хотя бы одну проверку условия, в результате которой обеспечивается переход на один из возможных вариантов решения;
3. **циклический** – алгоритм, предусматривающий многократное повторение одной и той же последовательности действий. Количество повторений обеспечивается некоторым логическим или числовым условием.

### Примеры блок-схем



# Системы программирования

Для разработки новых программ используют инструментальные средства или системы программирования.

**Система программирования** – программная система, предназначенная для разработки программ на конкретном языке программирования.

В состав системы программирования обязательно входят транслятор и отладчик.

**1. Трансляторы** бывают двух типов:

- **компиляторы**, которые переводят в машинные коды сразу всю программу и строят исполняемый файл (в операционной системе *Windows* он имеет расширение *.exe*);

- **интерпретаторы**, которые выполняют программу по частям:

обработав очередной фрагмент программы, интерпретатор сразу исполняет его.

**2. Отладчик** – программа для поиска ошибок в разрабатываемых программах.

Отладчик позволяет:

- выполнять программу в пошаговом режиме (по одной строчке);
- просматривать значения переменных в памяти;
- устанавливать *точки останова*, то есть отмечать места в программе, в которых выполнение программы временно приостанавливается;

**Среда программирования** обычно включает редактор текста программ, транслятор и отладчик.

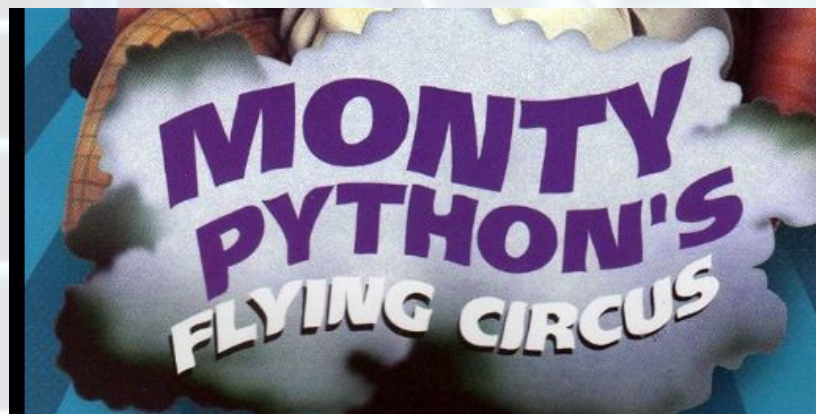
# Python.org – официальный сайт

Гвидо ван Россум

Guido van Rossum



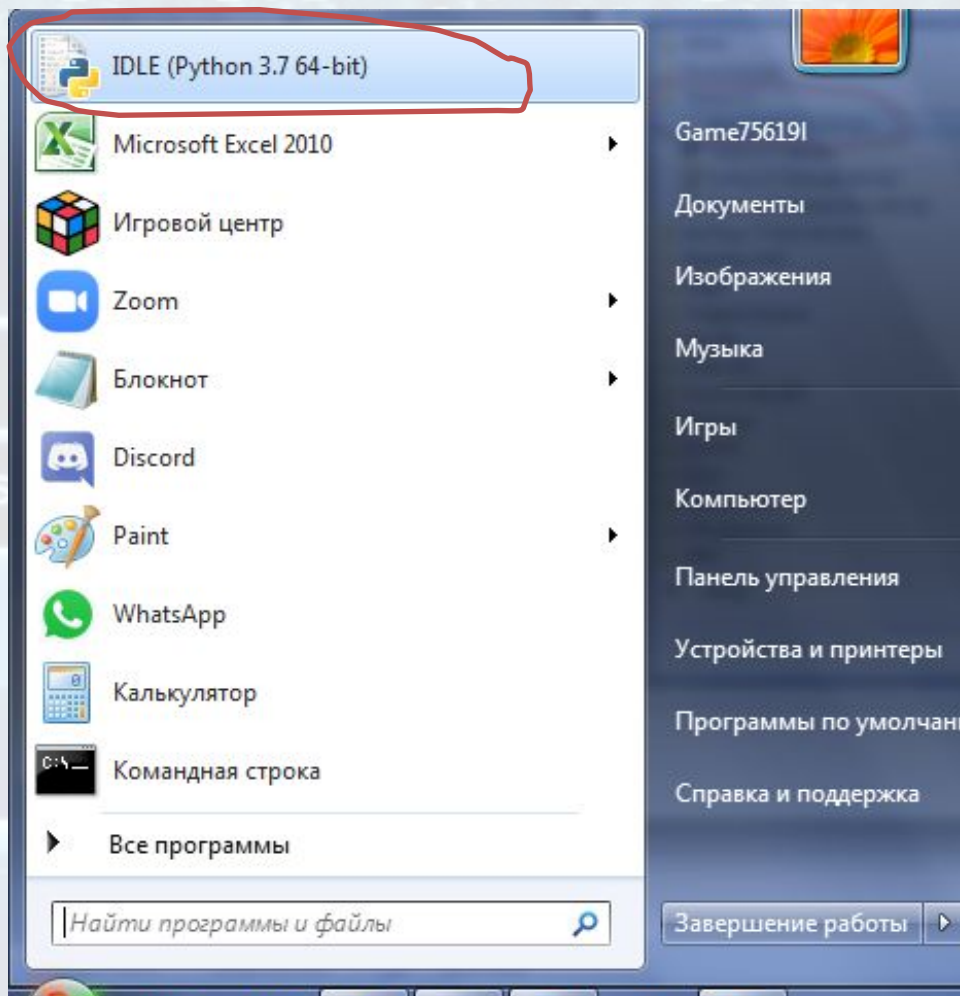
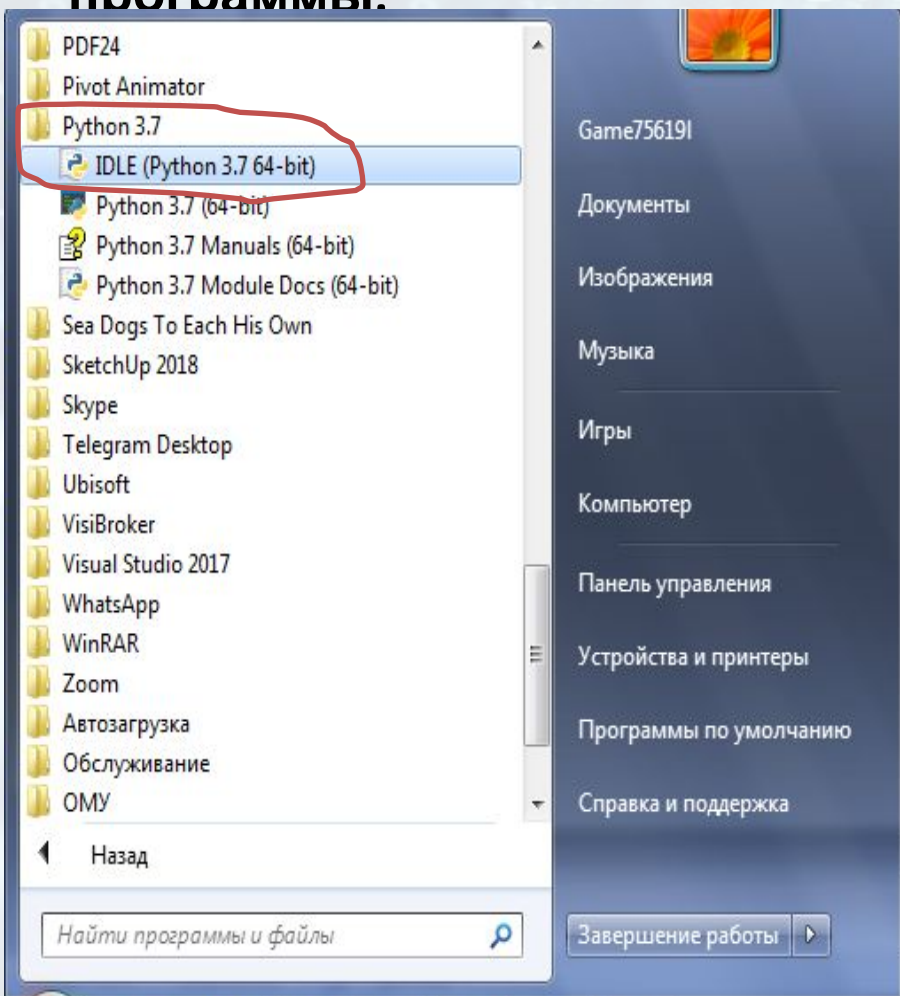
Python был разработан в конце 1989г.



