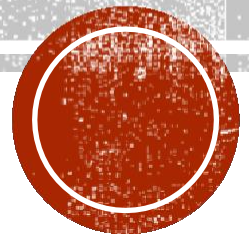
LABORATÓRIO DE PROGRAMAÇÃO

Aula 11 – Strings

Centro Universitário Luterano de Palmas
Departamento de Computação Professor:
Robson Gomes



STRING

- Uma string é como se fosse uma lista de caracteres

```
word = "palavra"
letra = word[3]
print (letra)
```

COMPRIMENTO (O TAMANHO)

- O velho e bom len

```
word = "palavra" comprimento = len(word) ultima = word[comprimento-1]
print(ultima)
```

FOR

```
word = "palavra"

for letra in word: print
(letra)
```

FOR

```
word = "palavra"

for i in range(len(word )): print
    (word[i])
```

BRINCANDO COM SLICING

```
nomes = "Pedro, Paulo e Maria"
print (nomes[0:5])print (nomes[7:12])print (nomes[15:21])
```

E SE QUISER MUDAR UMA LETRA?

```
palavra = "mundo"
palavra[1]="a"
```

Erroooooooooo

E SE QUISER MUDAR UMA LETRA?

```
palavra = "Paumas"
print(palavra)
palavra = palavra.replace("u","l")
print(palavra)
```

E SE QUISER MUDAR MAIS QUE UMA LETRA?

```
palavra = "Paunas "print(palavra)
palavra = palavra.replace("un","lm")
print(palavra)
```

FIND

```
palavra = "orangotango" indice = palavra.find("o")
print("Letra o na posição:", indice)

print("Letra t na posição:", palavra.find("t"))
```

COUNT

```
palavra = "orangotango"
quant = palavra.count("o")
print("Quantidade de \"o\":", quant) print("Quantidade de \"an\":", palavra.count("an"))
```

MAIÚSCULAS

```
palavra = “MAIÚSCULA e minúscula"
print(palavra.upper())
print(palavra)
palavra = palavra.lower()
print(palavra)
```

MAIS MAIÚSCULAS E MINUSCULAS

```
frase = "um Pequeno passo"
print(frase.capitalize())
print(frase.title())
print(frase.swapcase())
print(frase.title().swapcase())
```

```
Um pequeno passo
Um Pequeno Passo UM pEQUENO PASSOuM pEQUENO pASSO
```

ALGUMAS VERIFICAÇÕES

```
frase = input("Digite uma palavra: ")

if frase.islower():
    print("A frase está toda em minúsculas") elif
frase.isupper():
    print("A frase está toda em maiúsculas") elif
frase.istitle():
    print("A frase tem somente as primeiras letras em maiúsculas ")
```

ALGUMAS VERIFICAÇÕES

```
texto = input("Digite um texto: ")

if texto.isalnum():
    print("O texto contém somente letras ou números") else:
    print("O texto contém outros símbolos")
```

ALGUMAS VERIFICAÇÕES

```
texto = input("Digite um texto: ")

if texto.isalpha():
    print("O texto contém somente letras") elif
texto.isdigit():
    print("O texto contém somente números") elif
texto.isalnum():
    print("O texto contém somente letras ou números")
```

ALGUMAS VERIFICAÇÕES

```
texto = "abc"

if texto.startswith("a"): print("Começa
    com a")

if texto.endswith("c"): print("Termina
    com c")


palavra = "bc"

if palavra in texto:

    print(palavra, "está em", texto)
```

TRANSFORMANDO TEXTO EM LISTA

```
texto = "A bc def".split(" ")
lista = texto.split(" ")print
(lista)
```

TRANSFORMANDO TEXTO EM LISTA

```
texto = '111.222.333-44'
lista1 = texto.split('.')
print(lista1)
lista2 = list(texto)
print(lista2)
```

TRANSFORMANDO LISTA EM STRING

```
lista=['A','B','C']
juncao = ' antes de ' novaString = juncao.join(lista) print("Lista.=", lista)
print("Texto =", novaString)
```

ALGORITMOS E PROGRAMAÇÃO II
AULA 4 – DICIONÁRIOS

Centro Universitário Luterano de Palmas

Cursos de Sistemas de Informação e Ciência da Computação

Professores:

