1 Antes de Começar

- Indentação faz parte do seu código.
- Dois pontos ":" são usados para iniciar um escopo. Assim, uma indentação com mais espaços começa na linha seguinte ao dois pontos.

```
if <condicao> :
    print("Comando no if")
print("Comando fora do if")
```

- Utilize # para comentar linhas do código.

```
# Isto é um comentário
```

- Utilize aspas simples ('a') ou aspas duplas ("a") para strings.

```
msg = "Hello World!"
msg = 'Hello World!'
```

- Delimite strings contendo aspas simples com aspas duplas e vice-versa.

```
msg = "String com ' "
msg = 'String com " '
```

- Use três aspas simples ou três aspas duplas para strings com quebra de linha.

```
msg = """Hello World! com quebra
de linha e uma aspa simples '  no
meio da frase.    """
```

- Para adicionar acentos no código, adicione uma das linhas a seguir no topo do arquivo.

```
# encoding: utf-8
# encoding: iso-8859-1
```

2 Tipos

Tipos Básicos

- Em python, não precisamos declarar o tipo, apenas atribuir.

```
a = 10      #  int
b = 9.3     #  float
c = True    #  bool
d = False   #  bool
e = "asdf"  #  str
f = b"\xfe" #  bytes
```

- Para obter o tipo da variável, use o comando `type`.

```
x = "asdf"
print(type(x)) ## <type 'str'>
```

- Para saber se a variável é de um dado tipo, use o comando `isinstance`.

```
x = "asdf"
y =  isinstance(x, str)
print(y) ## True
```

Tipos Estruturados

- **Sequências Ordenadas**

```
# encoding: utf-8
## list: podemos mudar os valores
## e acrescentar novos.
seq = ["a", "e", "i", 1, 2]
k = seq[2]       # k recebe "i"
print(type(seq)) # <type 'list'>
seq.append(3)
print(seq) # ['a','e','i',1,2,3]

## tuple: sequência imutável
seq = ("a", "e", "i", 1, 2)
k = seq[2]       # k recebe "i"
print(type(seq)) # <type 'tuple'>
```

- **Dicionários**

```
## dict: conjunto de pares
## chave : valor
dic = {
    "key"   : "value",
    1       : "one",
    3.14    : "pi",
    "flag"  : True
}
x = dic["key"]   # recebe "value"
print(type(dic)) # <type 'dict'>

# Abaixo acrescentamos
# novos elementos
dic[2]      = "dois"
dic["dois"] = 2
```

- **Conjuntos**

```
# encoding: utf-8
# set: conjunto - os itens não
# estão ordenados dentro do
# conjunto e não há repetição
# de elementos.
conj = set([1,2,3,4,1])

## Note abaixo que 1 não está
# duplicado.
print(conj) # set([1,2,3,4])

## Abaixo acrescentamos um
## novo elemento.
## Observe que não existe
## uma ordenação dos
## elementos inseridos.
conj.add("um")
print(conj) # set(['um',1,2,3,4])
```

3 Operadores

- Operadores aritméticos, lógicos e de comparação.

```
# encoding: utf-8
a = 23.0
b = 7.0

## Operadores Aritméticos
a  +  b # Adição :          30.0
a  -  b # Subtração:        16.0
a  *  b # Multiplicação:    161.0
a  /  b # Divisão:          3.28571428571
a //  b # Divisão Piso:     3.0
a  %  b # Módulo:           2.0
a **  b # Exponenciação:    3404825447.0

## Operadores de Comparação
a  <  b # Menor que:        False
a <=  b # Menor ou igual:   False
a ==  b # igual:            False
a  >  b # Maior que         True
a >=  b # Maior ou igual    True
a !=  b # Diferente         True
a <>  b # Diferente         True

## Redefinindo variáveis, mas poderíamos
## continuar usando valores inteiros.
a = True
b = False

## Operadores Lógicos
a  and  b  # False
a   or  b  # True
not  a     # False
```