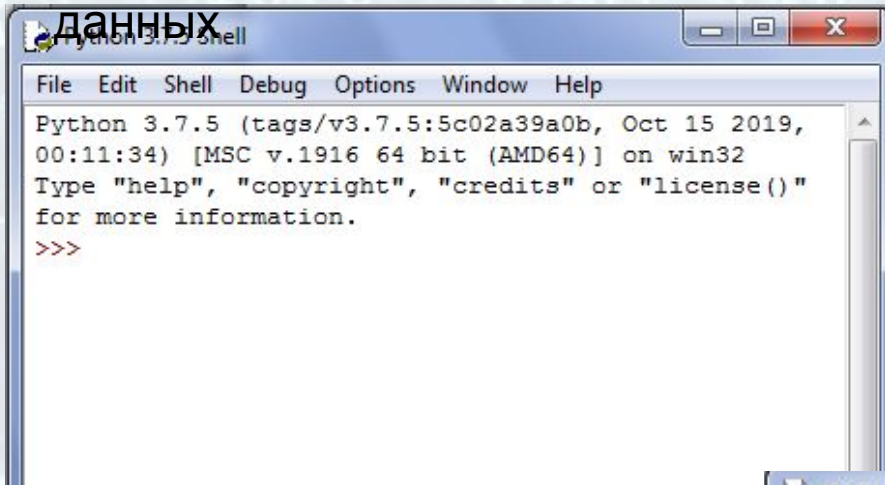


# Среда программирования - Python

## 1. Запуск программы.



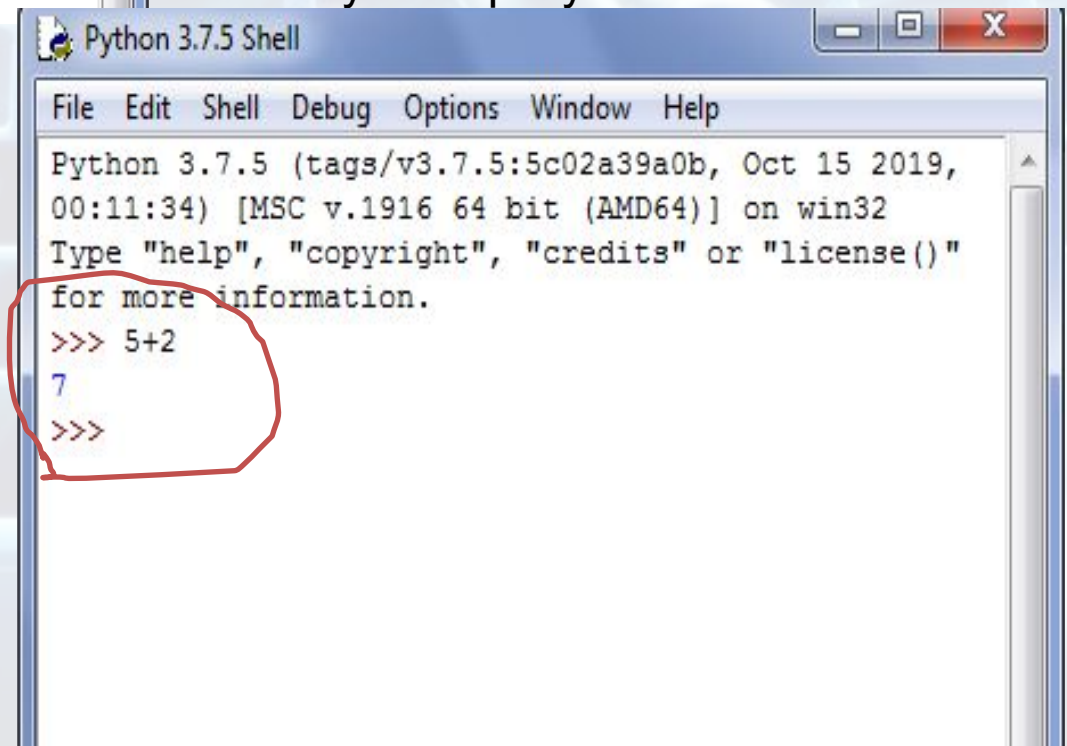
## 2. Построчный ввод



```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019,
00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()"
for more information.
>>>
```

3. Вводим команду и нажимаем Enter.

