

Relatório do primeiro Sprint

Python Basics, Expressões Regulares e Manipulação de Arquivos

Daniel Gomes, Érica Costa, Marcela Malaquias, Paulo Vicktor Felix

25 de Abril de 2017

Resumo

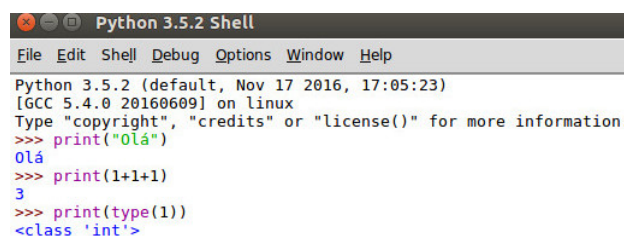
Neste sprint, o grupo devia aprender:

- **Básicos de python:** suas principais funções e funcionamento da linguagem
- **Expressões Regulares:** busca de padrões em textos e suas correspondências
- **Manipulação de Arquivos:** leitura e criação

Além disso, relatar o desenvolvimento do processo de aprendizagem, relacionando-o ao objetivo final do trabalho. Assim, concluir o relatório e fazer *upload* para o Git.

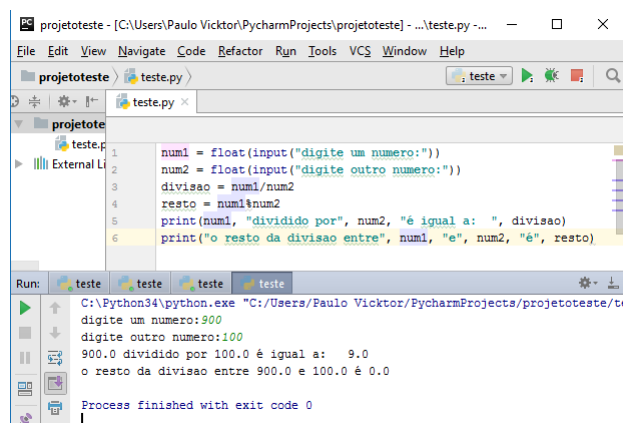
1 Básicos de Python

Iniciando com funções simples de saída (Figura 1), operações matemáticas (Figura 2) e entrada de dados (Figura 3), buscaremos aprender a utilizar as ferramentas disponíveis pela linguagem. Para isso, foram utilizadas video-aulas e livros de programação para que fosse possível agregar o conhecimento de programação.



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "copyright", "credits" or "license()" for more information.
>>> print("Olá")
Olá
>>> print(1+1)
3
>>> print(type(1))
<class 'int'>
```

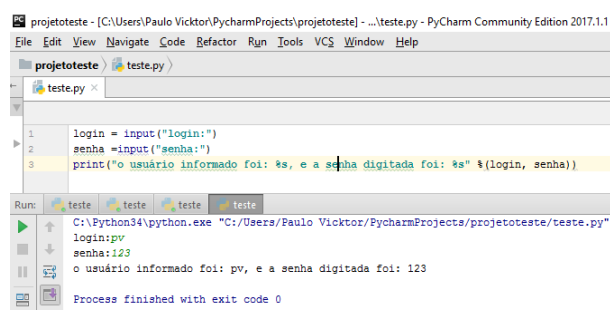
Figura 1: Uso da função print() e type().



```
projototeste - [C:\Users\Paulo Victor\PycharmProjects\projototeste] - ...teste.py - ...
File Edit View Navigate Code Refactor Run Tools VCS Window Help
projototeste > teste.py
1 num1 = float(input("digite um numero:"))
2 num2 = float(input("digite outro numero:"))
3 divisao = num1/num2
4 resto = num1%num2
5 print(num1, "dividido por", num2, "é igual a: ", divisao)
6 print("o resto da divisao entre", num1, "e", num2, "é", resto)

Run: teste teste teste teste
C:\Python34\python.exe "C:/Users/Paulo Victor/PycharmProjects/projetoteste/t...
digite um numero:900
digite outro numero:100
900.0 dividido por 100.0 é igual a: 9.0
o resto da divisao entre 900.0 e 100.0 é 0.0
Process finished with exit code 0
```

Figura 2: Uso de operadores matemáticos.



```
projototeste - [C:\Users\Paulo Victor\PycharmProjects\projototeste] - ...teste.py - PyCharm Community Edition 2017.1.1
File Edit View Navigate Code Refactor Run Tools VCS Window Help
projototeste > teste.py
1 login = input("login:")
2 senha = input("senha:")
3 print("o usuário informado foi: %s, e a senha digitada foi: %s" %(login, senha))

Run: teste teste teste teste
C:\Python34\python.exe "C:/Users/Paulo Victor/PycharmProjects/projetoteste/teste.py"
login:pv
senha:123
o usuário informado foi: pv, e a senha digitada foi: 123
Process finished with exit code 0
```

Figura 3: Uso de entrada de dados.

2 Expressões Regulares

Por meio da utilização do livro "Automate the boring stuff with python"[1],

conseguimos aprender a criar scripts para reconhecimento de padrões e criar correspondências por esses padrões. Um exemplo prático de aplicação para essa função é criar um programa que identifica se a string digitada corresponde a um email, por meio da quantidade e tipo dos caracteres (Figura 5).

