

IMD0033 - Probabilidade

Aula 03 - Introdução a Python I

Ivanovitch Silva
Julho, 2017



Agenda

- Versões do Python
- Tipos básicos
- Lista
- Arquivos e estruturas de repetição
- Estruturas condicionais
- Dicionários
- Funções e módulos

Atualizar o repositório

```
git clone https://github.com/ivanovitchm/IMD0033_Probabilidade.git
```

Ou

```
git pull
```



Version 3.x (<https://www.python.org/downloads/>)

Tipos básicos

Inteiros, ponto flutuante, tipos lógicos e caracter

```
x = 3
print(type(x)) # Prints "<class 'int'>"
print(x)       # Prints "3"
print(x + 1)   # Addition; prints "4"
print(x - 1)   # Subtraction; prints "2"
print(x * 2)   # Multiplication; prints "6"
print(x ** 2)  # Exponentiation; prints "9"
x += 1
print(x)       # Prints "4"
x *= 2
print(x)       # Prints "8"
y = 2.5
print(type(y)) # Prints "<class 'float'>"
print(y, y + 1, y * 2, y ** 2) # Prints "2.5 3.5 5.0 6.25"
```

Tipos básicos

Inteiros, ponto flutuante, tipos lógicos e caracter

```
hello = 'hello'           # String literals can use single quotes
world = "world"           # or double quotes; it does not matter.
print (hello)             # Prints "hello"
print (len(hello))        # String length; prints "5"
hw = hello + ' ' + world  # String concatenation
print(hw)                 # prints "hello world"
hw12 = '%s %s %d' % (hello, world, 12) # sprintf style string formatting
print(hw12)               # prints "hello world 12"
```

Tipos básicos

Inteiros, ponto flutuante, tipos lógicos e caracter

```
t = True
f = False
print(type(t)) # Prints "<class 'bool'>"
print(t and f) # Logical AND; prints "False"
print(t or f)  # Logical OR; prints "True"
print(not t)   # Logical NOT; prints "False"
print(t != f)  # Logical XOR; prints "True"
```

Listas

```
# Create the areas list
areas = ["hallway", 11.25, "kitchen", 18.0, "living room",
        20.0, "bedroom", 10.75, "bathroom", 9.50]

# Print out areas
print(areas)

# Print out the type of areas
print(type(areas))

# Print out second element from areas
print(areas[1])

# Print out last element from areas
print(areas[-1])

# Print out the area of the living room
print(areas[-5])

# Add two new elements to the end of the list
areas.append("laundry")
areas.append(8.75)
```


Arquivos e estruturas de repetição

```
#open the file  
f = open("crime_rates.csv", "r")  
  
#read the file  
data = f.read()  
  
#print data  
print(type(data))  
print(data)
```

Arquivos e estruturas de repetição

```
#split the crime_rates.csv based on '\n' filter  
rows = data.split('\n')  
  
#print the first five rows  
print(rows[0:5])
```

Arquivos e estruturas de repetição

```
#create an empty list  
int_crime_rates=[]  
  
#print the rate of crimes for each city using a list(int)  
for i in rows:  
    int_crime_rates.append(int(i.split(",")[1]))
```

Estruturas condicionais

```
found = False
for city in cities:
    if city == 'João Pessoa':
        found = True
```

Estruturas condicionais

```
value = 1500
if value > 500:
    if value > 1000:
        print("This number is HUGE!")
```



```
index.js
import React, { useState } from 'react';
import './index.css';

function App() {
  const [contacts, setContacts] = useState([
    { name: 'John Doe', phone: '123-456-7890' },
    { name: 'Jane Smith', phone: '987-654-3210' }
  ]);

  const handleClick = () => {
    // TODO: Add new contact logic
  };

  return (
    <div>
      <h1>Contact List</h1>
      <ul>
        {contacts.map(contact => (
          <li>{contact.name} {contact.phone}</li>
        ))}
      </ul>
      <button onClick={handleClick}>Add New Contact</button>
    </div>
  );
}

export default App;
```

```
index.html
<!DOCTYPE html>
<html>
  <head>
    <script src="index.js"></script>
  </head>
  <body>
    <div></div>
  </body>
</html>
```

Dicionários

(chave,valor)

```
# From string in countries and capitals, create dictionary europe
europe = {'spain':'madrid', 'france':'paris', 'germany':'berlin', 'norway':'oslo'}

# Print europe
print(europe)

# Print out the keys in europe
print(europe.keys())

# Print out value that belongs to key 'norway'
print(europe['norway'])
```

Dicionários de Dicionários

```
# Dictionary of dictionaries
```

```
europa = { 'spain': { 'capital': 'madrid', 'population': 46.77 },  
           'france': { 'capital': 'paris', 'population': 66.03 },  
           'germany': { 'capital': 'berlin', 'population': 80.62 },  
           'norway': { 'capital': 'oslo', 'population': 5.084 } }
```

```
# Print out the capital of France
```

```
print(europa['france']['capital'])
```

```
# Create sub-dictionary data
```

```
data = { 'capital': 'rome', 'population': 59.83 }
```

```
# Add data to europa under key 'italy'
```

```
europa['italy'] = data
```


Estruturas de repetição sobre Dicionários

```
world = { "afghanistan":30.55,  
          "albania":2.77,  
          "algeria":39.21 }  
  
for key, value in world.items() :  
    print(key + " -- " + str(value))
```

Funções

```
def sign(x):  
    if x > 0:  
        return 'positive'  
    elif x < 0:  
        return 'negative'  
    else:  
        return 'zero'  
  
for x in [-1, 0, 1]:  
    print(sign(x))
```

Módulos

```
#Import the math module as m
```

```
import math as m
```

```
#Use the sqrt() function from the math module
```

```
root = m.sqrt(33)
```

```
print(root)
```

Módulos

```
import csv
```

Next, we open the file:

```
f = open("my_data.csv")
```

Then, we call the module's `reader` function:

```
csvreader = csv.reader(f)
```

Finally, we convert the result to a list:

```
my_data = list(csvreader)
```



```
index.js
import React, { useState } from 'react';
import './index.css';

const App = () => {
  const [contacts, setContacts] = useState([
    { name: 'John Doe', phone: '1234567890', email: 'john.doe@example.com' },
    { name: 'Jane Smith', phone: '9876543210', email: 'jane.smith@example.com' },
  ]);

  const [name, setName] = useState('');
  const [phone, setPhone] = useState('');
  const [email, setEmail] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    setContacts([...contacts, { name, phone, email }]);
    setName('');
    setPhone('');
    setEmail('');
  };

  return (
    <div>
      <h1>Contact Manager</h1>
      <div>
        <input type="text" value={name} onChange={e => setName(e.target.value)} />
        <input type="text" value={phone} onChange={e => setPhone(e.target.value)} />
        <input type="text" value={email} onChange={e => setEmail(e.target.value)} />
        <button onClick={handleSubmit}>Add Contact</button>
      </div>
      <ul>
        {contacts.map(contact => (
          <li>
            {contact.name} {contact.phone} {contact.email}
            <button onClick={() => setContacts(contacts.filter(c => c !== contact))}>Remove</button>
          </li>
        ))}
      </ul>
    </div>
  );
};

export default App;
```

```
index.html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title>Contact Manager</title>
  </head>
  <body>
    <div>
      <h1>Contact Manager</h1>
      <div>
        <input type="text" />
        <input type="text" />
        <input type="text" />
        <button>Add Contact</button>
      </div>
      <ul>
        <li></li>
      </ul>
    </div>
  </body>
</html>
```