Получаем результат

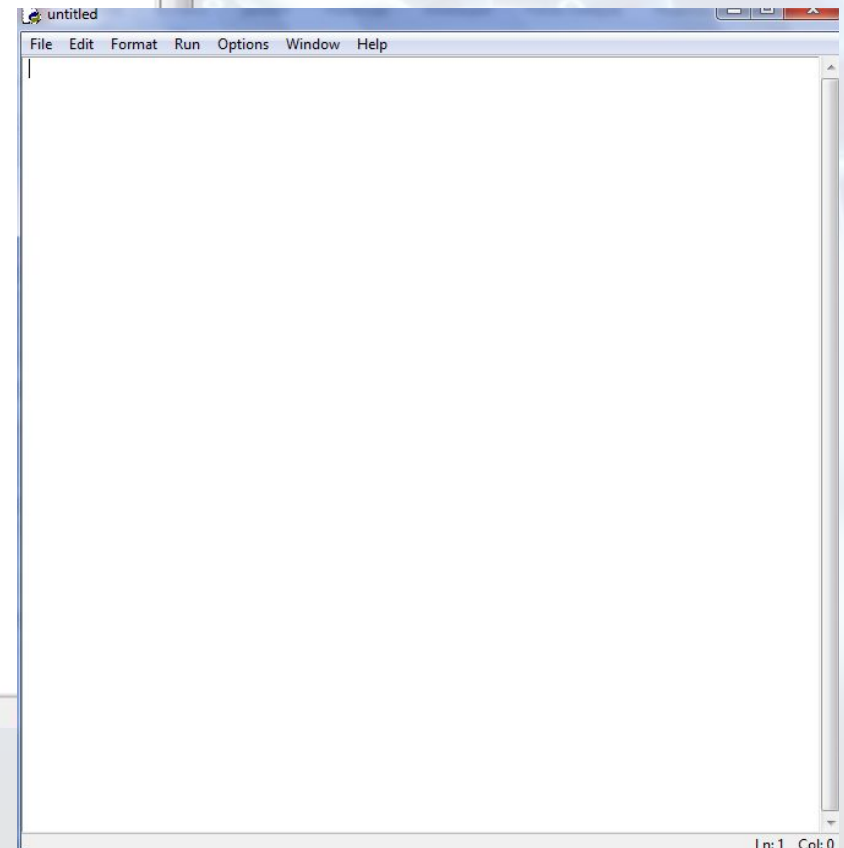
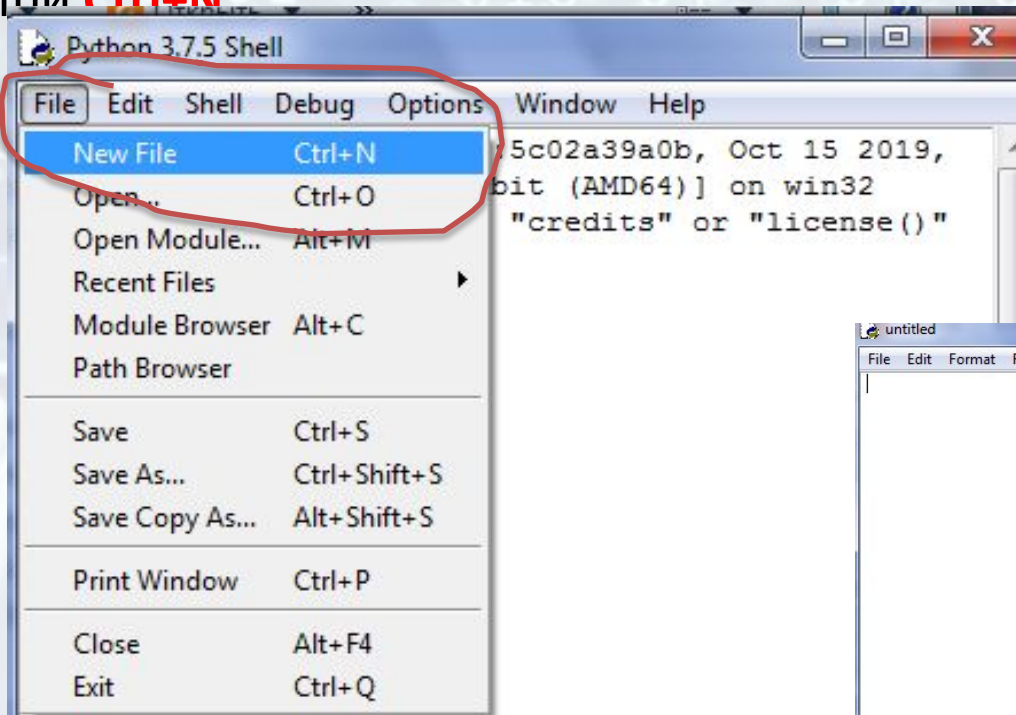


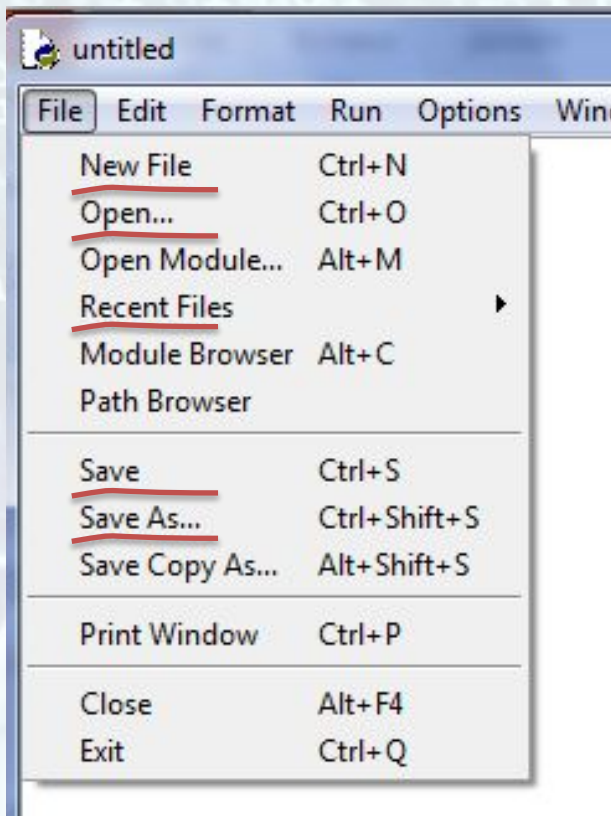
```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019,
00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()"
for more information.
>>> 5+2
7
>>>
```

### Недостатки:

- Программы не состоят из одной команды
- Невозможно сохранить результат

### 3. **Запуск** текстового редактора, для написания кода – Команда **File-New File,** ИЛИ **Ctrl+N**





**Меню File** содержит следующие команды:

**New File (Ctrl+N)** – открывает новое текстовое окно

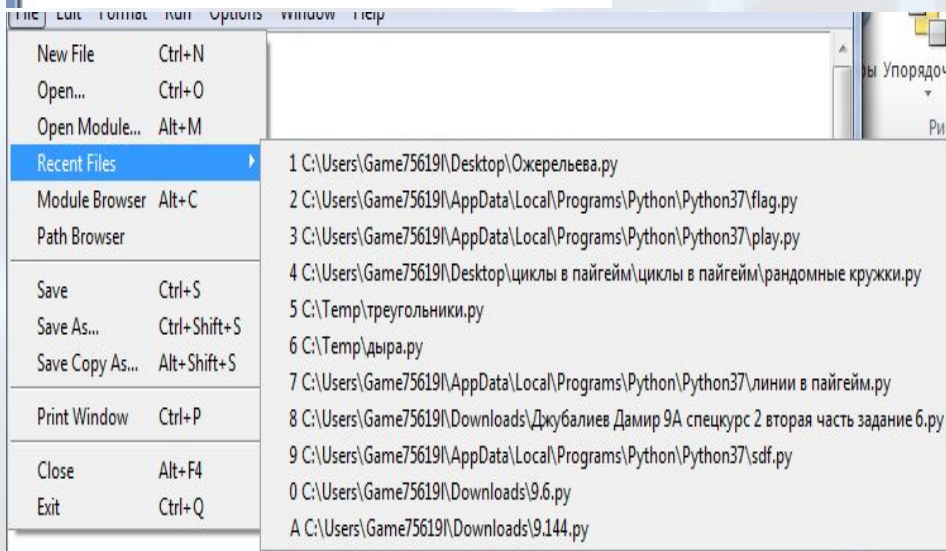
**Open (Ctrl+O)** – открыть файл какой-либо, уже написанной

