Python programming for beginners

Stefan Zhauryd
Instructor

Module 2

Data types, variables, basic input-output operations, basic operators

Python **input()** Function

Note:

the input() function is invoked with one argument - it's a string containing a message;
the message will be displayed on the console before the user is given an opportunity to enter anything;
input() will then do its job.

The **input()** function allows user input, example:

Ask for the user's name and print it:

```
print('Enter your name:')
x = input()
print('Hello, ' + x)
```

- Output:

```
Enter your name:
Hendi
Hello, Hendi
```

```
1  # Testing TypeError message.
2
3  anything = input("Enter a number: ")
4  something = anything ** 2.0
5  print(anything, "to the power of 2 is", something)
```

Python **input()** Function float(input())

```
Console >_

Enter a number: 5
Traceback (most recent call last):
  File "main.py", line 4, in <module>
    something = anything ** 2.0
TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'float'
```

Note:

the input() function is invoked with one argument - it's a string containing a message;
the message will be displayed on the console before the user is given an opportunity to enter anything;
input() will then do its job.

Python x =int(**input()**) Function

```
x = input() # ‘5.5’
x = float(x) # 5.5
```

Note:

- **the input() function is invoked with one argument - it's a string containing a message**;
- **the message will be displayed on the console before the user is given an opportunity to enter anything;**
- **input() will then do its job.**

- There may be times when you want to specify a type on to a variable.
- This can be done with casting.
- Python is an object-orientated language, and as such it uses classes to define data types, including its primitive types.
- **Casting in python is therefore done using constructor functions**:
  - **int()** - **constructs an integer** number from an **integer** literal, a **float** literal (by rounding down to the previous whole number), or a **string** literal (providing the string represents a whole number)
  - **float()** - **constructs a float** number **from** an **integer** literal, a **float** literal or a **string** literal (providing the string represents a float or an integer)
  - **str()** - **constructs a string from a wide variety of data types**, including **strings**, **integer** literals and **float** literals

Type **casting – to Integer**

- Example:

```
x = int(1)   # x will be 1
y = int(2.8) # y will be 2
z = int("3") # z will be 3
```

- Output:

```
1
2
3
```

Type **casting – to float**

- Example:

```
x = float(1)     # x will be 1.0
y = float(2.8)   # y will be 2.8
z = float("3")   # z will be 3.0
w = float("4.2") # w will be 4.2
print(x)
print(y)
print(z)
print(w)
```

- Output:

```
1.0
2.8
3.0
4.2
```

Type **casting – to string**

- Example:

```
x = str("s1") # x will be 's1'
y = str(2)    # y will be '2'
z = str(3.0)  # z will be '3.0'
print(x)
print(y)
print(z)
```

- Output:

```
s1
2
3.0
```

String operators - introduction

Concatenation

- The + (plus) sign, when applied to two strings, becomes a concatenation operator,
- Example:

```
1  fnam = input("May I have your first name, please? ")
2  lnam = input("May I have your last name, please? ")
3  print("Thank you.")
4  print("\nYour name is " + fnam + " " + lnam + ".")
```

Output:

```
May I have your first name, please? Hendi
May I have your last name, please? Hermawan
Thank you.

Your name is Hendi Hermawan.
```

String operators - **Replication**

Replication

- The * (asterisk) sign, when applied to a string and number (or a number and string, as it remains commutative in this position) becomes a replication operator, example:

- `"James" * 3` gives `"JamesJamesJames"`
- `3 * "an"` gives `"ananan"`
- `5 * "2"` (or `"2" * 5`) gives `"22222"` (not `10`!)

- Example 2:

```
1  print("+" + 10 * "-" + "+")
2  print(("|" + " " * 10 + "|\n") * 5, end="")
3  print("+" + 10 * "-" + "+")
```

- Output:

```
+----------+
|          |
|          |
|          |
|          |
|          |
+----------+
```

Examples

```
a = 100

print(f'{a = }')

print('{} {} {}'.format(a, a+5, a+10))
```

```
a = 100
100 105 110
```

```
>>> print(u"\\\\\\\\\\")
\\\\\
>>> print(u'\\\\\\')
\\\
>>> print(r"\\\\\\\\\\")
\\\\\\\\\\
>>>
```

Home work 2.0

```
x = int(input())   # Ffg.
y = ...
print(x, “+”, y, “=”, x+y)
..
```

Write the simple calculator.

Operations: + - / * ** // %

- Use the input() to get type of operation and data from user.
- input() int() float() and so on to convert user inputs to relevant data types
- **Just investigate, don't afraid of mistakes**

ЗАДАНИЯ

1) Прорешать всю классную работу
2) Выполнить все домашние задания

Почитать:

1) Byte of Python – книга в нашем канале в telegram
стр. 47 по 54
стр. 55 по 63

Крайний срок сдачи 28/09 в 21:00 (можно раньше, но не позже)

ЗАДАНИЯ

Название файлов, которые вы отправляете мне в telegram:

Vasia_Pupkin_class_work_L2_1.py
Vasia_Pupkin_L2_1.py

Формат сообщения которое вы присылаете мне
(после полного выполнения домашнего задания, только один раз) в Telegram:
Добрый день/вечер.
Я Вася Пупкин, и это мои домашние задания к лекции 2 часть 1.
И отправляете файлы

Крайний срок сдачи 28/09 в 21:00 (можно раньше, но не позже)

Create your possibilities. Bye bye.