# MAP 2112 – Introdução à Lógica de Programação e Modelagem Computacional

**1º Semestre - 2023**

**Prof. Dr. Luis Carlos de Castro Santos**

lsantos@ime.usp.br

# Python 3 Beginner's Reference Cheat Sheet

Alvaro Sebastian
http://www.sixthresearcher.com

## Main data types

boolean = *True* / *False*
integer = 10
float = 10.01
string = "123abc"
list = [ value1, value2, ... ]
dictionary = { key1:value1, key2:value2, ...}

## Numeric operators

| | |
|---|---|
| + | addition |
| - | subtraction |
| * | multiplication |
| / | division |
| ** | exponent |
| % | modulus |
| // | floor division |

## Comparison operators

| | |
|---|---|
| == | equal |
| != | different |
| > | higher |
| < | lower |
| >= | higher or equal |
| <= | lower or equal |

## Boolean operators

| | |
|---|---|
| and | logical AND |
| or | logical OR |
| not | logical NOT |

## Special characters

| | |
|---|---|
| # | coment |
| \n | new line |
| \<char> | scape char |

## String operations

| | |
|---|---|
| string[i] | retrieves character at position i |
| string[-1] | retrieves last character |
| string[i:j] | retrieves characters in range i to j |

## List operations

| | |
|---|---|
| list = [] | defines an empty list |
| list[i] = x | stores x with index i |
| list[i] | retrieves the item with index I |
| list[-1] | retrieves last item |
| list[i:j] | retrieves items in the range i to j |
| del list[i] | removes the item with index i |

## Dictionary operations

| | |
|---|---|
| dict = {} | defines an empty dictionary |
| dict[k] = x | stores x associated to key k |
| dict[k] | retrieves the item with key k |
| del dict[k] | removes the item with key k |

## String methods

| | |
|---|---|
| string.upper() | converts to uppercase |
| string.lower() | converts to lowercase |
| string.count(x) | counts how many times x appears |
| string.find(x) | position of the x first occurrence |
| string.replace(x,y) | replaces x for y |
| string.strip(x) | returns a list of values delimited by x |
| string.join(L) | returns a string with L values joined by string |
| string.format(x) | returns a string that includes formatted x |

## List methods

| | |
|---|---|
| list.append(x) | adds x to the end of the list |
| list.extend(L) | appends L to the end of the list |
| list.insert(i,x) | inserts x at i position |
| list.remove(x) | removes the first list item whose value is x |
| list.pop(i) | removes the item at position i and returns its value |
| list.clear() | removes all items from the list |
| list.index(x) | returns a list of values delimited by x |
| list.count(x) | Conta quantas vezes um elemento aparece na lista |
| list.sort() | sorts list items |
| list.reverse() | reverses list elements |
| list.copy() | returns a copy of the list |

## Dictionary methods

| | |
|---|---|
| dict.keys() | returns a list of keys |
| dict.values() | returns a list of values |
| dict.items() | returns a list of pairs (key,value) |
| dict.get(k) | returns the value associtated to the key k |
| dict.pop() | removes the item associated to the key and returns its value |
| dict.update(D) | adds keys-values (D) to dictionary |
| dict.clear() | removes all keys-values from the dictionary |
| dict.copy() | returns a copy of the dictionary |

**Legend:** x,y stand for any kind of data values, s for a string, n for a number, L for a list where i,j are list indexes, D stands for a dictionary and k is a dictionary key.

# Python 3 Beginner's Reference Cheat Sheet

Alvaro Sebastian
http://www.sixthresearcher.com

## Built-in functions

| | |
|---|---|
| print(x, sep='y') | prints x objects separated by y |
| input(s) | prints s and waits for an input that will be returned |
| len(x) | returns the length of x (s, L or D) |
| min(L) | returns the minimum value in L |
| max(L) | returns the maximum value in L |
| sum(L) | returns the sum of the values in L |
| range(n1,n2,n) | returns a sequence of numbers from n1 to n2 in steps of n |
| abs(n) | returns the absolute value of n |
| round(n1,n) | returns the n1 number rounded to n digits |
| type(x) | returns the type of x (string, float, list, dict ...) |
| str(x) | converts x to string |
| list(x) | converts x to a list |
| int(x) | converts x to a integer number |
| float(x) | converts x to a float number |
| help(s) | prints help about x |
| map(function, L) | Applies function to values in L |

## Conditional statements

```
if <condition> :
    <code>
else if <condition> :
    <code>
...
else:
    <code>

if <value> in <list>:
```

## Data validation

```
try:
    <code>
except <error>:
    <code>
else:
    <code>
```

## Working with files and folders

```
import os
os.getcwd()
os.makedirs(<path>)
os.chdir(<path>)
os.listdir(<path>)
```

## Loops

```
while <condition>:
    <code>

for <variable> in <list>:
    <code>

for <variable> in
range(start,stop,step):
    <code>

for key, value in
dict.items():
    <code>
```

