Язык программирования Python

Основы языка Python

Линейные программы

С. В. Чайченков, МБОУ Грушевская СОШ

Оператор (команда) присваивания

Оператор присваивания записывает в переменную, имя которой находится слева от знака «=» значение выражения, находящегося справа. Старое значение переменной при этом стирается.

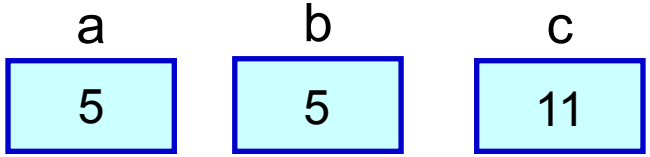
Общий вид оператора:

```
<имя переменной> = <выражение>
```

Например:

```
a = 5
b = a
c = a+b
c = c+1
```

В памяти:



В языке Python допускается множественное присваивание:

Запись оператора:	*Равносильная запись:*
`a, b = 0, 1`	`a = 0` `b = 1`
`a = b = 0`	`a = 0` `b = 0`

Допускается запись нескольких операторов в одной строке через символ «;».

С. В. Чайченков, МБОУ Грушевская СОШ

Оператор вывода

Вывод данных из оперативной памяти на экран осуществляется с помощью оператора (функции) вывода **print** («печатать»):

```
print(<выражение1>, <выражение2>, ..., <выражениеN>)
```

- На экран выводятся значения переменных и выражений, строковые значения выводится на экран без кавычек.
- Выводимые значения разделяются пробелом (по умолчанию).
- После выполнения оператора происходит автоматический переход на новую строку.

Например:

```
print ("Масса равна", m, "кг");
```

Для m=15 на экране появится:

```
Масса равна 15 кг
```

Здесь и далее символом □ обозначен пробел.

Оператор вывода

- Вместо пробела можно использовать другие символы в качестве разделителя, указав их после слова **sep** («separator»).
- Чтобы убрать переход на новую строку после выполнения оператора, используется параметр **end**.

Нужный вариант вывода	Оператор	На экране
По умолчанию	`print (1, 20, 300)`	1 20 300
Без разделителя	`print (1, 20, 300, sep="")`	120300
Через запятую и пробел	`print (1, 20, 300, sep=", ")`	1, 20, 300
Каждое с новой строки	`print (1, 20, 300, sep="\n")`	1 20 300
Без перехода на новую строку	`print (1, end="")` `print (20)`	120

Оператор ввода

Для ввода значений переменных с клавиатуры в процессе выполнения программы используется оператор (функция) ввода **input** («ввод»):

```
<имя_переменной> = input()
```

При выполнении оператора:

- компьютер переходит в режим ожидания данных;
- пользователь вводит с клавиатуры данные в виде строки символов;
- для завершения ввода пользователь нажимает клавишу Enter;
- введенная строка записывается в указанную переменную.

Если вводится не строка, а число, необходимо выполнить преобразование типов с помощью функций **int** (для целых) и **float** (для вещественных).

Например:

```
print("Введите слово и два числа:")
x = input()
y = int(input())
z = float(input())
print(x, y, z)
```

На экране:

```
Введите слово и два числа:
ноль
1
2
ноль 1 2.0
```

Оператор ввода

Можно в скобках указать текст подсказки для пользователя.

Например:

```
x = input("Введите слово: ")
y = int(input("Введите целое число: "))
z = float(input("Введите вещественное число: "))
print (x, y, z)
```

На экране:

```
Введите слово: ноль
Введите целое число: 1
Введите вещественное число: 2
ноль 1 2.0
```

Оператор ввода

Можно в одной строке ввести несколько значений через пробел. Для этого используется функция **split** («расщепить»). Затем данные необходимо преобразовать к нужному типу по отдельности.

Например:

```
a, b, c = input("Введите a,b,c через пробел: ").split()
a, b, c = int(a), int(b), int(c)
print (a, b, c)
```

На экране:

```
Введите a,b,c через пробел: 1 2 3
1 2 3
```

Оператор комментария

Используется для включения в программу любых пояснений, предназначенных человеку.

Комментариями считается любой текст после символа **#** до конца строки. При выполнении программы комментарии игнорируются.

Пример программы:

```
# Длина окружности и площадь круга
r = float(input("Введите радиус: "))
c = 2*3.14*r    # длина окружности
s = 3.14*r**2      # площадь круга
print ("c=", "{:7.3f}".format (c))
print ("s=", "{:7.3f}".format (s))
```

На экране:

```
Введите радиус: 10
c=    62.80
s=   314.00
```

Стандартные функции

Функции имеют определенное *имя* и один или несколько *аргументов* в скобках. Функция возвращает свое значение в то место программы, из которого она вызывается.

Некоторые стандартные функции, встроенные в ядро языка Python

Функция	Назначение	Тип аргумента	Тип результата
`abs(x)`	абсолютная величина (модуль числа x)	`int, float`	как у аргумента
`int(x)`	преобразование вещественного числа к целому значению (отбрасывание дробной части)	`float`	`int`
`round(x)`	округление вещественного числа до заданного количества знаков после точки (по умолчанию – до ближайшего целого)	`float`	`int, float`

Стандартные функции

Большинство стандартных функций языка Python разбиты на группы по назначению, каждая группа записана в отдельном файле, который называется **модулем**. Подключение модуля осуществляется командой **`import`**.

Например:

```
# подключаем все функции из модуля math
from math import *
```

Стандартные функции модуля `math`

Функция	Назначение	Тип аргумента	Тип результата
`sqrt(x)`	квадратный корень из x	`int, float`	`float`
`sin(x)`	синус угла x в радианах	`int, float`	`float`
`cos(x)`	косинус угла x в радианах	`int, float`	`float`

Стандартные функции

После подключения модуля к его функциям можно обращаться так же, как к встроенным. Например:

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

```
(-b + sqrt(b**2 - 4*a*c))/(2*a)
```

6 5 1 4 2 3 8 7

Можно подключать не все функции, а только необходимую. *Например:*

```
# подключаем функцию randint() из модуля random
from random import randint
```

Стандартные функции модуля `random`

Функция	Назначение	Тип аргумента	Тип результата
`random()`	случайное число из полуинтервала [0, 1)	–	`float`
`randint(a, b)`	случайное число из отрезка [a, b]	`int`	`int`

Стандартные функции

Пример со стандартными функциями:

```
# Стандартные функции
a = 3.56
print (a)
print (round(a))
print (round(a, 1))
print (int(a))
from math import *
b = 16
print (sqrt(b))
from random import randint
x = randint(1, 10)
y = randint(1, 10)
z = randint(1, 10)
print (x, y, z)
```

На экране:

```
3.56
4
3.6
3
4.0
2 10 8
```

https://forms.gle/ToW6bPndf3ysuFDQ8