Fabiano Fagundes

Edeilson Milhomem da Silva

DICIONÁRIOS EM PYTHON

- Dicionários são estruturas de dados que implementam mapeamentos
- Mapeamento é uma relação entre pares de valores, entre uma chave e seu “conteúdo” ou “significado”.
- Mapeamento é uma generalização da ideia de acessar dados por índices, exceto que em um mapeamento os índices (ou chaves) podem ser de outro tipo além de inteiro

EXEMPLO DE MAPEAMENTO COM LISTAS

```
nome = ['Pedro','Maria','Paulo']

idade = ['21','18','19']

# Pode ser assim:
indice = nome.index('Maria')
print(idade[indice])

# Ou assim, de forma mais direta
print(idade[nome.index('Maria')])


# Se quiser adicionar mais um nome e sua idade

nome.append('Ana') idade.append('20')
# Testando o mapeamento
print(idade[nome.index('Ana')])
```

EXEMPLO DE MAPEAMENTO COM DICIONÁRIO

```
# Criando um dicionário e testando
telefone={'Maria':'84400454','Ana':'84565421'}
print(telefone['Maria'])


# Inserindo uma nova associação e testando
telefone['Pedro']='97475485'
print(telefone['Pedro'])
```

CRIANDO UM DICIONÁRIO DO NADA

```
# Criando um dicionário do nada

matricula['Pedro']='033033021'
print(matricula['Pedro'])
# Dá pau sim!!!
```

CRIANDO UM DICIONÁRIO DO NADA – AGORA SIM

```
# Criando um dicionário do nada – Agora da forma correta matricula={}
matricula['Pedro']='033033021'

print(matricula['Pedro'])
```

ALTERANDO VALOR DA ASSOCIAÇÃO

```
print(telefone['Pedro']) print
(telefone)

# Trocando valor da associação

telefone['Pedro'] = '000111222'
print('Depois de alterar:')
print(telefone['Pedro'])
print (telefone)
```

IMPRIMINDO EM ORDEM

```
for i in sorted(telefone): print(i,
    telefone[i])
```

APAGANDO ASSOCIAÇÃO

```
print (telefone)

del telefone['Pedro']
print (telefone)
```

TAMANHO DO DICIONÁRIO

```
#O len também funciona aqui

print (telefone)

print (len(telefone))
```

KEYS E VALUES

```
# keys
print(telefone.keys()) #
values

print(telefone.values())
```

IN E NOT IN

```
if 'Augusto' in telefone:
    print('O telefone de Augusto é:',telefone['Augusto']) else:
    print('Augusto não tem telefone')


if 'Augusto' not in telefone:
    print('Augusto não tem telefone')
else:
    print('O telefone de Augusto é:',telefone['Augusto'])


if 'Pedro' in telefone:
    print('O telefone de Pedro é:',telefone['Pedro']) else:
    print('Pedro não tem telefone')
```

DICIONÁRIO É UMA VARIÁVEL MUTÁVEL

```
antonimos = {'alto':'baixo', 'correto':'errado', 'verdadeiro':'falso'}
referencia = antonimos copia = antonimos.copy()

print('Antonimos:', antonimos)
print('Referencia:',referencia)
print('Copia:',copia)

referencia['correto']='incorreto'

print('Antonimos:', antonimos)
print('Referencia:',referencia)
print('Copia:',copia)
```

GET

```
print (telefone) print(telefone['Pedro'])
print(telefone['Paulo'])
# Deu erro (KeyError) no telefone de Paulo, não foi?
```

GET

```
print (telefone)
print(telefone['Pedro'])
print(telefone['Paulo'])
# Deu erro (KeyError) no telefone de Paulo, não foi?


# Agora tenta assim:

print (telefone) print(telefone.get('Pedro'))
print(telefone.get('Pedro','Não existe'))
print(telefone.get('Paulo','Não existe'))
```

BRINCANDO COM GET

```
contaLetras = {}
for letra in "orangotango":
    contaLetras[letra] = contaLetras.get(letra,0) + 1
print(contaLetras)
```

BRINCANDO COM FOR EM DICIONÁRIOS

```
for item in telefone.keys(): print(item)

for item in telefone.values(): print(item)

for chave in telefone.keys():
    print('Nomes:',chave)

for (chave,valor) in telefone.items():
    print ('Nome:', chave,'tem telefone:',valor)
```

Laboratório de Programação

AULA 12 – ROTEIRO DE DICIONÁRIOS EM PYTHON – 16/10/2023

Para exemplificar os exemplos deste roteiro vamos usar o seguinte exemplo: em um sistema é necessário realizar um controle dos usuários que estão usando os computadores dos laboratórios de computação, anotando as seguintes informações sobre ele: login (que é único) e data do último acesso.