## Loop control statements

| | |
|---|---|
| break | finishes loop execution |
| continue | jumps to next iteration |
| pass | does nothing |

## Running external programs

```
import os
os.system(<command>)
```

## Functions

```
def function(<params>):
    <code>
    return <data>
```

## Modules

```
import module
module.function()

from module import *
function()
```

## Reading and writing files

```
f = open(<path>,'r')
f.read(<size>)
f.readline(<size>)
f.close()

f = open(<path>,'r')
for line in f:
    <code>
f.close()

f = open(<path>,'w')
f.write(<str>)
f.close()
```

Legend: x,y stand for any kind of data values, s for a string, n for a number, L for a list where i,j are list indexes, D stands for a dictionary and k is a dictionary key.

P2:

Escreva um script Python que peça ao usuário uma sequência de palavras separadas por vírgula e imprima de volta essa sequência ordenada (ordem alfabética).


BRAIN
STORM

p2_ex01.py

```
1
2   entrada = input("Digite uma sequencia de palavras separadas por vírgula:\n")
3
4   lista_p = entrada.split(",")
5
6   print(lista_p)
7
8   lista_p.sort()
9
10  print(lista_p)
11
12  virgula = ","
13
14  print(virgula.join(lista_p))
```

```
In [32]: runfile('C:/Users/win/Dropbox/USP/2023/remote/MAP2112_2023/scripts/
aula_ex01/p2_ex01.py', wdir='C:/Users/win/Dropbox/USP/2023/remote/
MAP2112_2023/scripts/aula_ex01')
Digite uma sequencia de palavras separadas por vírgula:
pera,maça,abacate,caqui
['pera', 'maça', 'abacate', 'caqui']
['abacate', 'caqui', 'maça', 'pera']
abacate,caqui,maça,pera
```

P3:

Escreva um script Python que calcule a soma todos os itens de uma lista, o valor máximo e o valor mínimo (assumindo que a lista é de valores numéricos) e armazene em outra lista


BRAIN
STORM

p3_ex01.py

```
1
2   def check_lista(lista_num):
3       soma_num = sum(lista_num)
4       max_num = max(lista_num)
5       min_num = min(lista_num)
6       lista_out = [soma_num, max_num, min_num]
7       return lista_out
8
9   lista_teste = [1,2,3,4,5,6,7,8,9,10]
10
11  out = check_lista(lista_teste)
12
13  print("soma, max e min:")
14  print(out)
```

```
In [60]: runfile('C:/Users/win/Dropbox/USP/20
aula_ex01/p3_ex01.py', wdir='C:/Users/win/Drop
scripts/aula_ex01')
soma, max e min:
[55, 10, 1]
```

P4:

Escreva um script Python para remover itens únicos de uma lista que contém itens duplicados .


BRAIN
STORM

p4_ex01.py

```
1
2   lista_a = [10,20,30,20,10,50,60,40,80,50,40]
3
4   unicos = []
5
6   for x in lista_a:
7       if (lista_a.count(x) == 1):
8           unicos.append(x)
9
10
11  print(unicos)
```

```
In [42]: runfile('C:/Users/win/Dropbox/USP/2023/remote/MAP2112_2023/
scripts/aula_ex01/p4_ex01.py', wdir='C:/Users/win/Dropbox/USP/2023/
remote/MAP2112_2023/scripts/aula_ex01')
[30, 60, 80]
```

P5:

Escreva um script Python que verifica se cada inteiro de uma lista é primo. Se todos os inteiros da lista forem primos retorna TRUE se não retorna FALSE.

Por exemplo:

[0, 3, 4, 7, 9] -> False
[3, 5, 7, 13] -> True
[1, 5, 3] -> False


BRAIN
STORM

p5_ex01.py

```python

def test(nums):
    for i in nums:
        if(eh_primo(i) == False):
            return False
    return True

def eh_primo(n):
    if (n==1):
        return False
    elif (n==2):
        return True;
    else:
        for x in range(2,n):
            if(n % x==0):
                return False
            return True

nums = [0, 3, 4, 7, 9]
print("Lista de números de teste:")
print(nums)
print("Resultado do teste:")
print(test(nums))
nums = [3, 5, 7, 13]
print("\nLista de números de teste:")
print(nums)
print("Resultado do teste:")
print(test(nums))
nums = [1, 5, 3]
print("\nLista de números de teste:")
print(nums)
print("Resultado do teste:")
print(test(nums))
```

```
In [47]: runfile('C:/Users/win/D
aula_ex01/p5_ex01.py', wdir='C:/
scripts/aula_ex01')
Lista de números de teste:
[0, 3, 4, 7, 9]
Resultado do teste:
False

Lista de números de teste:
[3, 5, 7, 13]
Resultado do teste:
True

Lista de números de teste:
[1, 5, 3]
Resultado do teste:
False
```

P6:

Escreva um script Python que calcula a média dos elementos de uma tuple composta de tuples e retorna em uma lista.