4 Sintaxe Básica

- **Primeiro programa**

```
msg = "Hello World!"
print(msg)
```

- **Argumentos da linha de comando**

```
import sys
msg = sys.argv[1]
print("Recebido: %s" % msg)
```

- **Comando IF**

```
# encoding: utf-8
import sys

## Recebendo input do usuário
flag = sys.argv[1]
if flag == "cmd1" :
    print("Usuário escreveu cmd1")
elif flag == "cmd2" :
    print("Usuário escreveu cmd2")
else :
    print("Outros inputs")
```

- **Comando WHILE**

```
count  = 0
while count < 10 :
    print("Count = %i" % count)
    count = count + 1
```

- **Comando FOR**

```
# encoding: utf-8
seq = ["a", "e", "i", "o", "u"]
for el in seq :
    # Faça algo com cada elemento
    # da sequência.
    print(el)

for el in range(10) :
    # Faça algo 10 vezes
    # el varia de 0 a 9
    print(el)

for el in range(5,10) :
    # Faça algo 5 vezes
    # el varia de 5 a 9
    print(el)

dic = {
    "key"   : "value",
    1       : "one",
    3.14    : "pi",
    "flag"  : True
}
for key in dic.keys() :
    print("%s:%s" % (key,
                     dic[key])
          )
# Output
# 1:one
# flag:True
# key:value
# 3.14:pi
```

- **Funções**

```
# encoding: utf-8
## Criando funções
def hello_world() :
    print("Hello World!")

hello_world() ## Invocando função

## Passando argumentos
def hello_world(user) :
    print("Hello %s!" % user)

hello_world("Ulisses")

## Argumentos default
def hello_world(user = "World") :
    print("Hello %s!" % user)

hello_world()
hello_world("Ulisses")

## Retornando valores
def soma_media(a, b) :
    soma  = a + b
    media = float(soma) / 2
    return soma, media

s, m = soma_media(5,10)
print("S = %s, M = %s" % (s,m))
```

5 Tratamento de Exceções

- **try ... except ... else**

```
import sys
try :
    ## Lendo o que inserido na
    ## linha de comando
    inserido = int(sys.argv[1])
    fraction = 100 / inserido
except IndexError :
    print("Insira alguma coisa")
except ValueError :
    print("Insira um inteiro")
except ZeroDivisionError :
    print("Insira diferente de zero")
else :
    print("Inserido corretamente")
```

6 Strings

- **Acessando characteres**

```
my_string = "Hello World!"
my_string[4] # Retorna "o"
```

- **Iterando sobre os elementos**

```
my_string = "Hello World"
for el in mystring :
    print(el)
```

- **Dividindo strings em arrays** (`split`)

```
my_string = "Hello World!"
## Quebra a string em cada
## character do tipo passado
## como parâmetro

## Este caso quebra a string no
## espaço
x = my_string.split(" ")
print(x) # ['Hello', 'World!']

## Este caso quebra a string na
## letra "r"
x = my_string.split("r")
print(x) # ['Hello Wo', 'ld!']
```

- **Unindo arrays em strings** (`join`)

```
x =  ["Hello", "World!"]

## cria uma string colocando
## um espaço entre  os
## elementos do array

## Este caso une a lista
## com espaços.
" ".join(x) ## Hello World!

## Este caso une a lista
## com o character "-".
"-".join(x) ## Hello-World!
```

- **Formatando strings**

```
str1 = "Ulisses"
str2 = "Tecnologia"
str3 = "Foco"
str4 = "Faculdade de Tecnologia"

my_string = """Hello %s!
Welcome to %s em %s
%s
""" % (str1, str2, str3, str4)

# Hello Ulisses!
# Welcome to Tecnologia em Foco
# Faculdade de Tecnologia
#
```

- **Fatiando strings (slice)**

```
my_string = "Hello World"
print(my_string[1:4]) # ell

my_list = ["a","e","i","o","u"]
print(my_list[1:4]) #["e","i","o"]
```

