Python

Michael

2021.12.5

-1 Compiler

To save the time in our class and memory space in your computer, we will use an online compiler on Programiz. You may access it by clicking this or scan the QR-code below.

0 Does it work?

Try this line below on Programiz → main.py:

- `print("Hello World!")`

Try this line below on Programiz → Shell:

- `1 + 1`

1 What is Python?

Python is a high-level general-purpose programming language. It is created by Guido van Rossum in 1989 when he was looking for a project to keep him occupied in Christmas. It is now one of the most popular programming languages in the world.

1.1 Why Python ?

- Works on different operating systems
- Simple syntax similar to English language
- Runs on interpreter systems, so that code can be executed as soon as it's written
- Hundreds of libraries
- Can be appied to various fields

1.2 What can Python Do ?

- Handle Big data to perform statistic analysis
- Backend (server) of web applications
- AI and Machine Learning
- Programming applications

2 Variables and `print()`

2.1 Variables

There are many data types, one of which is integer, abbreviated `int`, in Python. We will focus on this data type today. Nevertheless, we still list some basic data types here for your further studying:

- Integers
- Floating-Point Numbers
- Complex Numbers
- Strings
- Boolean Type

As a coder, we need "variables" to store some data for further use. For each variable assignment, it would be better to specify its data type. See the example below.

2.2 `print()`

`print()` is a function helping us to display the value of a variable.

2.2.1 Example

Try this on Programiz → main.py and see what happens.

```
a = int(3)
b = int(5)
print(b)
print(a)
```

2.3 Formatted `print()`

We can print a formatted string with `f''` or `f""`.
Inside this string, you can write a Python expression between { and } characters that can refer to variables or literal values.

2.3.1 Example

```
    a = int(3)
    b = str('Michael')
    print(f'The value of a is {a}')
    print(f'my name is {b}')
```

Now, we believe you come familiar with the assignment operator `=` and a function `print()`. Wait... what do I mean by "operator"?

3 Arithmetic and Bitwise Operators

We've learned how to store a value in a variable. However, we would like to play more tricks on those variables.

3.1 Arithmetic Operators

- `+`
- `-`
- `*`
- `//`
- `**`

3.2 Bitwise Operators

Before going into this subsection, we'd better first know what binary representation is.

Binary Representation

In decimal representation, 7050 actually means $7 \times 10^3 + 0 \times 10^2 + 5 \times 10^1 + 0 \times 10^0$.

- Base: __________
- Digits: __________

What if the base is not ten?

Memos

Exercise

- What is the binary representation of 32 ?

- What is the binary representation of 102 ?

3.2.1 List of Bitwise Operators

- `&` : Performs logical conjunction on the corresponding bits of its operands.
- `|` : Performs logical disjunction. For each corresponding pair of bits.
- `^` : Performs exclusive disjunction on the bit pairs
- `>>` : Drop the n bits on the right.
- `<<` : Moves the bits to the left by n bits.

3.3 Assignment Operators

- `=`
- `+=`
- $\vdots$
- `//=`
- `&=`
- $\vdots$
- `<<=`

4 Statements and Conditions

Sometimes, we may want our program do different things based on different conditions.

- `if`
- `else`
- `elif`

Moreover, things are usually complicated so we need some "conjunctions".

- `and`
- `or`

4.1 Exercise

Write a program to determine whether a number is divisible by 7 or 5, between 1500 and 2700. If not, your program need to output whether the number is ”Out of range” or ”Not divisible”.

4.1.1 Hints

1. Store the number in a variable
2. Write a if statement to check whether the number is in range
3. Write a if statement inside the previous one and check if the number is divisible by 7 or 5
4. print anything you want if both statement is true
5. Figure out the rest by yourself !

5 Lists

Lists are used to store multiple items in a single variable.

5.1 Constructing a list

5.2 Accessing Elements in List

We use ”index” to access elements in a list.
the first item in lists has index 0, the second item has index 1 etc.
We can access the third element using `my_list [2]`.
Use `[start:end]` to access a part of the list.

Examples
L = [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
`L[2]` ⇒ 15
`L[-2]` ⇒ 45
`L[2:5]` ⇒ [15, 20, 25]
`L[:3]` ⇒ [5, 10, 15]

`L[6:]` ⇒ [35, 40, 45, 50]

btw, you can print out the whole list by `print (my_list)`

5.3 Add or Remove Elements

- Use `append()` to add element to the end of the list.
  e.g. `my_list.append(50)`
- Use `insert()` to add element to a specific index of the list.
  e.g. `my_list.insert(i, elem)`
- Use `remove()` to remove an element in the list.
  e.g. `my_list.remove(60)`
- Use `pop()` to remove an element in a specific index.
  e.g. `my_list.pop(1)`

5.4 Some Functions of Lists

- `len([5, 3, 1])` ⇒ 3
- `max([1, 2, 3, 4, 5])` ⇒ 5
- `min([0, 55, 3, 75])` ⇒ 0
- `sum([1, 2, 3, 4, 5])` ⇒ 15

6 For Loop and While Loop

6.1 for loop

We can use for loops to make our program do repetitive things

```
for i in range (0, 5):
   (Do something)
```

You could also use it to iterate through a list

```
L = [5, 2, 88]
for i in L :
   print (i)
```

(Try on your computer to see what happens)

6.2 while loop

Execute a set of statements as long as a condition is true.

```
i = 0
while (i < 5):
   print (i)
   i += 1
```

6.3 `break()` and `continue`

Use `break()` to break out of a loop.

Use `continue` to skip rest of the code and start a new iteration.

6.4 Exercise

7 Functions

A function is a block of code that runs when it's called.
You can pass parameters into a function, it'll do some calculation and return the value to you.
We've already used functions before ! (recall functions of lists)
It's great to use functions so that you don't have to copy and paste your code everywhere.

```
def myFunction(num1, num2) :
    return num1**2 + num2**2
```

8 Conclusion

Now that you've learned the basic syntax for Python,
you can explore various packages for Python !
For example, numpy, pandas, matplotlib, scipy...

Have fun !