

Ссылка на урок:

<https://coreapp.ai/app/player/lesson/67054fbde1d74c83b99c45c7> (для учеников).

Занятие 4

Тема. Оператор присваивания. Переменные и константы

1. Теоретическая часть

Самый известный оператор, без которого никак не обойтись ни в одном языке программирования – это **оператор присваивания**, который служит для присваивания переменным значений. Общий вид оператора:

<имя переменной> = <значение или вычисляемое выражение>

Выражение, стоящее справа от оператора присваивания, вычисляется, и полученное значение присваивается переменной, стоящей слева от оператора присваивания. При этом предыдущее значение, хранящееся в переменной, стирается и заменяется на новое. Если такой переменной не существует, то она создается и в неё записывается значение.

Примеры:

a = 25

b = "Привет ! "

c = 1.4 + 5.7 * a

d = a < c

e = "Мир! " + b

В языке Python разрешено множественное присваивание. Запись **a, b = 19, 25** равносильна паре операторов присваивания:

a = 19

b = 25

Также в языке Python есть сложные операторы присваивания, они представлены в таблице:

Сложные операторы присваивания

Обозначение	Описание
+=	Значения левой и правой частей оператора присваивания суммируются, результат присваивается переменной в левой части
-=	Вычитается значение правой части из значения переменной в левой части, результат присваивается переменной в левой части

Переменные

Любая программа состоит из набора команд (инструкций) и данных. Чтобы хранить данные, в любом языке программирования, и Python не исключение, есть такое понятие как *переменная*.

Переменная – это выделенная именованная область памяти для хранения каких-либо данных.

В переменной может быть сохранён промежуточный или конечный результат работы программы.

В Python, в отличие от некоторых других языков программирования, объявлять переменную отдельно не нужно — достаточно сразу её **инициализировать**, то есть присвоить значение:

`greeting = 'Father!'`

↑
имя
переменной

↑
оператор
присваивания

↑
значение

Для имени переменной используется любой набор допустимых символов, к которым относятся буквы английского алфавита, цифры и знак `_`. При этом цифру нельзя ставить в начале. Имена переменных регистрозависимы, то есть имя `hello` и имя `HELLO` — это два разных имени для двух разных переменных. Имена переменных могут быть простыми, например, **`greeting`**, а могут состоять из нескольких слов. В этом случае они записываются строчными буквами и разделяются символом подчеркивания `_`, например, **`rubles_count`**, **`dollars_count`** (стиль *snake_case* – стиль написания составных слов).

Количество создаваемых переменных неограниченно. Большие программы содержат десятки и сотни тысяч имен переменных.

Чтобы программу было удобно читать, принято создавать переменные как можно ближе к тому месту, где они используются.

Например:

```
greeting1 = 'Father!'  
print(greeting1)
```

Также, для повышения читабельности программы, имена переменных должны соответствовать содержанию.

Само слово «переменная» подсказывает, что ее можно менять. И действительно, со временем внутри программы значения переменных могут изменяться.

Например:

```
# greeting - переводится как приветствие  
greeting = 'Father!'  
print(greeting) # => Father!  
  
greeting = 'Mother!'  
print(greeting) # => Mother!
```

Переменную нужно сначала инициализировать, а потом использовать.

Пример ошибки, которую часто допускают новички:

```
print(greeting)  
  
greeting = 'Father!'
```

Разновидностью в переменной Python является константа. Константы играют важную роль в программировании. Они используются для хранения значений,

которые не изменяются в процессе выполнения программы (например, число π).

В Python нет встроенного механизма для объявления констант. Значения переменных могут быть изменены в любой момент, и нет прямого способа указать, что переменная должна оставаться константой после ее инициализации.

Но есть общепринятый подход для создания констант в Python, который основан на соглашениях об именовании. Согласно этому соглашению (PEP 8), **имена констант должны быть написаны прописными (заглавными) буквами и менять это значение ни в коем случае нельзя.**

Например:

```
PI = 3.14
print(PI)    # => 3.14
GRAVITY = 9.8
print(GRAVITY)    # => 9.8

# Вычисления длины окружности, если известен радиус.
pi=3.14159
r = float(input("Введите радиус окружности: "))
l = 2*pi*r
print('Длина окружности =', f'{l:0.3}')
```

! Решение будет на следующей странице

2. Просмотр учебного видеоролика «Переменные и присваивание в Python» (с целью расширения знаний, наглядного представления и закрепления материала):

<https://rutube.ru/video/25d504f3e4c596eb05e78e0f49f0ebdf/?r=plemwd>

(длина ролика: 8:23).

3. Закрепление знаний

ЗАДАНИЕ 1.

Напишите и выполните программу на платформе Programiz.com, которая вычисляет длину окружности, если известен радиус. Значение числа π принять равным 3.14159, значение радиуса задать равным 15 (см).

Решение.

main.py	Run	Output
<pre>1 # Вычисления длины окружности, если известен радиус. 2 PI = 3.14159 3 r = 15 4 l = 2*PI*r 5 print(l) 6</pre>		<pre>94.2477 === Code Execution Successful ===</pre>

ЗАДАНИЕ 2.

Напишите и выполните программу на платформе Programiz.com, которая:

- а) вычисляет среднее арифметическое переменных x_1 и x_2 ;
- б) уменьшает на единицу значения переменной k ;
- в) увеличивает на единицу значения переменной i .

Решение.

а)

main.py	Run	Output
<pre>1 2 x1 = 10 3 x2 = 20 4 y = (x1 + x2)/2 5 print(y)</pre>		15.0 === Code Execution Successful ===

б)

main.py	Run	Output
<pre>1 2 k = 10 3 k = k - 1 4 print(k)</pre>		9 === Code Execution Successful ===

в)

main.py	Run	Output
<pre>1 2 i = 10 3 i = i + 1 4 print(i)</pre>		11 === Code Execution Successful ===