Computação I - Python
Aula 8 - Teórica: Estrutura de Repetição : for

João C. P. da Silva

Carla A. D. M. Delgado

Ana Luisa Duboc

Dept. Ciência da Computação - UFRJ

Estrutura de Repetição *while*

Estrutura que permite a repetição de um conjunto de comandos. Até o momento vimos o *while*:

```
while <condição>:
    <sequência de comandos>
```

- Com *while* podemos implementar qualquer algoritmo que envolva repetição.
- **DICA**: o *while* é mais recomendado quando não se sabe ao certo quantas vez a repetição será feita, pois a condição é um teste booleano qualquer e não necessariamente uma contagem.

Estrutura de Repetição *while*

Lembre: Faça uma função que gere números aleatórios entre 1 e 10 e calcule a soma destes números até que seja gerado o número 5.

```
1  from random import randint
2
3  def somaAleatoria():
4
5  """Paramentro de entrada: nao tem
6  Valor de retorno: int"""
7
8    soma = 0
9    numero = randint(1,10)
10   while numero != 5:
11     soma = soma + numero
12     numero = randint(1,10)
13   return soma
```

O número de repetições dos comandos associados ao laço *while* depende de quando sair o número 5. Podem ser 2 vezes ou 1000 vezes!

Estrutura de Repetição *while*

Faça uma função *somaPares* que recebe uma tupla de números inteiros e calcula a soma de todos os números pares que ocorrem nesta tupla.
Por exemplo, a chamada $somaPares((3, 1, 2, 4, 6, 7, 2))$ deve retornar 14 e $somaPares((1, 3, 5, 7))$ deve retornar 0.

Estrutura de Repetição *while*

Faça uma função *somaPares* que recebe uma tupla de números inteiros e calcula a soma de todos os números pares que ocorrem nesta tupla.
Por exemplo, a chamada $somaPares((3, 1, 2, 4, 6, 7, 2))$ deve retornar 14 e $somaPares((1, 3, 5, 7))$ deve retornar 0.

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    indice = 0
8    while indice < len(tupla):
9      if tupla[indice] % 2 == 0:
10       soma = soma + tupla[indice]
11     indice = indice + 1
12   return soma
```

Estrutura de Repetição *while*

Faça uma função *somaPares* que recebe uma tupla de números inteiros e calcula a soma de todos os números pares que ocorrem nesta tupla.
Por exemplo, a chamada $somaPares((3, 1, 2, 4, 6, 7, 2))$ deve retornar 14 e $somaPares((1, 3, 5, 7))$ deve retornar 0.

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    indice = 0
8    while indice < len(tupla):
9      if tupla[indice] % 2 == 0:
10       soma = soma + tupla[indice]
11     indice = indice + 1
12   return soma
```

Estamos usando a varíavel *indice* para percorrer a *tupla*.

É possível acessar os elementos na tupla sem precisarmos da variável *indice*!

Estrutura de Repetição *for*

Faça uma função *somaPares* que recebe uma tupla de números inteiros e calcula a soma de todos os números pares que ocorrem nesta tupla.
Por exemplo, a chamada $somaPares((3, 1, 2, 4, 6, 7, 2))$ deve retornar 14 e $somaPares((1, 3, 5, 7))$ deve retornar 0.

Com o comando for, podemos pegar um a um os elementos que formam a tupla dada como entrada:

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0:

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - *soma* = 0 + 10 = 10

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - *soma* = 0 + 10 = 10
  - if 21 % 2 == 0:

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - *soma* = 0 + 10 = 10
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0:

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0: (True)

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0: (True)
    - $soma = 10 + 32 = 42$

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0: (True)
    - $soma = 10 + 32 = 42$
  - if 43 % 2 == 0:

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0: (True)
    - $soma = 10 + 32 = 42$
  - if 43 % 2 == 0: (False)

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

- somaPares((10,21,32,43))
- soma = 0
- for elemento in (10,21,32,43):
  - if 10 % 2 == 0: (True)
    - $soma = 0 + 10 = 10$
  - if 21 % 2 == 0: (False)
  - if 32 % 2 == 0: (True)
    - $soma = 10 + 32 = 42$
  - if 43 % 2 == 0: (False)
- return 42

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

Poderíamos ter usado uma lista ao invés de uma tupla ?