Podemos fazer a associação entre o login e a data usando lista:

a) Uma lista para cada informação de usuário, sendo que a associação entre o login e a data é feito pelo índice (numérico).

```
logins = ['minnie','mickey','pateta']
datas  =  ['20/06/2021','21/06/2021','18/06/2021']

#Para acessar o data associada ao login
indice = logins.index('mickey')
print(datas[indice])

# Ou assim, de forma mais direta
print(datas[logins.index('mickey')])

# Se quiser adicionar mais um login e sua data
logins.append('donald')
datas.append('23/06/2021')
```

b) Uma lista de sublistas, sendo que cada sublista guarda informações de um usuário.

```
usuarios = [['minnie', '20/06/2021'],
            ['mickey','21/06/2021'],
            ['pateta','18/06/2021']]

# Para acessar o data associada ao login
for indice in range(len(usuarios)):
    if usuarios[indice][0]=='mickey':
        print(usuarios[indice][1])

# Se quiser adicionar mais um login e sua data
usuarios.append(['donald', '23/06/2021'])
```

E se o usuário tentar inserir dois usuários com o mesmo login? Para evitar isso a validação terá que ser implementada no código.

Dicionário é uma estrutura que faz um mapeamento*(relação, associação) entre pares de valores, que são uma chave e os dados relacionados a ela (conteúdo ou significado).
* Mapeamento é uma generalização da ideia de acessar dados por índices, exceto que em um mapeamento os índices (ou chaves) podem ser de outro tipo além de inteiro.

```
#Criando um dicionário e testando
usuarios={'minnie':'20/06/2021', 'mickey':'21/06/2021', 'clarabela':'24/06/2021'}
print(usuarios['minnie'])

#Inserindo uma nova associação e testando
usuarios['donald'] = '23/06/2021'
print(usuarios['donald'])
```

E se você fizer da seguinte forma para criar um dicionário?

```
#Código inicia aqui, criando uma associação "do nada"
usuarios['donald'] = '23/06/2021'
print(usuarios['donald'])

# Dá "ruim" sim!!!
```

A maneira correta de criar um dicionário vazio e inserir individualmente os pares **chave:dados** é a seguinte:

```
#Código inicia aqui, criando um dicionário vazio
usuarios = {}
usuarios['donald'] = '23/06/2021'
print(usuarios['donald'])

# Assim dá certo!!!
```

Dicionário é um objeto **mutável.** Por exemplo, é possível alterar uma associação existente.

```
#Código inicia aqui! Criando um dicionário e testando
usuarios={'minnie': '20/06/2021', 'mickey': '21/06/2021', 'clarabela': '24/06/2021'}
print(usuarios['minnie'])

#Inserindo uma nova associação e testando
usuarios['donald'] = '23/06/2021'
print(usuarios['donald'])

#Alterando uma associação existente e testando
print('Antes', usuarios['mickey'])
usuarios['mickey'] = '23/06/2022'
print('Depois:', usuarios['mickey'])
```

Atenção: percebam que a mesma instrução é utilizada para inserir uma associação ou alterar uma existente. A lógica é, se a associação não existir ela é criada, caso contrário, o seu conteúdo é alterado.

