Курс повышения квалификации

- Язык программирования Python

Python

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

- Высокоуровневый язык программирования
- Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.
- Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование.

На Python были написаны следующие программы:

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

Урок 1: Типы данных и операторы

Уровень: начинающий
Курс: введение Python

Содержание

- Добро пожаловать на урок Типы данных и Операторы! Вы изучите:
- Типы данных: целые числа, числа с плавающей точкой, логические переменные, строки;
- Операторы: арифметические операторы, операторы присвоения, сравнения, логические;
- Встроенные функции, преобразование типов;
- Рекомендации по разделителю и оформлению.

Арифметические операторы

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

```
>>> print(2+2)
4
>>> print(5-3)
2
>>> print(4*3)
12
>>> print(9/3)
3.0
>>> print(10%3)
1
>>> print(4**2)
16
>>> print(20//6)
3
```

\+ Сложение;

\- Вычитание;

* Умножение;

/ Деление;

% Деление по модулю (возвращает остаток от деления);

** Возведение в степень (обратите внимание, что оператор ^ не выполняет эту операцию, как вы, вероятно, видели в других языках);

// Делит и округляет до ближайшего целого.

Обычный порядок математических операций выполняется в Python.

Тест: Средний счет за электроэнергию

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

Пришло время попробовать вычисления в Python!

Мои счета за электроэнергию за последние три месяца составили $23, $32 и $64. Каков среднемесячный счет за электроэнергию за трехмесячный период? Напишите выражение для вычисления среднего значения и используйте print () для просмотра результата.

Тест: Рассчитать

В этом задании вы будете делать некоторые расчеты для плиточника. Две части пола нуждаются в плиточном покрытии. **В одной части 9 плиток по ширине и 7 плиток по длине**, другая - **5 плиток по ширине и 7 плиток по длине**. Плитка поставляется в упаковках по 6 плиток в каждой.

1. Сколько плиток нужно?
2. Вы покупаете **17 упаковок плиток, по 6 плиток в каждой**. Сколько плиток останется?

ВОПРОС 3 ИЗ 3:

- Какие из этих строк кода Python хорошо отформатированы? Как бы вы улучшили считываемость кода без хорошего форматирования? (выберите все подходящие варианты

```
print(((3+ 32))+ -15//2)
```

```
print((17 - 6)%(5 + 2))
```

```
print ((1 + 2 + 4) / 13)
```

```
print(4/2 - 7*7)
```

Переменные и операторы присвоения

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

- **Переменная I**

```
>>> city_population = 584605
>>> print (city_population)
584605
```

- **“city_population”** это название переменной;
- **“=“** - оператор присвоения;
- **“584605”** – значение переменной.

Переменные и операторы присвоения

- Если x = y, y **не равняется** x;
- **В Python, "=" это оператор, который присваивает значение справа названию переменной слева.**

Переменные II

- Несколько примеров:

```
>>> x = 2
>>> y = x
>>> print (y)
2
```

```
>>> x = 2
>>> y = z
Traceback (most recent call last):
  File "<pyshell#13>", line 1, in <module>
    y = z
NameError: name 'z' is not defined
```

```
>>> x = 3
>>> y = 2
>>> z = 1
>>> x, y, z = 3, 2, 1
```

Переменные II

- Используйте описательное название переменной:

```
>>> city_population = 584605
>>> city_area = 142354
>>> city_density = city_population / city_area
>>> print (city_density)
4.106698793149472
```

- Используйте обычные буквы, цифры или подчеркивание в названиях переменных:
  - table6 или table_6 – правильно;
  - table 6 или 6table 6 – неправильно.

Переменные II

Вы не можете использовать зарезервированные слова или встроенные идентификаторы, которые выполняют важную функцию в Python

Keywords in Python programming language

False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Переменные II

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

Способ именования переменных в Python заключается в использовании всех строчных символов и подчеркивания для разделения слов.

