Hello, world!

April 7, 2020

This is a first simple notebook showing how to print to console and how to read from it.

---

1 I/O to console

1.1 Print to console

Let's try to be polite

```
[1]: print('Hello, world!')
```

```
Hello, world!
```

When printing to console, we can concatenate more sentences

```
[2]: print('Hello, world!', 'What a beautiful day.', 'Indeed.')
```

```
Hello, world! What a beautiful day. Indeed.
```

1.2 Read from console

What if we want to send personal greetings!? :)

```
[3]: print('What is your name?')
     name=input()
     print('Hello,', name)
```

```
What is your name?
Andrea
Hello, Andrea
```

The text is presented in a weird order when waiting for the user's input (the textbox to write is above the question). This solution offers a better result.

```
[4]: name=input('What is your name? ')
     print('Hello,', name)
```

```
What is your name? Andrea
Hello, Andrea
```

[5]:
```python
sign=input('What is your zodiac sign?')
```

```
What is your zodiac sign?Sagittarius
```

[6]:
```python
print('I foresee that in the coming months',sign,'will learn Python.')
```

```
I foresee that in the coming months Sagittarius will learn Python.
```

2 Variables

Note that **name** and **sign** are **variables** that will store the read inputs.

[7]:
```python
print('Hello again,', name)
```

```
Hello again, Andrea
```

[8]:
```python
print('Dear',name,'I foresee that in the coming months',sign,'will learn Python.
 ↪')
```

```
Dear Andrea I foresee that in the coming months Sagittarius will learn Python.
```

Every variable has a *name*. This name can start with a letter, a __,

[9]:
```python
n=5
print(n)
```

```
5
```

[10]:
```python
_n=5
print(_n)
```

```
5
```

Variables can contain other characters, but not as first one.

[11]:
```python
n1=5
print(n1)
```

```
5
```

[12]:
```python
1n=5
print(1n)
```

```
  File "<ipython-input-12-7a3b216150d8>", line 1
    1n=5
```

```
            ^
SyntaxError: invalid syntax
```

---

3 Next week...

Next week we will see that every variable has a *type*. Actually, everything has a type in Python. Everything is an *object* ...

```
[13]:  n1 = 5
       n2 = 'Ciao'
       n3 = 1e3
       n4 = [1,2,3]
       n5 = (1,2,3)
```

```
[ ]:
```