30 Days Of Python: Day 2 - Variables, Builtin Functions

LinkedIn | Follow @asabeneh
Author: Asabeneh Yetayeh
Second Edition: July, 2021

<< Day 1 | Day 3 >>

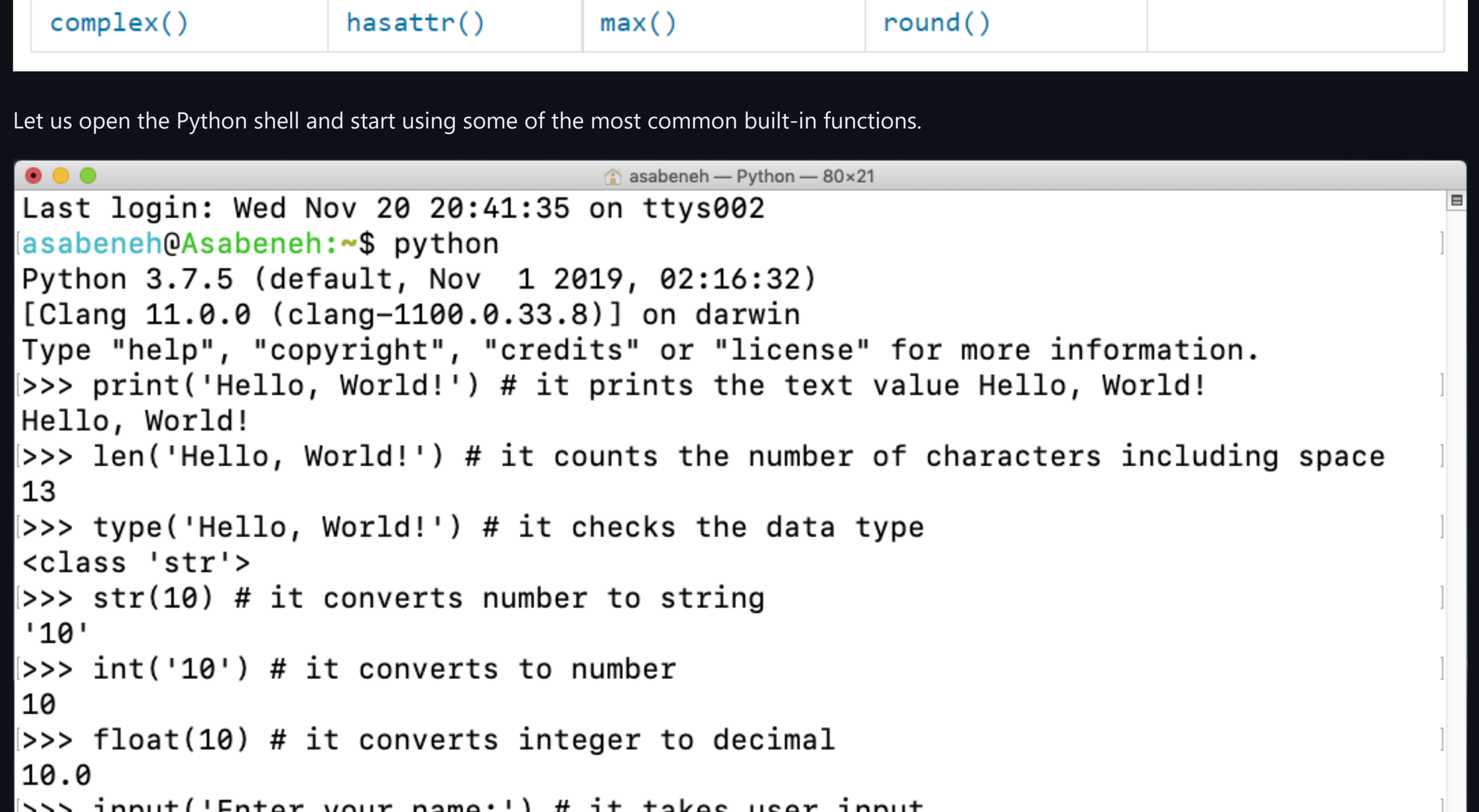
📘 Day 2

Built in functions

In Python we have lots of built-in functions. Built-in functions are globally available for your use that mean you can make use of the built-in functions without importing or configuring. Some of the most commonly used Python built-in functions are the following: *print()*, *len()*, *type()*, *int()*, *float()*, *str()*, *input()*, *list()*, *dict()*, *min()*, *max()*, *sum()*, *sorted()*, *open()*, *file()*, *help()*, and *dir()*. In the following table you will see an exhaustive list of Python built-in functions taken from python documentation.