YES

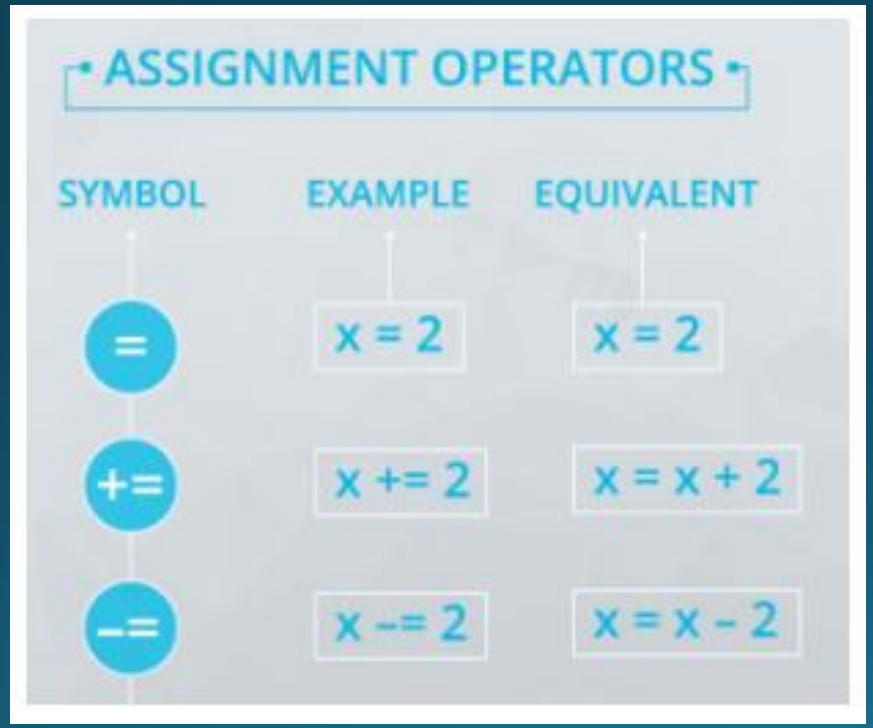
```
my_height = 58
my_lat = 40
my_long = 105
```

NO

```
my height = 58
MYLONG = 40
MyLat = 105
```

Оператора присвоения

- Ниже приведены операторы присвоения из видео. Вы также можете использовать *= аналогичным образом. Но это менее распространено, чем операции, представленные ниже.



Ввод данных

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

- Для ввода данных в программу мы используем функцию input(). Она считывает одну строку.Вот программа, которая считывает имя пользователя и приветствует его:

```
print('Как вас зовут?')
name = input()  # считываем строку и кладём её в переменную name
print('Здравствуйте, ' + name + '!')
```

Тест: назначение и изменение переменных

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

1. Создайте 2 переменные и поменяйте их значения местами
2. Напишите программу, которая приветствует пользователя, выводя слово Hello, введенное имя и знаки препинания по образцу:

ввод: Harry

вывод: Hello, Harry

Целые числа и числа с плавающей запятой

ELABORATION.ASIA
INTERNATIONAL EDUCATION CENTRE

- Существует два типа данных Python, которые можно использовать для числовых значений:
  - **int** - для целочисленных значений;
  - **float** - для десятичных значений или значений с плавающей точкой.
- Вы можете создать значение, соответствующее типу данных, используя следующий синтаксис:

```
x = int(4.7)    # x is now an integer 4
y = float(4)    # y is now a float of 4.0
```

Целые числа и числа с плавающей запятой

- Вы можете проверить тип, используя функцию **типа**:

```
>>> print(type(x))
int
>>> print(type(y))
float
```

•Поскольку значение с плавающей точкой или аппроксимация для 0.1 фактически немного больше 0.1, когда мы сложим несколько из них вместе, мы увидим разницу между математически правильным ответом и тем, который создает Python.

0.1 * 7 = 0.7000000000000001

Целые числа и числа с плавающей запятой

Следуйте этим инструкциям, чтобы порадовать других программистов и себя в будущем!

Good

```
print(4 + 5)
```

Bad

```
print(                    4 + 5)
```