программы

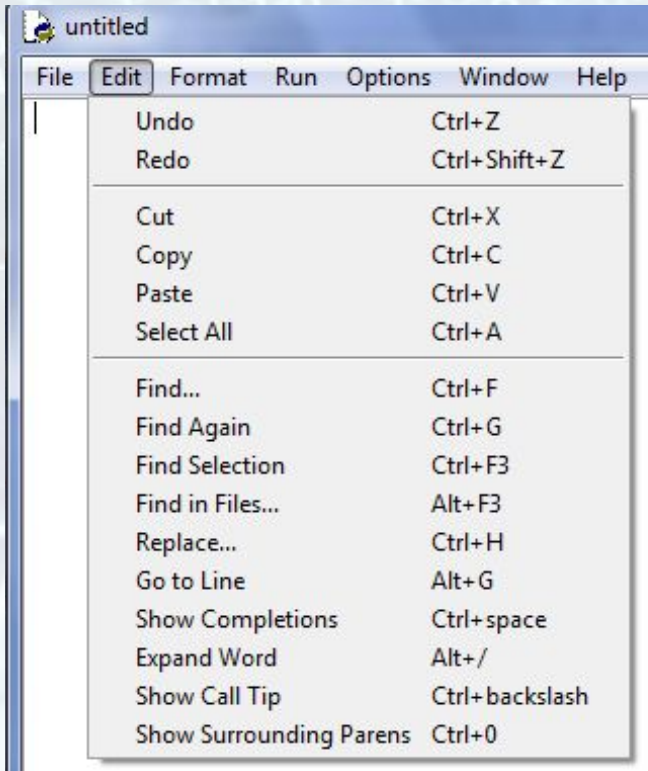
**Recent Files** – последние 10 файлов, с которыми вы работали, открыть из списка

**Save (Ctrl+S)** – сохранить изменения в программе, для первого сохранения работает как Save as

**Save as (Ctrl+Shift+S)** – сохранить с выбором места сохранения



**Все команды в МЕНЮ  
работают только в  
режиме Английского  
языка**



**Меню Edit** содержит следующие команды:

**Undo (Ctrl+Z)** – отменить действие

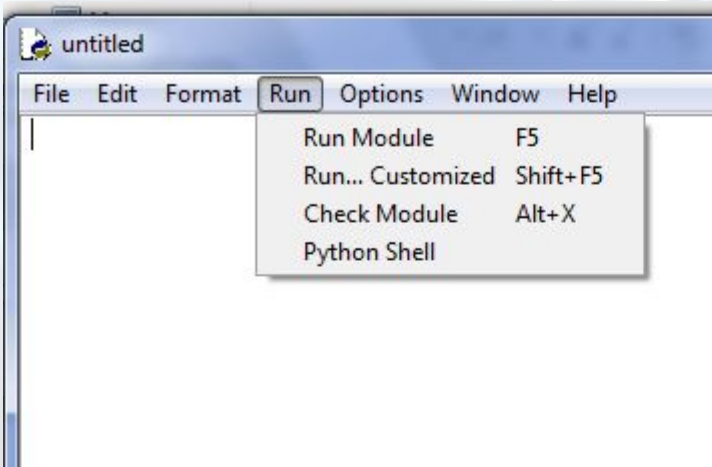
**Redo (Ctrl+Shift+Z)** – вернуть отмененное действие

**Cut (Ctrl+X)** – вырезать выделенный объект (вырезанный объект помещается в Буфер обмена)

**Copy (Ctrl+C)** – копировать выделенный объект

**Paste (Ctrl+V)** – вставить скопированный объект

**Select All (Ctrl+A)** – выделить все



**Меню Run** содержит следующие команды:

**Run Module (F5)** – запуск программы на выполнение (отладка и компиляция)

**Python Shell** – открыть окно с результатом

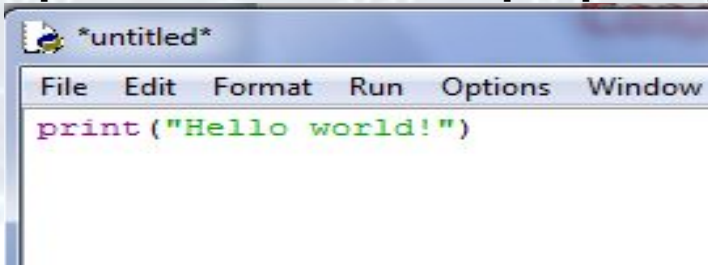
# Создание первой программы

Запускаем IDLE (изначально запускается в интерактивном режиме), после чего уже можно начинать писать первую программу. Традиционно, первой программой у нас будет “hello world”.

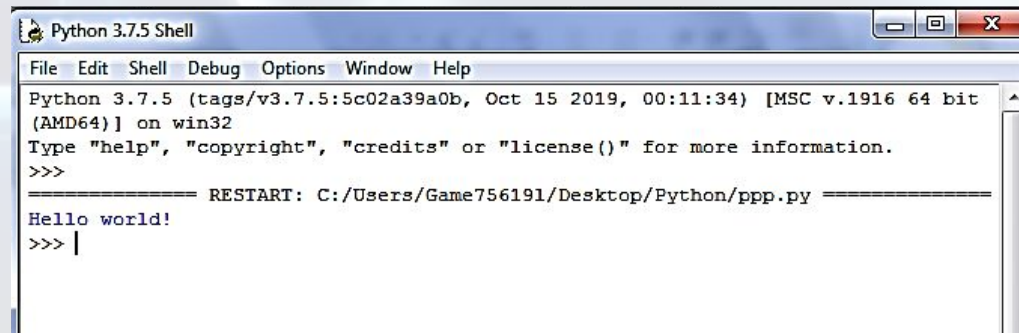
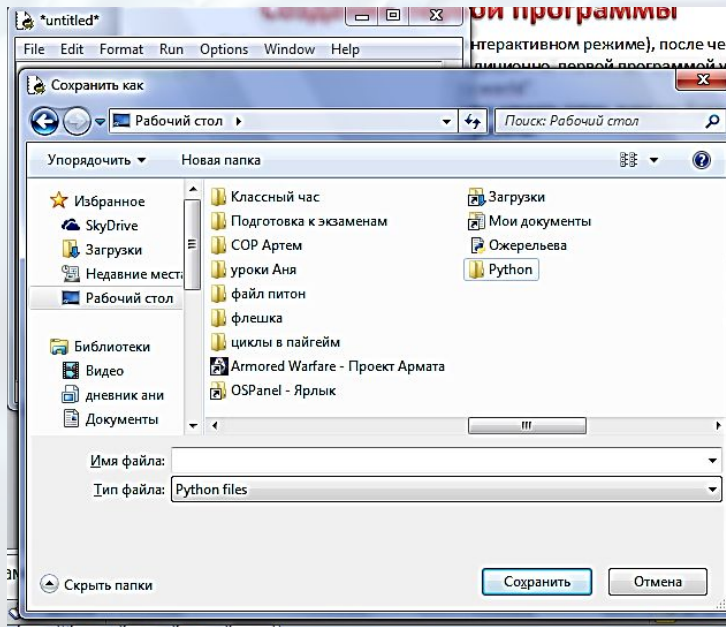
Прежде чем писать программу необходимо создать папку, куда мы будем класть наши работы.

Чтобы написать “hello world” на python, достаточно всего одной строки:

Вводим этот код в IDLE и нажимаем F5. Так как это наша первая программа, Python попросит нас ее сохранить, выбираем нашу Папку для сохранения (я создала папку Python на Рабочем столе). После сохранения мы увидим результат.