BRAIN
STORM

p6_ex01.py

```python

def media_tuple(nums):
    result =[]
    for x in nums:
      result.append(sum(x) / len(x))
    return result

nums = ((10, 10, 10, 12), (30, 45, 56, 45), (81, 80, 39, 32), (1, 2, 3, 4))
print ("Tuple Composta Original:")
print(nums)
print("Média das Tuples da Tuple Composta:\n",media_tuple(nums))

nums = ((1, 1, -5), (30, -15, 56), (81, -60, -39), (-10, 2, 3))
print ("\nTuple Composta Original:")
print(nums)
print("Média das Tuples da Tuple Composta:\n",media_tuple(nums))

nums = ((10, 10, 10, 12),(-10, 2, 3))
print ("\nTuple Composta Original:")
print(nums)
print("Média das Tuples da Tuple Composta:\n",media_tuple(nums))
```

```
In [52]: runfile('C:/Users/win/Dropbox/USP/2023/remote/MAP2112_2023/sc
aula_ex01/p6_ex01.py', wdir='C:/Users/win/Dropbox/USP/2023/remote/MAP2
scripts/aula_ex01')
Tuple Composta Original:
((10, 10, 10, 12), (30, 45, 56, 45), (81, 80, 39, 32), (1, 2, 3, 4))
Média das Tuples da Tuple Composta:
 [10.5, 44.0, 58.0, 2.5]

Tuple Composta Original:
((1, 1, -5), (30, -15, 56), (81, -60, -39), (-10, 2, 3))
Média das Tuples da Tuple Composta:
 [-1.0, 23.666666666666668, -6.0, -1.6666666666666667]

Tuple Composta Original:
((10, 10, 10, 12), (-10, 2, 3))
Média das Tuples da Tuple Composta:
 [10.5, -1.6666666666666667]
```

P7:

Dado um dicionário com nomes e médias, crie um novo dicionário apenas com os aprovados dada uma média de aprovação.

Exemplo para teste:

turma = {'Vega': 75, 'Cantrell': 80, 'Gentry': 65, 'Cox': 90}

Média de aprovação = 70


BRAIN
STORM

p7_ex01.py

```
1
2   turma = {'Vega': 75, 'Cantrell': 80, 'Gentry': 65, 'Cox': 90}
3   print("Dicionario Original:")
4   print(turma)
5
6   nota = 70
7   print("Notas acima de ",nota," :")
8
9   aprovados = {}
10
11  for (key, value) in turma.items():
12      if value >= nota:
13          aprovados.update({key: value})
14
15  print(aprovados)
```

```
In [57]: runfile('C:/Users/win/Dropbox/USP/2023/remote/MAP2112
aula_ex01/p7_ex01.py', wdir='C:/Users/win/Dropbox/USP/2023/rem
scripts/aula_ex01')
Dicionario Original:
{'Vega': 75, 'Cantrell': 80, 'Gentry': 65, 'Cox': 90}
Notas acima de  70  :
{'Vega': 75, 'Cantrell': 80, 'Cox': 90}
```

P1:

Escreva um script Python para contar as ocorrências de cada palavra numa frase (entrada por string)


BRAIN
STORM

p1_b_ex01.py

```
1
2   def contapalavras (frase):
3       contagem = {}
4       lista_p = frase.split()
5
6       for p in lista_p:
7           x = lista_p.count(p)
8           contagem.update({p:x})
9       return contagem
10
11
12  teste = "Sei que às vezes uso palavras repetidas , mas quais são as palavras" \
13      + " que nunca são ditas"
14
15  conta = {}
16
17  conta = contapalavras(teste)
18
19  print(conta)
```

```
In [3]: runfile('/Users/lccs13/Library/CloudStorage/Dropbox/USP/2023/
remote/MAP2112_2023/scripts/aula_ex01/p1_b_ex01.py', wdir='/Users/
lccs13/Library/CloudStorage/Dropbox/USP/2023/remote/MAP2112_2023/
scripts/aula_ex01')
{'Sei': 1, 'que': 2, 'às': 1, 'vezes': 1, 'uso': 1, 'palavras': 2,
'repetidas': 1, ',': 1, 'mas': 1, 'quais': 1, 'são': 2, 'as': 1,
'nunca': 1, 'ditas': 1}
```

# Fim Aula Ex01