Acrescente no código as instruções a seguir para imprimir em **ordem alfabética**:

```
#Imprimindo em ordem crescente – ordenado pela chave
print('\nUsuários - ordenado pela chave:')
for chave in sorted(usuarios):
        print(chave, usuarios[chave])

#Imprimindo em ordem decrescente – ordenado pela chave
print('\nUsuários - em ordem decrescente de chave:')
for chave in sorted(usuarios, reverse = True):
    print(chave, usuarios[chave])

#imprimindo em ordem crescente – ordenado pelo valor/conteúdo
print('\nUsuários - ordenado pelo conteúdo:')
for i in sorted(usuarios, key = usuarios.get):
    print(i, usuarios[i])

#imprimindo valores em ordem decrescente – ordenado pelo valor/conteúdo
print('\nUsuários – em ordem decrescente de conteúdo:')
for i in sorted(usuarios, key = usuarios.get, reverse = True):
    print(i, usuarios[i])
```

Acrescente no código as instruções a seguir para **apagar/deletar** o usuário:

```
#Apagando o usuário e testando
del usuarios['mickey']
print(usuarios)
```

Acrescente no código as instruções a seguir para **verificar o tamanho** do dicionário:

```
#Imprimindo o tamanho
print('Tamanho:', len(usuarios))
```

Acrescente no código as instruções a seguir para imprimir as chaves - **keys** - e os dados (valores, conteúdos) - **values** - do dicionário:

```
# Imprimindo chaves (keys) e valores (values)
print('Chaves:', usuarios.keys())
print('Valores:',usuarios.values())
```

Vamos brincar um pouco com o comando **for** no código a seguir:

```
#Código inicia aqui! Brincando com o método get()
usuarios={'minnie': '20/06/2021', 'mickey': '21/06/2021', 'clarabela': '24/06/2021',
'donald':'23/06/2021'}
print(usuarios)

print('\nLogins (keys):')
for chave in usuarios.keys():
    print(chave)

print('\nÚltimos acessos (values):')
for item in usuarios.values():
    print(item)

print('\nItens (items):')
for (chave,valor) in usuarios.items():
    print ('Último acesso de', chave,'foi em',valor)
```

Pode-se usar o **in** e o **not in** para verificar a existência (ou não) de uma chave em um dicionário. Acrescente no código as instruções a seguir e veja como fazer.

```
#verificando a existência de chaves
if 'minnie' in usuarios:
    print('O último acesso de minnie foi:', usuarios['minnie'])
else:
    print('Não encontrado!!')

#verificando a inexistência de chaves
if 'pateta' not in usuarios:
    print('Login pateta não existe!')
else:
    print('O último acesso de pateta foi:', usuarios['pateta'])
```

Execute o código a seguir:

```
#Código inicia aqui! Criando um dicionário e testando
usuarios={'minnie': '20/06/2021', 'mickey': '21/06/2021', 'clarabela': '24/06/2021',
'donald':'23/06/2021'}
print(usuarios)
print(usuarios['minnie'])
print(usuarios['pateta'])

#Deu erro (KeyError) no login 'pateta', não foi?
```

O método **get()** recebe um valor e vai pesquisá-lo entre as chaves que existem dentro do dicionário, caso encontre, retornará os dados relativos à chave encontrada. Em caso contrário, não dá erro. Execute o código a seguir e analise o funcionamento do get().

```
#Código inicia aqui! Usando o método get()
usuarios={'minnie': '20/06/2021', 'mickey': '21/06/2021', 'clarabela': '24/06/2021', 'donald':'23/06/2021'}
print(usuarios)
print(usuarios.get('minnie'))
print(usuarios.get('minnie','Não existe'))
print(usuarios.get('pateta'))
print(usuarios.get('pateta','Não existe'))
```

- GET(chave, msg) ➔ Caso a chave exista imprime o valor associado a ela. Caso a chave não exista, retorna a mensagem (**msg**) digitada (quando forem passados dois argumentos) ou não apresenta retorno (**none)** (quando for passada apenas a **chave** como argumento).

Fugindo do contexto para brincar com o método get() . Execute o código a seguir e analise o resultado da execução.

```
#Código inicia aqui! Brincando com o método get()
contaLetras = {}
for letra in 'orangotango':
    contaLetras[letra] = contaLetras.get(letra,0) + 1
    print('No for. letra:', letra, 'dicionário:', contaLetras)

print('\n\nDicionário completo:', contaLetras)
```

****E se eu quiser juntar dois dicionários ?**

```
dic1={'minnie': '20/06/2021', 'mickey': '21/06/2021'}
dic2 = {'clarabela': '24/06/2021', 'donald':'23/06/2021'}

print('**antes***')
print('dic1 -', dic1)
print('dic2 -', dic2)

dic1.update(dic2)

print('**depois***')
print('dic1 -', dic1)
print('dic2 -', dic2)
```

****E se existirem vários dados (conteúdos) associados a uma única chave?**

Para exemplificar essa situação, vamos expandir o nosso contexto: em um sistema é necessário realizar um controle dos usuários que estão usando os computadores dos laboratórios de computação, anotando as seguintes informações sobre ele: login (que é único), nome, data do último acesso e a máquina em que ele se conectou no último acesso.