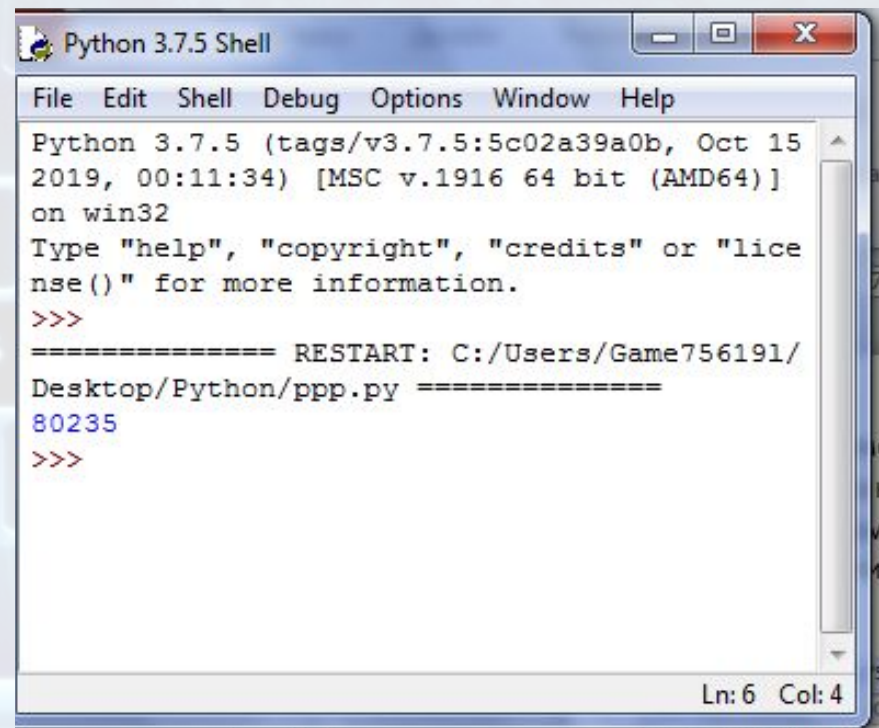
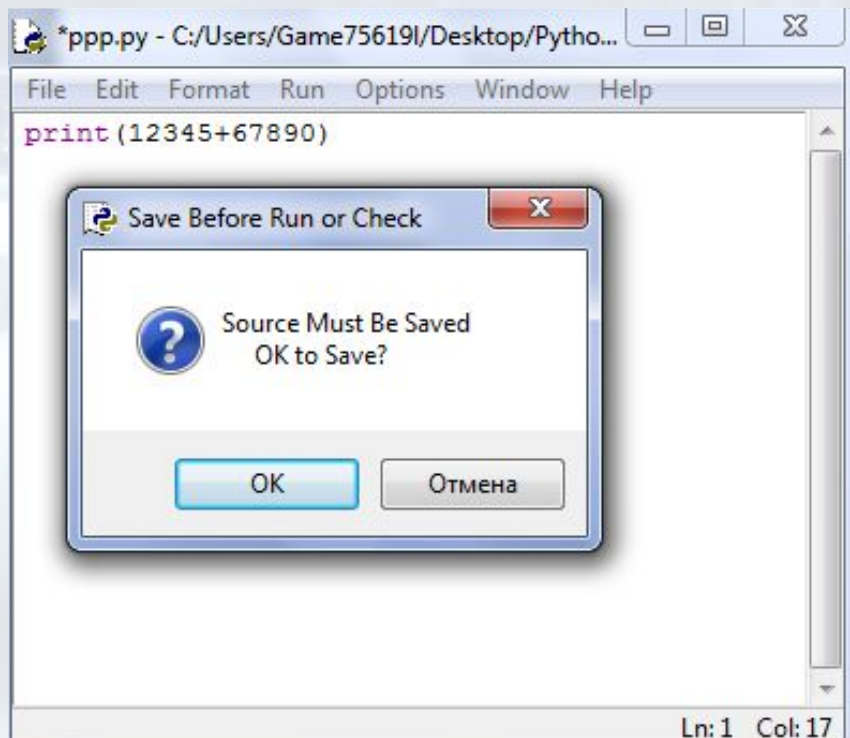


```
File Edit Format Run Options Window
print("Hello world!")
```



```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/Game756191/Desktop/Python/ppp.py =====
Hello world!
>>> |
```

Попробуйте поменять программу, запишите в скобках числа `print( 12345 + 67890 )`, обратите внимания, что их мы пишем без **кавычек**. Нажмите **F5**, компьютер спросит хотим ли мы изменить программу. Учтите, что нажав **ОК**, вы удалите предыдущую программу. Если вы создали новую программу, то для сохранения надо нажать **Save as**, в меню **File**.



Но недостаток этой программы состоит в том, что она складывает только два заранее известных числа. Если нужно сложить другие числа, придётся менять программу.

Чтобы программа могла выполнять расчёты при различных исходных данных, их вводят с клавиатуры, из файла, с какого-то устройства или через компьютерную сеть. Исходные данные (числа), которые введёт человек, нужно сохранить в памяти компьютера. Для этого используют *переменные*.

**Переменная — это величина, которая имеет имя, тип и значение. Значение переменной может изменяться во время выполнения программы.**

Таким образом, в переменных можно хранить данные во время работы программы и использовать их при вычислениях, когда они понадобятся.

**Идентификатор — это имя переменной**  
(от слова идентифицировать – отличать один объект от другого).

Имена переменных в Python могут включать латинские буквы (строчные и заглавные буквы *различаются*), цифры и знак подчеркивания «\_». Имя не может начинаться с цифры, иначе транслятору будет сложно определить, где начинается Желательно давать переменным «говорящие» имена, чтобы можно было сразу понять, зачем нужна та или иная переменная. Например, переменная с именем `name`, скорее всего, служит для хранения какого-то имени, а о назначении переменной `abc` догадаться очень сложно.





Определите, какие из следующих идентификаторов допустимы, а какие – нет.

1	Vasya	СУ-27	@mail_ru
m11	Петя	СУ_27	lenta.ru
1m	Митин брат	_27	"Pes barbos"
m 1	Quo vadis	СУ(27)	<Ладья>

В отличие от многих языков программирования (Паскаль, С, Java) переменные в языке Python не нужно объявлять. Память для переменной выделяется автоматически тогда, когда ей присваивается новое значение.

Присвоим переменной значение 5:

```
a = 5
```

Знак «**=**» обозначает специальную команду – **оператор присваивания**, с его помощью присваивают новое значение переменной.

Оператор присваивания также позволяет изменить значение переменной:

```
name = "Платон"
```

```
name = "Сократ"
```

Переменная может хранить только одно значение. При записи в неё нового значения «старое» стирается, и его уже никак не восстановить.

В языке Python каждая переменная имеет свой тип.

Тип нужен для того, чтобы определить:

- какие значения может принимать переменная;
- какие операции можно выполнять с этой переменной;
- как хранить её значения в памяти.