		Built-in Functions		
abs()	delattr()	hash()	memoryview()	set()
all()	dict()	help()	min()	setattr()
any()	dir()	hex()	next()	slice()
ascii()	divmod()	id()	object()	sorted()
bin()	enumerate()	input()	oct()	staticmethod()
bool()	eval()	int()	open()	str()
breakpoint()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	

Let us open the Python shell and start using some of the most common built-in functions.



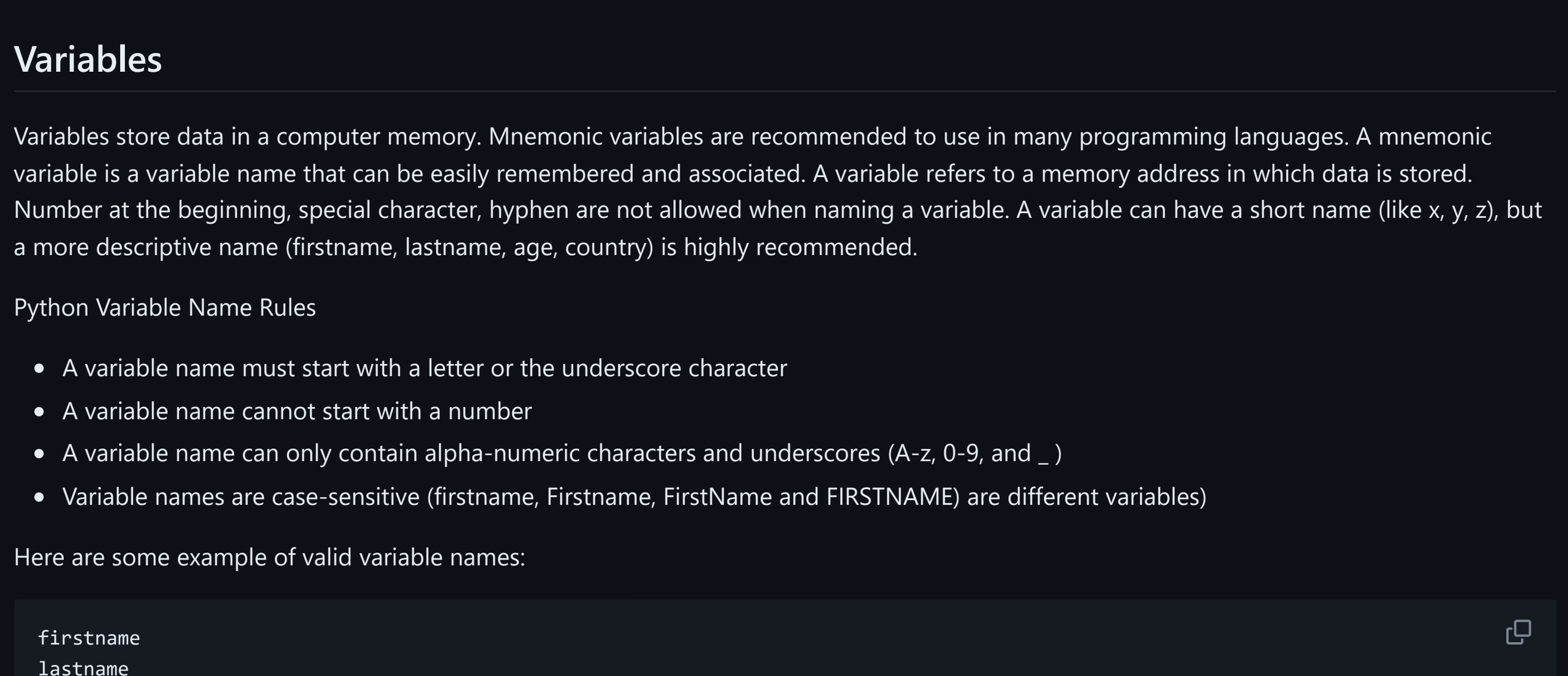
```
Last login: Wed Nov 20 20:41:35 on ttys002
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov  1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello, World!') # it prints the text value Hello, World!
Hello, World!
>>> len('Hello, World!') # it counts the number of characters including space
13
>>> type('Hello, World!') # it checks the data type
<class 'str'>
>>> str(10) # it converts number to string
'10'
>>> int('10') # it converts to number
10
>>> float(10) # it converts integer to decimal
10.0
>>> input('Enter your name:') # it takes user input
Enter your name:Asabeneh
'Asabeneh'
>>> exit()
```

Let us practice more by using different built-in functions

```
Last login: Wed Nov 20 20:42:55 on ttys002
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov  1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> help('keywords') # prints all python reserved words

Here is a list of the Python keywords.  Enter any keyword to get more help.

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

>>> help(str) # give informaton about string

>>> dir(str) # give information about string
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '_
_eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs
__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__le__'
, '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce__',
'__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__'
, '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'count', 'e
ncode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'index', 'isal
num', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'islower', '
isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rindex', 'rjust',
 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startswith', 'strip',
 'swapcase', 'title', 'translate', 'upper', 'zfill']
>>>
```

As you can see from the terminal above, Python has got reserved words. We do not use reserved words to declare variables or functions. We will cover variables in the next section.