```

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS
GNU nano 2.5.3 Arquivo: padrao.py

#Exemplo 1, expressões regulares

import re #Importando módulo para Expressões Regulares
string_base = 'olá este é o primeiro sprint para esof'
padrao = re.search('\w\w\w\w', string_base) #Procura as primeiras 4 letras seguidas
print(padrao.group())

Obter Ajuda Gravar Onde está? Recort txt Justificar Pos atual
Sair Ler o arq Substituir Colar txt Lintar Ir p/ linha

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS 86x18
daniel@daniel-ubuntu:~/Documentos/ESOF/PROGS$ python3 padrao.py
este

```

Figura 4: Identificação de um padrão

```

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS
GNU nano 2.5.3 Arquivo: email.py

#
#Exemplo 2, expressões regulares

import re

email = str(input("Digite um email:"))
padrao_email = re.search(r'\w+@\w+\.\w+', email)
if padrao_email.group() == email:
    print("É email")
else:
    print("Não é email")

Obter Ajuda Gravar Onde está? Recort txt Justificar Pos atual
Sair Ler o arq Substituir Colar txt Lintar Ir p/ linha

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS 80x8
daniel@daniel-ubuntu:~/Documentos/ESOF/PROGS$ python3 email.py
Digite um email:marcelo@ufu.br
marcelo@ufu.br
É email
daniel@daniel-ubuntu:~/Documentos/ESOF/PROGS$

```

Figura 5: Verificar se é email

Além disso, é possível categorizar em uma lista, por meio dos padrões observados no texto, como na Figura 6, de modo um pouco mais complexo.

```

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS
GNU nano 2.5.3 Arquivo: contato.py

#Exemplo 3, expressões regulares

import re

contactInfo = 'Daniel, 11611EAU017: 9999-0000'
infos = re.search(r'(\w+), (\w+): (\S+)', contactInfo)
nome = infos.group(1)
matricula = infos.group(2)
telefone = infos.group(3)

print(nome)
print(matricula)
print(telefone)

Obter Ajuda Gravar Onde está? Recort txt Justificar Pos atual
Sair Ler o arq Substituir Colar txt Lintar Ir p/ linha

daniel@daniel-ubuntu: ~/Documentos/ESOF/PROGS 80x8
daniel@daniel-ubuntu:~/Documentos/ESOF/PROGS$ python3 contato.py
Daniel
11611EAU017
9999-0000
daniel@daniel-ubuntu:~/Documentos/ESOF/PROGS$

```

Figura 6: Lista de informações

3 Manipulação de arquivos

Nesta parte, o grupo ficou responsável de buscar informações sobre como abrir, criar e editar arquivos em python. Para isso, foi utilizado o livro citado anteriormente, e como resultado do estudo, obtivemos alguns programas.

Na Figura 7 foi criado um arquivo, depois editado e lido. Já na Figura 8, é cri-

```

teste - [C:\Users\Paulo Victor\PycharmProjects\teste] - ...testando.py - PyCharm ...
File Edit View Navigate Code Refactor Run Tools VCS Window Help

teste testando.py
testando.py

1 import os
2 baconFile = open('C:\Users\Paulo Victor\Downloads\hello.txt', 'w')
3 baconFile.write('hello Word!\n')
4 baconFile.close()
5 baconFile = open('C:\Users\Paulo Victor\Downloads\hello.txt', 'a')
6 baconFile.write('bacon is not a vegetable')
7 baconFile.close()
8 baconFile = open('C:\Users\Paulo Victor\Downloads\hello.txt', 'r')
9 content = baconFile.read()
10 print(content)

Run testando
C:\Python34\python.exe "C:\Users\Paulo Victor\PycharmProjects\teste\testando.py"
hello Word!
bacon is not a vegetable
Process finished with exit code 0

```

Figura 7: Manipulando um arquivo

ado um arquivo binário de informações, o qual pode ser acessado futuramente. Sendo possível acessar do mesmo programa ou de outro programa a ser desenvolvido.

4 Referências

[1] SWEIGART, Al. "Automate the boring stuff with python", 2015.

[2] Aula 14 - Expressões Regulares: <https://www.youtube.com/watch?v=aPk2ruhi2Zk>. Acesso em 20 de Abril de 2017.

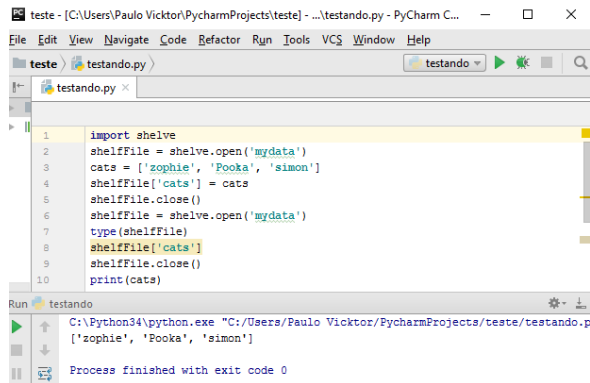


Figura 8: Usando módulo Shelve

E, finalmente, na Figura 9, é criado um arquivo '.py' que pode ser rodado pelo interpretador e convocado em outro programas.

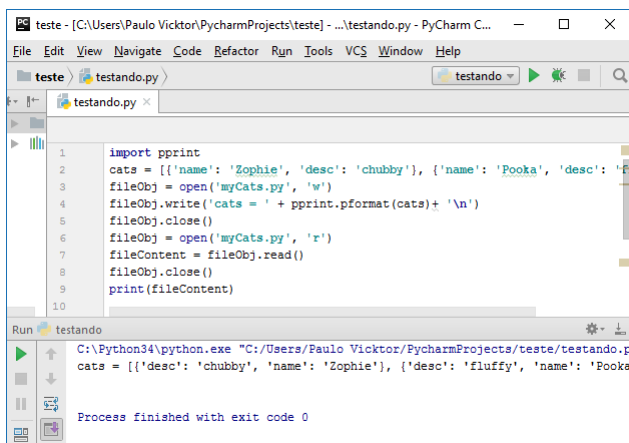


Figura 9: Criando '.py'