**str** – символьная строка (от англ. *string*),

**int** – целое число (от англ. *integer*),

**float** – вещественное число (от англ. *float*).

**Задание 1:** Что появится на экране после выполнения программы:

```
a=3
c = 5
print( a+c )
print( "a+c" )
```

Ответ:

```
8
a+c
```

**Задание 2:** Вначале переменные имели значения  $a = 4$  и  $b = 7$ . Чему будут равны значения этих переменных после выполнения программы:

```
a = a + 1
b = b + 1
a = a + b
b = b + a
a = a + 1
```

Ответ:

```
a=14
b=21
```

# Команда `print()` – вывод данных на экран

`a=3`

`c = 5`

`b=a+c`

`print (b)` – вывод числового значения /результат **8**

`print( a+c )` – вывод числового значения/ результат **8**

`print( "a+c" )` – вывод текста / результат **a+c**

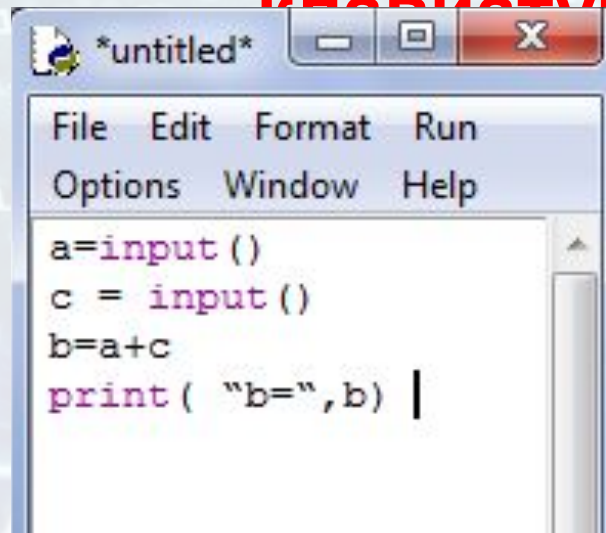
`print( "b=",b)` – вывод и текстового и числового значения/ результат **b=8**

`print( "a+c=",b)` - вывод и текстового и числового значения/ результат

**a+c=8**

`print( "b=",a+c)` - вывод и текстового и числового значения/ результат **b=8**

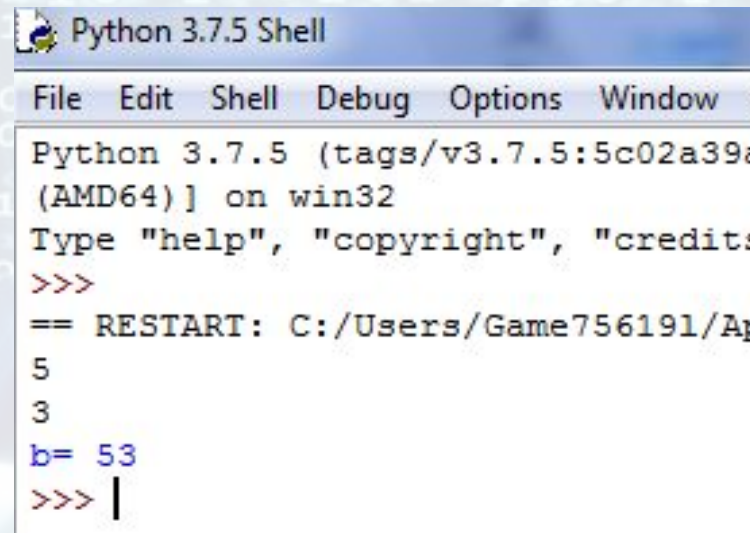
# Команда input() – ввод данных с клавиатуры



```
File Edit Format Run
Options Window Help

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

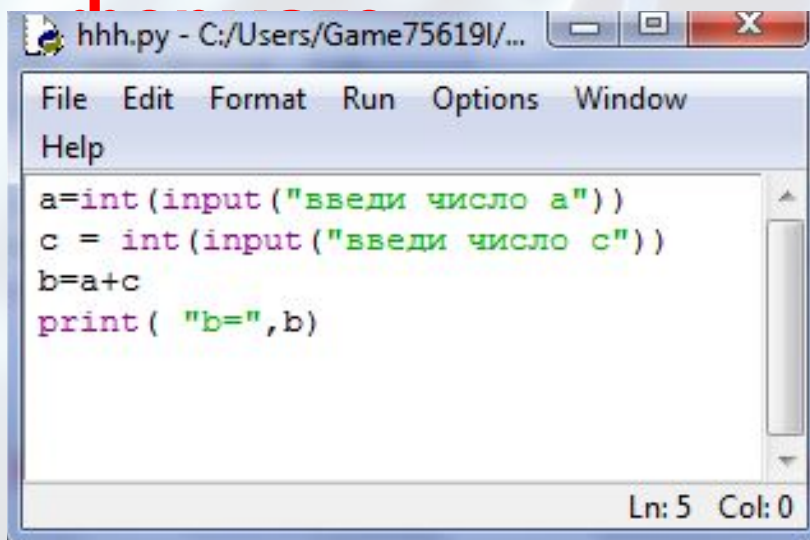
Результат



```
Python 3.7.5 Shell
File Edit Shell Debug Options Window

Python 3.7.5 (tags/v3.7.5:5c02a39a
(AMD64)] on win32
Type "help", "copyright", "credits
>>>
== RESTART: C:/Users/Game756191/Ag
5
3
b= 8
>>> |
```

# Команда input () вводит данные в текстовом

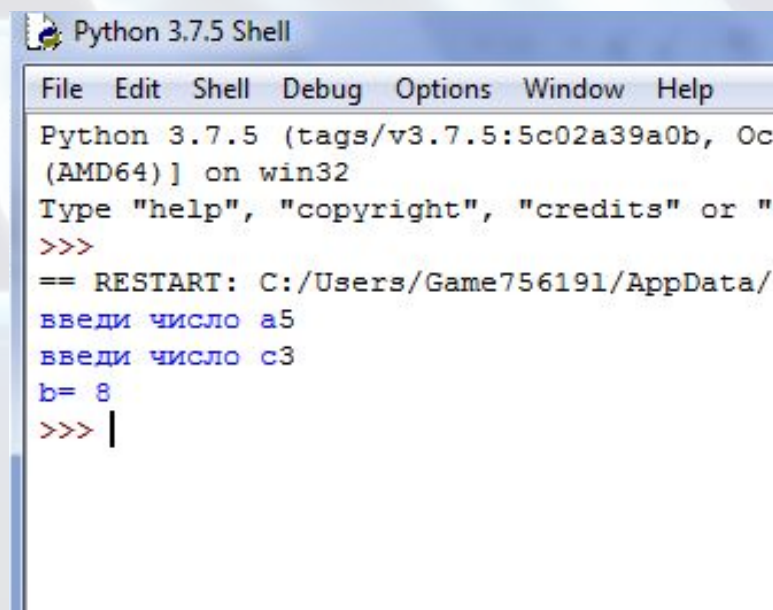


```
File Edit Format Run Options Window
Help

a=int(input("введи число a"))
c = int(input("введи число c"))
b=a+c
print( "b=",b)