I believe, by now you are familiar with built-in functions. Let us do one more practice of built-in functions and we will move on to the next section.



```
Last login: Wed Nov 20 21:12:11 on ttys000
asabeneh@Asabeneh:~$ python
Python 3.7.5 (default, Nov  1 2019, 02:16:32)
[Clang 11.0.0 (clang-1100.0.33.8)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> min(20, 30, 40, 50) # gives the minimum value
20
>>> max(20, 30, 40, 50) # gives the maximum value
50
>>> min([20, 30,40,50]) # it takes list as an argument and return min
20
>>> max([20, 30,40,50]) # it takes list as an argument and return max
50
>>> sum([20, 30,40,50]) # it takes only list as an argument and return the sum
140
>>> exit()
```

Variables

Variables store data in a computer memory. Mnemonic variables are recommended to use in many programming languages. A mnemonic variable is a variable name that can be easily remembered and associated. A variable refers to a memory address in which data is stored. Number at the beginning, special character, hyphen are not allowed when naming a variable. A variable can have a short name (like x, y, z), but a more descriptive name (firstname, lastname, age, country) is highly recommended.

Python Variable Name Rules

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (firstname, Firstname, FirstName and FIRSTNAME) are different variables)

Here are some example of valid variable names:

```
firstname
lastname
age
country
city
first_name
last_name
capital_city
_if # if we want to use reserved word as a variable
year_2021
year2021
current_year_2021
birth_year
num1
num2
```

Invalid variables names

```
first-name
first@name
first$name
num-1
1num
```

We will use standard Python variable naming style which has been adopted by many Python developers. Python developers use snake case(snake_case) variable naming convention. We use underscore character after each word for a variable containing more than one word(eg. first_name, last_name, engine_rotation_speed). The example below is an example of standard naming of variables, underscore is required when the variable name is more than one word.

When we assign a certain data type to a variable, it is called variable declaration. For instance in the example below my first name is assigned to a variable first_name. The equal sign is an assignment operator. Assigning means storing data in the variable. The equal sign in Python is not equality as in Mathematics.

Example:

```
# Variables in Python
first_name = 'Asabeneh'
last_name = 'Yetayeh'
country = 'Finland'
city = 'Helsinki'
age = 250
is_married = True
skills = ['HTML', 'CSS', 'JS', 'React', 'Python']
person_info = {
   'firstname':'Asabeneh',
   'lastname':'Yetayeh',
   'country':'Finland',
   'city':'Helsinki'
   }
```

Let us use the *print()* and *len()* built-in functions. Print function takes unlimited number of arguments. An argument is a value which we can be passed or put inside the function parenthesis, see the example below.

Example:

```
print('Hello, World!') # The text Hello, World! is an argument
print('Hello',',', 'World','!') # it can take multiple arguments, four arguments have been passed
print(len('Hello, World!')) # it takes only one argument
```

Let us print and also find the length of the variables declared at the top:

Example:

```
# Printing the values stored in the variables

print('First name:', first_name)
print('First name length:', len(first_name))
print('Last name: ', last_name)
print('Last name length: ', len(last_name))
print('Country: ', country)
print('City: ', city)
print('Age: ', age)
print('Married: ', is_married)
print('Skills: ', skills)
print('Person information: ', person_info)
```

Declaring Multiple Variable in a Line

Multiple variables can also be declared in one line:

Example:

```
first_name, last_name, country, age, is_married = 'Asabeneh', 'Yetayeh', 'Helsink', 250, True

print(first_name, last_name, country, age, is_married)
print('First name:', first_name)
print('Last name: ', last_name)
print('Country: ', country)
print('Age: ', age)
print('Married: ', is_married)
```

Getting user input using the *input()* built-in function. Let us assign the data we get from a user into first_name and age variables. **Example:**

```
first_name = input('What is your name: ')
age = input('How old are you? ')

print(first_name)
print(age)
```

Data Types

There are several data types in Python. To identify the data type we use the *type* built-in function. I would like to ask you to focus on understanding different data types very well. When it comes to programming, it is all about data types. I introduced data types at the very beginning and it comes again, because every topic is related to data types. We will cover data types in more detail in their respective sections.