- **Verificando Pertinência**

```
my_string = "Hello World"
print("o" in my_string) # True
print("b" in my_string) # False

my_list = ["a","e","i","o","u"]
print("o" in my_list) # True
print("b" in my_list) # False

my_tuple = ("a","e","i","o","u")
print("o" in my_tuple) # True
print("b" in my_tuple) # False

my_set = set(["a","e","i","o","u"])
print("o" in my_set) # True
print("b" in my_set) # False

my_dic   = {"a" : 1,
            "e" : 2,
            "i" : 3,
            "o" : 4,
            "u" : 5
            }
print("o" in my_dic) # True
print("b" in my_dic) # False
print(2   in my_dic) # False
```

7 Arquivos

- **Leitura**

```
fobject = open("docentes.txt",'r')
file_array  = []
for line in fobject :
    line = line.rstrip()
    file_array.append(line)
print(file_array)
```

- **Escrita**

```
fobject = open("docentes.txt", 'w')
fobject.write("Ulisses Dias\n")
```

8 Expressões Regulares

Comandos Básicos

^	Início de uma linha
$	Final de uma linha
.	Qualquer character
\s	Character de espaçamento
\S	Character diferente de espaçamento
*	Repete character (zero ou mais)
*?	Mesmo do anterior, mas não guloso
+	Repete character (uma ou mais)
+?	Mesmo do anterior, mas não guloso
[aeiou]	Qualquer character do conjunto
[a-z]	Qualquer character no intervalo)
[^XYZ]	Apenas character fora do conjunto
(	Início da extração de uma string
)	Final da extração de uma string

Pacote re

- **Validação**

```
import re

my_emails = open("emails.txt")

for l in my_emails :
    ## Procura linhas que começam
    ## com "From:" e tenham algum
    ## @ no meio.
    if re.search('^From:.*@', l) :
        print(l)
```

- **Busca**

```
import re

x = """Meus 3 números favoritos
 são 7, 18 e 19
"""
y = re.findall('[0-9]+', x)
print(y) # ['3', '7', '18', '19']
```

- **Extração**

```
import re

x = """
From: secgrad@ft.unicamp.br 22
Thursay, September 28 2017
From: ulisses@ft.unicamp.br 23
Friday, September 29 2017
From: informatica@ft.unicamp.br 24
Saturday, September 30 2017
"""
y = re.findall('From: (\S+@\S+)',x)
print(y)
# ['secgrad@ft.unicamp.br',
# 'ulisses@ft.unicamp.br',
# 'informatica@ft.unicamp.br']
```

9 Web

- **urllib**

```
import urllib

fweb = urllib.urlopen(
    'http://www.ft.unicamp.br'
   )
for line in fweb :
    print(line.strip())
```

- **beatifulsoup**

```
# encoding: utf-8
import sys
import urllib
from BeautifulSoup import
 ↪  BeautifulSoup

## Note o comando "read"
fweb = urllib.urlopen(
    sys.argv[1] ## linha de comando
   ).read()
soup = BeautifulSoup(fweb)
urls = soup('a') # links
for url in urls :
    print(url.get('href', None))
```

- **XML Element Tree**

```
import xml.etree.ElementTree as ET

xml = """
<users>
  <docentes id = "2">
    <name>Ulisses</name>
    <place>Unicamp</place>
  </docentes>
</users>
"""

users = ET.fromstring(xml)
doc   = users.findall('docentes')
print(len(doc))

for item in doc :
    print(item.find('name').text)
    print(item.get('id'))
```

- **json**

```
import json
json_str = """
{
 "logradouro":"R. Paschoal Marmo",
 "bairro":"Jardim Piratininga",
 "localidade": "Limeira",
 "uf": "SP"
}
"""
try :
    js = json.loads(json_str)
except :
    js = None
bairro = js["bairro"]
uf     = js["uf"]
print(bairro)
```