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

Poderíamos ter usado uma lista ao invés de uma tupla ? SIM !

Estrutura de Repetição *for*

```
1  def somaPares(tupla):
2
3  """Paramentro de entrada: tupla
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

Poderíamos ter usado uma lista ao invés de uma tupla ? SIM !

```
1  def somaParesLista(lista):
2
3  """Paramentro de entrada: list
4  Valor de retorno: int"""
5
6    soma = 0
7    for elemento in tupla:
8      if elemento % 2 == 0:
9        soma = soma + elemento
10   return soma
```

somaParesLista([10,21,32,43]) retorna 42 também !

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.
Como seria essa função com while?

Estrutura de Repetição *while*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5. Como seria essa função com while?

```
1  from random import randint
2
3  def soma10():
4
5  """ Funcao que soma 10 numeros gerados aleatoriamente no intervalo de 1 a 5
6  Parametro de entrada: nao tem
7  Valor de retorno: int"""
8
9    contador = 0
10   soma = 0
11   while contador < 10:
12     numero = randint(1,5)
13     soma = soma + numero
14     contador = contador + 1
15   return soma
```

O número de repetições será 10 em qualquer execução da função, independente dos números aleatórios gerados.

Estrutura de Repetição *for*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.
Como seria essa função com for ?

Estrutura de Repetição *for*

Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5. Como seria essa função com for ? Na função, a variável *contador* vai assumir os valores 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9.

```
1  from random import randint
2
3  def soma10usandofor():
4
5  """ Funcao que soma 10 numeros gerados aleatoriamente no intervalo de 1 a 5 usando o
       comando for
6  Parametro de entrada: nao tem
7  Valor de retorno: int"""
8
9    soma = 0
10   for contador in range(10):
11     numero = randint(1,5)
12     soma = soma + numero
13   return soma
```

```
for var in range(n):          OU      for var in [0,..., n-1]:
    <comandos>                                <comandos>
```

Estrutura de Repetição *for*

- A função range(...) pode ter 1, 2 ou 3 argumentos:
  - range(numero): faz com que a variável do **for** assuma valores de 0 a numero-1

    `for x in range(10):` → x recebe 0,1,2,...,9
  - range(inf,sup): faz com que a variável do **for** assuma valores de inf a sup-1

    `for x in range(3,8):` → x recebe 3,4,5,6,7
  - range(inf, sup, inc): faz com que a variável do **for** assuma valores de inf a sup-1 com incremento de inc

    `for x in range(3,8,2):` → x recebe 3,5,7

Estrutura de Repetição *for*

Faça uma função que determina a soma de todos os números pares desde 100 até 200. (Usando **for** ao invés de **while**)

Estrutura de Repetição *for*

Faça uma função que determina a soma de todos os números pares desde 100 até 200. (Usando **for** ao invés de **while**)

```
1  def somaPares():
2
3  """Funcao que soma todos os numeros pares de 100 ate 200
4  Parametro de entrada: nao tem
5  Valor de retorno: int"""
6
7    soma = 0
8    for par in range(100,202,2):
9      soma = soma + par
10   return soma
```

OU

```
1  def somaPares():
2
3  """Funcao que soma todos os numeros pares de 100 ate 200
4  Parametro de entrada: nao tem
5  Valor de retorno: int"""
6
7    soma = 0
8    lista = range(100,202,2)
9    for par in lista:
10     soma = soma + par
11   return soma
```

Estrutura de Repetição

IMPORTANTE: diferença de uso entre *while* e *for*:

- **while**: decisão sobre repetir ou não baseia-se em teste booleano. Risco de loop infinito. :-(
- **for**: Contagem automática do número de repetições.

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'e')

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'e')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'e')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'e')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- vogaisPalavra('testando')
- vogais = ['a','e','i','o','u','A','E','I','O','U'], resposta = [ ]
- for letra in 'testando':
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'e' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'e')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 't' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e','a'],'o')

Estrutura de Repetição *for*

Faça uma função que dada uma palavra retorna uma lista formada por todas as vogais que aparecem na palavra.