Checking Data types and Casting

- Check Data types: To check the data type of certain data/variable we use the *type* **Example:**

```
# Different python data types
# Let's declare variables with various data types

first_name = 'Asabeneh'     # str
last_name = 'Yetayeh'       # str
country = 'Finland'         # str
city= 'Helsinki'            # str
age = 250                   # int, it is not my real age, don't worry about it

# Printing out types
print(type('Asabeneh'))     # str
print(type(first_name))     # str
print(type(10))             # int
print(type(3.14))           # float
print(type(1 + 1j))         # complex
print(type(True))           # bool
print(type([1, 2, 3, 4]))     # list
print(type({'name':'Asabeneh','age':250, 'is_married':250}))    # dict
print(type((1,2)))                                              # tuple
print(type(zip([1,2],[3,4])))                                   # set
```

- Casting: Converting one data type to another data type. We use *int()*, *float()*, *str()*, *list*, *set* When we do arithmetic operations string numbers should be first converted to int or float otherwise it will return an error. If we concatenate a number with a string, the number should be first converted to a string. We will talk about concatenation in String section.

  Example:

```
# int to float
num_int = 10
print('num_int',num_int)         # 10
num_float = float(num_int)
print('num_float:', num_float)   # 10.0

# float to int
gravity = 9.81
print(int(gravity))             # 9

# int to str
num_int = 10
print(num_int)                  # 10
num_str = str(num_int)
print(num_str)                  # '10'

# str to int or float
num_str = '10.6'
print('num_int', int(num_str))      # 10
print('num_float', float(num_str))  # 10.6

# str to list
first_name = 'Asabeneh'
print(first_name)               # 'Asabeneh'
first_name_to_list = list(first_name)
print(first_name_to_list)            # ['A', 's', 'a', 'b', 'e', 'n', 'e', 'h']
```

Numbers

Number data types in Python:

1. Integers: Integer(negative, zero and positive) numbers Example: ... -3, -2, -1, 0, 1, 2, 3 ...
2. Floating Point Numbers(Decimal numbers) Example: ... -3.5, -2.25, -1.0, 0.0, 1.1, 2.2, 3.5 ...
3. Complex Numbers Example: 1 + j, 2 + 4j, 1 - 1j

🌕 You are awesome. You have just completed day 2 challenges and you are two steps ahead on your way to greatness. Now do some exercises for your brain and muscles.

💻 Exercises - Day 2

Exercises: Level 1

1. Inside 30DaysOfPython create a folder called day_2. Inside this folder create a file named variables.py
2. Write a python comment saying 'Day 2: 30 Days of python programming'
3. Declare a first name variable and assign a value to it
4. Declare a last name variable and assign a value to it
5. Declare a full name variable and assign a value to it
6. Declare a country variable and assign a value to it
7. Declare a city variable and assign a value to it
8. Declare an age variable and assign a value to it
9. Declare a year variable and assign a value to it
10. Declare a variable is_married and assign a value to it
11. Declare a variable is_true and assign a value to it
12. Declare a variable is_light_on and assign a value to it
13. Declare multiple variable on one line

Exercises: Level 2

1. Check the data type of all your variables using type() built-in function
2. Using the *len()* built-in function, find the length of your first name
3. Compare the length of your first name and your last name
4. Declare 5 as num_one and 4 as num_two
   i. Add num_one and num_two and assign the value to a variable total
   ii. Subtract num_two from num_one and assign the value to a variable diff
   iii. Multiply num_two and num_one and assign the value to a variable product
   iv. Divide num_one by num_two and assign the value to a variable division
   v. Use modulus division to find num_two divided by num_one and assign the value to a variable remainder
   vi. Calculate num_one to the power of num_two and assign the value to a variable exp
   vii. Find floor division of num_one by num_two and assign the value to a variable floor_division
5. The radius of a circle is 30 meters.
   i. Calculate the area of a circle and assign the value to a variable name of *area_of_circle*
   ii. Calculate the circumference of a circle and assign the value to a variable name of *circum_of_circle*
   iii. Take radius as user input and calculate the area.
6. Use the built-in input function to get first name, last name, country and age from a user and store the value to their corresponding variable names
7. Run help('keywords') in Python shell or in your file to check for the Python reserved words or keywords

🎉 CONGRATULATIONS ! 🎉

<< Day 1 | Day 3 >>