Exemplos:

Login: Mickey
Nome: Mickey Mouse
Último acesso: 22/06/2021
Máquina: larc01

Login: minnie
Nome: Minerva Mouse
Último acesso: 21/06/2021
Máquina: labinIV15

Execute o código a seguir e analise a sua saída. Os comentários no código ajudarão na análise e compreensão do funcionamento.

```
#Código inicia aqui! Criando um dicionário e testando
usuarios = {'minnie':['Minerva Mouse', '21/06/2021','labinIV15'],
            'mickey':['Mickey Mouse','22/06/2021', 'larc01']}
print('Usuários - incio:', usuarios)

#Inserindo um usuário
usuarios['donald']=['Donald Duck', '22/06/2021', 'lbdes03']
#Imprimindo o dicionário
print('Usuários após inserir Donald:')
for chave, valor in usuarios.items():
    print('\tLogin: ', chave)
    print('\tDados: ', valor)

#Alterando a data do último acesso e máquina do Donald
usuarios['donald'][1]='23/06/2021'
usuarios['donald'][2]='labinV15'

#Excluindo a máquina de Minnie
del usuarios['minnie'][1]

#Imprimindo o dicionário - ***diferente da impressão anterior
print('Usuários após inserir Donald:')
for chave in usuarios.keys():
    print('\tLogin: ', chave)
    print('\tDados: ', usuarios.get(chave))

#funções que retornam listas a partir dos dicionários
lista1 = list(usuarios.values())
print(lista1)
lista2 = list(usuarios.keys())
print(lista2)
lista3 = list(usuarios.items())
print(lista3) #neste, cada item da lista é uma tupla
print(lista3[0][1])
```

**Para preparar esse roteiro, alguns dos exemplos foram tirados do material do professor Fabiano.

LABORATÓRIO DE PROGRAMAÇÃO

AULA 13 – MANIPULAÇÃO DE ARQUIVOS TEXTO EM PYTHON

Antes de executar esse exemplo, crie um arquivo chamado “notas.txt” com o seguinte conteúdo:

```
nome,n1,n2
Ana,10,10
Beto,8,7
Carla,7,7
Daniel,5,7
Eva,6,8
```

Agora, execute o código a seguir, uma parte de cada vez, leia os comentários e tente entender o que acontece:

```
arq = open("notas.txt", "r")

#Lê uma linha por vez até encontrar o final do arquivo
conteudo = '--'
while conteudo!="":
    conteudo = arq.readline() #lê uma linha por vez
    print(conteudo)

#Lê e guarda cada linha em uma lista
conteudo = arq.readlines()
print(conteudo)#Veja que o \n  fica na lista

###Nesse momento a posição corrente é o final do arquivo###

###E se eu quiser ler novamente??###

#Posiciona no início do arquivo
arq.seek(0)

#Lê tudo em uma String única
conteudo = arq.read()
print(conteudo)

#divide o conteúdo lido em uma lista
lista = conteudo.split("\n")
print(lista)

#imprime cada elemento da lista em uma linha
for linha in lista:
    print(linha)


###Preciso ler só a partir da segunda linha, para desconsiderar o cabeçalho###

#posiciona no início da segunda linha
tamPri = len(arq.readline())
arq.seek(tamPri+1,0) #argumento 1: deslocamento em bytes – argumento 2: desloca a partir da posição 0
conteudo = arq.read()

#imprime cada elemento da lista em uma linha, a partir da segunda linha
conteudo = arq.read()
lista = conteudo.split("\n")
print(lista)

for linha in lista:
    print(linha)
```