```
1  def vogaisPalavra(palavra):
2
3  """Funcao que retorna todas as vogais que aparecem em uma palavra
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    resposta = [ ]
9    for letra in palavra:
10     if letra in vogais:
11       list.append(resposta,letra)
12   return resposta
```

- for letra in 'testando':
  - if 'a' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e'],'a')
  - if 'n' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['e','a'],'o')
- return ['e','a','o']

Estrutura de Repetição *for*

Faça uma função que dada uma string retorna uma lista formada por todas as palavras que começam com vogais.
Dica: use a função str.split para separar as palavras na frase.

Estrutura de Repetição *for*

Faça uma função que dada uma string retorna uma lista formada por todas as palavras que começam com vogais.
Dica: use a função str.split para separar as palavras na frase.

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que come am com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que come am com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'],

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que come am com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta ,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que come am com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta ,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['Amanhã'],'outro')

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['Amanhã'],'outro')
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']:

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['Amanhã'],'outro')
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)

Estrutura de Repetição *for*

```
1  def palavrasComecandoComVogal(frase):
2
3  """Funcao que retorna todas as palavras que comeam com vogais
4  Parametro de entrada: str
5  Valor de retorno: list"""
6
7    vogais = ['a','e','i','o','u','A','E','I','O','U']
8    lista = str.split(frase)
9    resposta = [ ]
10   for palavra in lista:
11     if palavra[0] in vogais:
12       list.append(resposta,palavra)
13   return resposta
```

- palavrasComecandoComVogal('Amanhã será outro dia.')
- vogais = ['a','e','i','o','u','A','E','I','O','U'],
  lista = ['Amanhã','será','outro','dia.'], resposta = [ ]
- for palavra in ['Amanhã','será','outro','dia.']:
  - if 'A' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append([ ],'Amanhã')
  - if 's' in ['a','e','i','o','u','A','E','I','O','U']: (False)
  - if 'o' in ['a','e','i','o','u','A','E','I','O','U']: (True)
    - list.append(['Amanhã'],'outro')
  - if 'd' in ['a','e','i','o','u','A','E','I','O','U']: (False)
- return ['Amanhã','outro']

Estrutura de Repetição *for*

Resolva usando for:

1. Dada uma lista de nomes e um número inteiro n, faça uma função que conte quantos nomes de tamanho maior que n aparecem nesta lista.

2. Faça uma função que calcule o valor de N!, onde N é passado como parâmetro. (Sem usar o factorial do módulo *math*).

3. Faça uma função que calcule e retorne o valor de

$$S = \frac{1}{1} + \frac{3}{2} + \frac{5}{3} + \frac{7}{4} + \cdots + \frac{99}{50}$$

4. Faça uma função que calcule e retorne o valor de

$$S = \frac{1}{1} - \frac{2}{4} + \frac{3}{9} - \frac{4}{16} + \cdots - \frac{10}{100}$$

Atenção ao tipo de dado !!!

Autores

- João C. P. da Silva ▸ Lattes
- Carla Delgado ▸ Lattes
- Ana Luisa Duboc ▸ Lattes

Colaboradores

- Anamaria Martins Moreira ▸ Lattes
- Fabio Mascarenhas ▸ Lattes
- Leonardo de Oliveira Carvalho ▸ Lattes
- Charles Figueiredo de Barros ▸ Lattes
- Fabrício Firmino de Faria ▸ Lattes

Computação I - Python
Aula 8 - Teórica: Estrutura de Repetição : for

João C. P. da Silva

Carla A. D. M. Delgado

Ana Luisa Duboc

Dept. Ciência da Computação - UFRJ