Ln: 5 Col: 0
```

Результат



```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help

Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oc
(AMD64)] on win32
Type "help", "copyright", "credits" or "
>>>
== RESTART: C:/Users/Game756191/AppData/
введи число a5
введи число c3
b= 8
>>> |
```

# Поменяем тип int на

float

Результат

T

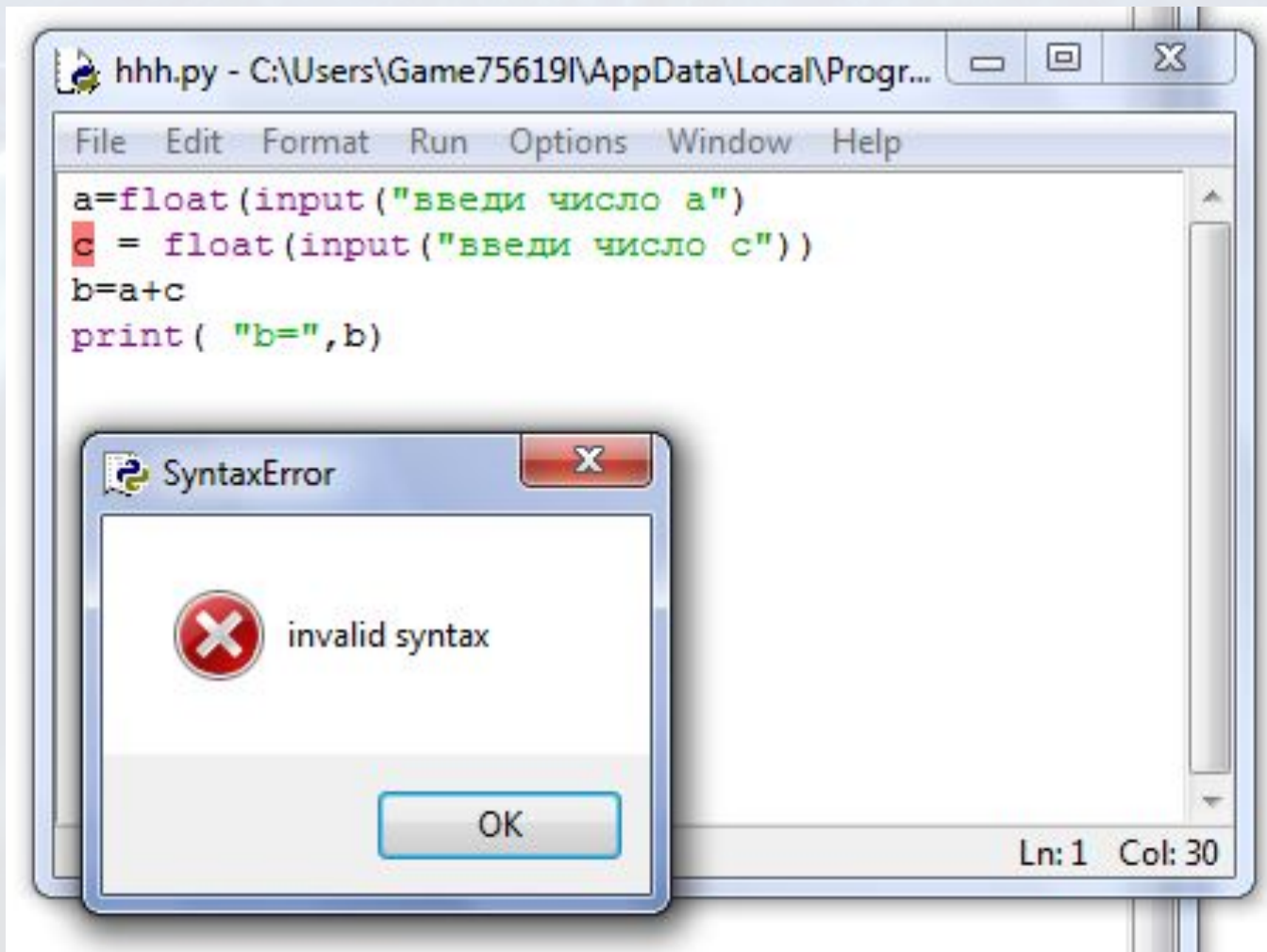


```
*hhh.py - C:/Users/Game756191/...
File Edit Format Run Options Window
Help
a=float(input("введи число a"))
c = float(input("введи число c"))
b=a+c
print( "b=",b)
Ln: 2 Col: 9
```

```
Python 3.7.5 Shell
File Edit Shell Debug Options Window
Python 3.7.5 (tags/v3.7.5:5c02a3
(AMD64)] on win32
Type "help", "copyright", "credi
>>>
== RESTART: C:/Users/Game756191/
введи число a5
введи число c3
b= 8.0
>>> |
```

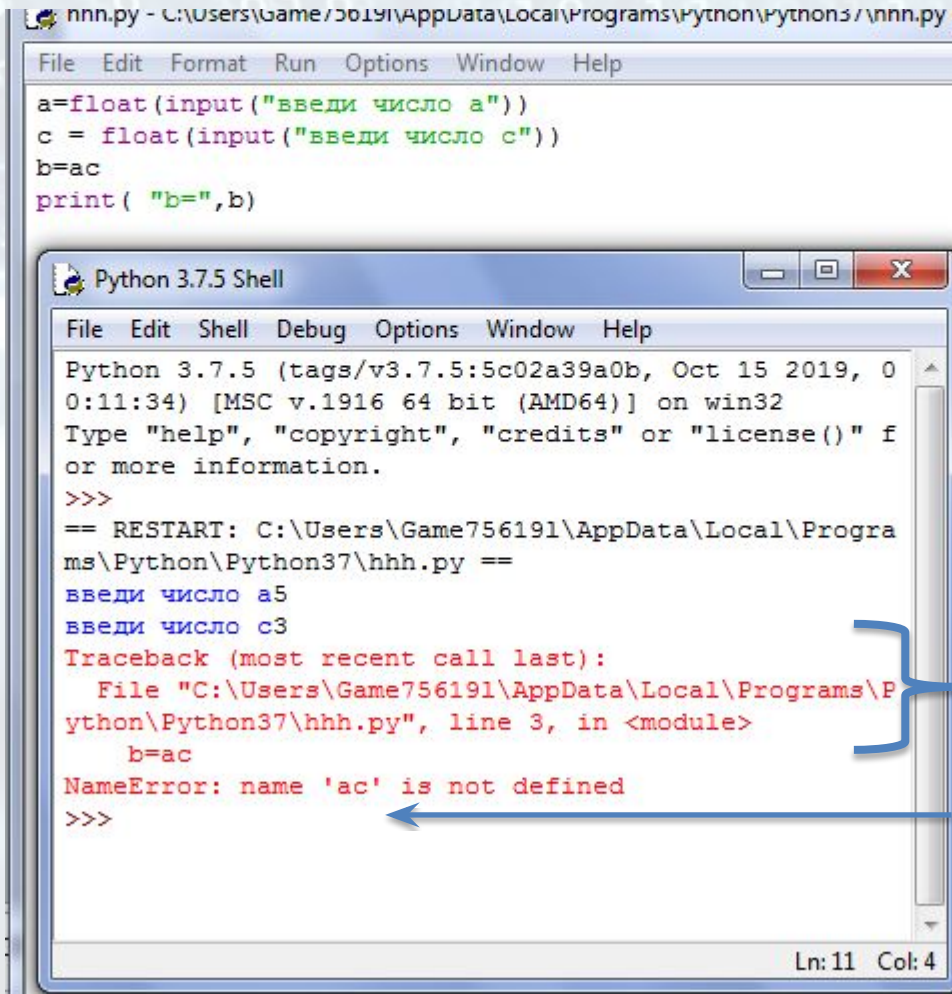
# Виды ошибок

1. **Синтаксическая ошибка** – пропущен знак, означает, что в строке перед выделенной с пропущен синтаксический знак. Это может быть скобка, кавычки, запятая и тд. Посмотрим внимательно. Пропущен знак `)` в конце строки.



# Виды ошибок

2. Другие виды ошибок программа указывает в окне ответа красным цветом. Прочитаем, что написал нам компьютер.



The image shows a screenshot of a Python IDE. The top window displays a script named 'hnh.py' with the following code:

```
a=float(input("введи число a"))
c = float(input("введи число c"))
b=ac
print( "b=",b)
```

The bottom window, titled 'Python 3.7.5 Shell', shows the execution of the script. It displays the prompt 'введи число a' followed by '5', and 'введи число c' followed by '3'. A red error message is shown:

```
Traceback (most recent call last):
  File "C:\Users\Game756191\AppData\Local\Programs\Python\Python37\hnh.py", line 3, in <module>
    b=ac
NameError: name 'ac' is not defined
```

Blue arrows point from the error message to the text boxes on the right.

В Моем файле, расположенного по данному адресу с именем hnh.py ошибка в строке 3, а именно в строке b=ac

Компьютер не понимает что такое ac

**Ошибка: между ac пропущен**

# Определи

```
hhh.py - C:\Users\Game756191\AppData\L...
File Edit Format Run Options Window Help
a=float(input("введи число a"))
c = float(input("введи число c"))
b=a+c
prin( "b=",b)
```

```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\Game756191\AppData\Local\Programs\Python\Python37\hhh.py ==
введи число a5
введи число c3
Traceback (most recent call last):
  File "C:\Users\Game756191\AppData\Local\Programs\Python\Python37\hhh.py", line 4, in <module>
    prin( "b=",b)
NameError: name 'prin' is not defined
>>> |
```

```
hhh.py - C:\Users\Game756191\AppData\L...
File Edit Format Run Options Window Help
a=float(input("введи число a"))
c = input("введи число c")
b=a+c
print( "b=",b)
```

```
Python 3.7.5 Shell
File Edit Shell Debug Options Window Help
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\Game756191\AppData\Local\Programs\Python\Python37\hhh.py ==
введи число a5
введи число c3
Traceback (most recent call last):
  File "C:\Users\Game756191\AppData\Local\Programs\Python\Python37\hhh.py", line 3, in <module>
    b=a+c
TypeError: unsupported operand type(s) for +: 'float' and 'str'
>>>
```



## ЗАДАНИЕ:

Напишите программу, определяющую Сумму, Разность, Произведение, Частное, Сумму квадратов двух целых чисел.

Рез

```
== RESTART: C:\Users\Game756191\AppData
ta\Local\Programs\Python\Python37\hhh
.py ==
введи число a5
введи число b3
a+b= 8
a-b= 2
b-a= -2
b*a= 15
b/a= 0.6
a/b= 1.6666666666666667
a*a+b*b= 34
>>> |
```

Ln: 14 Col: 4

# Арифметические выражения

Арифметические выражения обычно записываются в одну строчку. Они могут содержать константы (постоянные значения), имена переменных, знаки арифметических операций, круглые скобки (для изменения порядка действий).

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

в программе запишется как

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

Если мы забудем поставить скобки, то получим совсем другое выражение

$$a = c + b - 1/2*d \text{ - в математическом виде } \rightarrow a = c + b - \frac{1}{2} \cdot d$$

Как видите, мы получили совсем другое выражение. Если поставить скобки не там, тоже получим другое выражение:

$$a = c + b - 1/(2*d) \longrightarrow a = c + b - \frac{1}{2 \cdot d}$$

## Правила:

1. Числитель берется в скобки всегда, когда в нем есть знаки + и –
2. Знаменатель берется в скобки всегда, когда в нем есть более одной переменной или более одного числа

Для записи математических функций используются специальные команды:

$\sqrt{x}$  - sqrt(x)

Пример:  $\sqrt{x+5}$  → sqrt(x+5)

$x^y$  – x \*\* y

Пример:  $x^5$  → x \*\* 5

|x| - abs(x)

Пример: |x-8| → abs(x-8)

sinx – sin(x)

Пример:  $\sin^2x$  → (sin(x))\*\*2       $\sin x^2$  → sin(x\*\*2)

cosx – cos(x)

Пример: cos2x → cos(2\*x)

Для вычисления сложных выражений, оно делится на части:

$$\frac{\frac{100x+200y}{a-b} + \frac{1}{a}}{\frac{a+b}{a-b} - \frac{10a}{(a+b)(a+10b)}}$$

Для записи выражения берем дополнительные переменные **n** и **m**. Переменной **n** присваиваем значение верхней части выражения и значению **m** присваиваем значение **n** деленного на нижнюю часть выражения.

$$n = (100x + 200y) / (a - b) + 1 / a$$

$$m = n / ((a + b) / (a - b) - (10a) / ((a + b)(a + 10b)))$$

**1.15.** Получить линейную запись следующих выражений:

а)  $\frac{-1}{x^2}$ ;

г)  $\frac{a+b}{2}$ ;

ж)  $\frac{-b + \frac{1}{a}}{\frac{2}{c}}$ ;

к)  $2^{m^n}$ .

б)  $\frac{a}{bc}$ ;

д)  $5,45 \cdot \frac{a+2b}{2-a}$ ;

з)  $\frac{1}{1 + \frac{a+b}{2}}$ ;

в)  $\frac{a}{b}c$ ;

е)  $\frac{-b + \sqrt{b^2 - 4ac}}{2a}$ ;

и)  $\frac{1}{1 + \frac{1}{2 + \frac{1}{2 + \frac{3}{5}}}}$ ;

**Задание**

:

**1.16.** Перевести из линейной записи в обычную следующие выражения:

а)  $a/b/c$ ;

б)  $a \cdot b/c$ ;

в)  $a/b \cdot c$ ;

г)  $a+b/c$ ;

д)  $a+b/c$ ;

е)  $a+b/b+c$ ;

ж)  $(a+b)/(b+c)$ ;

1.17. Записать по правилам изучаемого языка программирования следующие выражения:

а)  $\sqrt{x_1^2 + x_2^2}$ ;

ж)  $2\pi R$ ;

н)  $\frac{ad + bc}{ad}$ ;

б)  $x_1x_2 + x_1x_3 + x_2x_3$ ;

з)  $b^2 - 4ac$ ;

о)  $\sqrt{1 - \sin^2 x}$ ;

в)  $v_0t + \frac{at^2}{2}$ ;

и)  $\gamma \frac{m_1m_2}{r^2}$ ;

п)  $\frac{1}{\sqrt{ax^2 + bx + c}}$ ;

г)  $\frac{mv^2}{2} + mgh$ ;

к)  $I^2R$ ;

р)  $\frac{\sqrt{x+1} + \sqrt{x-1}}{2\sqrt{x}}$ ;

д)  $\frac{1}{R_1} + \frac{1}{R_2}$ ;

л)  $absinc$ ;

с)  $|x| + |x+1|$ ;

е)  $mg \cos \alpha$ ;

м)  $\sqrt{a^2 + b^2 - 2ab \cos c}$ ;

т)  $|1 - |x||$ .