arq.seek(0, 2)
Posiciona no final do arquivo

```
###E se eu quiser separar o nome das notas para calcular as médias?

#Separa os dados de cada elemento (linha/String) da lista
for linha in lista:
    dados = linha.split(",")
    nome = dados[0]
    n1 = float(dados[1])
    n2 = float(dados[2])
    print("Média de", nome, "=", (n1+n2)/2)

arq.close()

###E se eu quiser guardar as médias em um novo arquivo de texto###
arq2 = open("medias.txt","w")
arq2.write("nome,média\n")
for linha in lista:
    dados = linha.split(",")
    nome = dados[0]
    n1 = float(dados[1])
    n2 = float(dados[2])
    arq2.write(nome+','+str((n1+n2)/2)+"\n")

arq2.close()
```

```
arquivo = open("alunos.txt", "r")
dadosAlunos = arquivo.read().strip().splitlines()
arquivo.close()

alunos = []

dadosAlunos.pop(0)
for dados in dadosAlunos:
    nome, g1, g2 = dados.split(",")
    g1, g2 = float(g1), float(g2)
    media = (g1 + (g2 * 2))/3
    alunos.append({
        "nome": nome,
        "g1": g1,
        "g2": g2,
        "media": round(media, 2) #round(media, numero de casas) -> arredonda a média para
numero de casas decimais
})

arquivo = open("medias.txt", "w") # "w" ele cria um a arquivo se não existir;
arquivo.write("nome, media\n")
for aluno in alunos:
    arquivo.write(f"{aluno['nome']}, {aluno['media']}\n") # Eu estou concatenando strings e
variáveis
arquivo.close()
```

nome, media

fulano, 9.33

ciclano, 8.0

Beltrano, 4.67

```
"""
Faça o programa abaixo funcionar, para isso, implemente o módulo
com as funções correspondentes.
Obs.: os parâmetros das funções não devem ser alterados!
"""


from funcoes import *

# Produto / Quantidade
lista_de_compras = {"arroz": 2, "refrigerante": 10}

while True:
    print("-" * 30 + "MENU" + 30 * "-")
    menu = input(
        "[1] - Adicionar item à lista\n"
        "[2] - Exibir a lista completa\n"
        "[3] - Atualizar quantidade de um item\n"
        "[4] - Remover item à lista\n"
        "[0] - Sair\n"
        "-> "
    )

    if menu == "1":
        # 0,25
        nome_produto = input("Nome do produto: ").lower()
        quantidade = int(input("Quantidade: "))

        adicionar_item(lista_de_compras, nome_produto, quantidade)
    elif menu == "2":
        # 0,25
        exibir_lista_completa(lista_de_compras)
        """
        Exemplo de saída:
        1 - Item: Arroz | Qtde: 2
        2 - Item: Refrigerante | Qtde: 10
        ...
        """
    elif menu == "3":
        # 0,25
        exibir_lista_completa(lista_de_compras)
        nome_produto = input("Nome do produto: ").lower()
        quantidade = int(input("Quantidade: "))

        atualizar_quantidade_item(lista_de_compras, nome_produto, quantidade)
    elif menu == "4":
        # 0,25
        exibir_lista_completa(lista_de_compras)
        nome_produto = input("Nome do produto: ").lower()

        remover_item(lista_de_compras, nome_produto)
    elif menu == "0":
```

```
        break
    else:
        print("Opção inválida!")
```

```
def adicionar_item(lista_de_compras, nome_produto, quantidade):
    lista_de_compras[nome_produto] = quantidade


def exibir_lista_completa(lista_de_compras: dict):
    cont = 1
    for nome_produto, quantidade in lista_de_compras.items():
        print(f"{cont} - Item: {nome_produto} | Qtde: {quantidade}")
        cont += 1

def atualizar_quantidade_item(lista_de_compras, nome_produto, quantidade):
    lista_de_compras[nome_produto] = quantidade


def remover_item(lista_de_compras: dict, nome_produto):
    lista_de_compras.pop(nome_produto)
```

```
def menores_palavaras(frase: str):
    palavras = frase.split(" ")
    tam_menor = len(palavras[0])
    menores = []

    for palavra in palavras:
        if len(palavra) < tam_menor:
            tam_menor = len(palavra)

    for palavra in palavras:
        if len(palavra) == tam_menor:
            menores.append(palavra)

    return menores
```

```
from funcoes import *

frase = "Uma frase a b qualquer"
print(menores_palavaras(frase))
```