# COMPUTAÇÃO 1 – PYTHON
# AULA 8 TEÓRICA

SILVIA BENZA

SILVIABENZA@COS.UFRJ.BR

# ESTRUTURAS DE REPETIÇÃO

## VAMOS A APRENDER COMO TRABALHAR COM LOOPS! PARTE 2


THE ROAD SO FAR

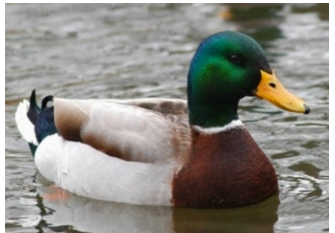
# WHILE

**Permite que o programador especifique que o programa deve repetir um conjunto de comandos enquanto uma dada condição for verdadeira.**

```
while condição:
    conjunto de comandos
```

***Com while podemos implementar qualquer algoritmo que envolva repetição.***

- ***DICA****: o while é mais recomendado quando não se sabe ao certo quantas vezes a repetição será feita, pois a condição é um teste booleano qualquer e não necessariamente uma contagem*



# WHILE

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

```
from random import randint

def somaAleatoria():
    soma = 0
    numero = randint(1,10)
    while numero != 5:
        soma = soma + numero
        numero = randint(1,10)
    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**

# WHILE

**Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.**

```
#função que soma 10 números aleatórios entre 1 e 5
# none -> int
from random import randint
def soma10():
    contador = 0
    soma = 0
    while contador < 10:
        numero = randint(1,5)
        soma = soma + numero
        contador = contador + 1
    return soma
```

***O número de repetições será 10 em qualquer execução do programa, independente dos números aleatórios gerados.***


NOW
THE CW

# FOR

**Utilizado para iterar sobre os elementos de uma lista**

**Forma geral:**

```
for var in [0,..., n-1]:
    comandos
```

- Os comandos são repetidos para cada valor de lista
- Durante a repetição, var possui o valor corrente da lista

**Lembram da função RANGE?**

- Utilizamos a função range para construir a lista de iteração (caso ainda não a tenhamos!)

```
for var in range(n):
    comandos
```

# (THEN)
# FUNÇÃO RANGE

**A função range(...) pode ter 1, 2 ou 3 argumentos:**
**range(numero):** retorna uma lista contendo uma sequência de valores de 0 a numero-1

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

**range(inf,sup):** retorna uma lista contendo uma sequência de valores de inf a sup-1

```
>>> range(3, 8)
[3, 4, 5, 6, 7]
```

**range(inf, sup, inc):** retorna uma lista contendo uma sequência de valores de inf a sup-1 com incremento de inc

```
>>> range(3, 8, 2)
[3, 5, 7]
```

# (NOW) UTILIZANDO O FOR PARA A QUESTÃO ANTERIOR!

**Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.**

**Como seria essa função com for?**

# UTILIZANDO O FOR PARA A QUESTÃO ANTERIOR!

**Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.**

**Como seria essa função com for?**

```
from random import randint
def soma10():
    soma = 0
    for contador in range(10):
        numero = randint(1,5)
        soma = soma + numero
    return soma
```

# UTILIZANDO O FOR PARA A QUESTÃO ANTERIOR!

**Faça uma função que some 10 números gerados aleatoriamente no intervalo de 1 a 5.**

**Como seria essa função com for?**

```
from random import randint
def soma10():
    soma = 0
    for contador in range(10):
        numero = randint(1,5)
        soma = soma + numero
    return soma
```

Contador vai assumir os valores 0,1,2,3,4,5,6,7,8,9

# WHILE VS FOR

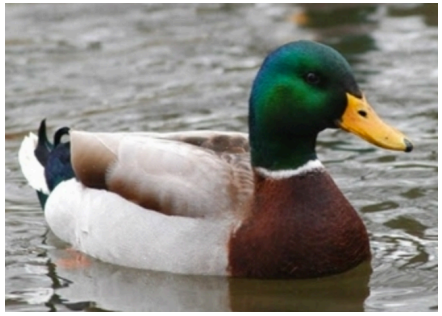
```
from random import randint
def soma10():
    contador = 0
    soma = 0
    while contador < 10:
        numero = randint(1,5)
        soma = soma + numero
        contador = contador + 1
    return soma
```

```
from random import randint
def soma10():
    soma = 0
    for contador in range(10):
        numero = randint(1,5)
        soma = soma + numero
    return soma
```

*While: decisão sobre repetir ou não, baseia-se em teste booleano.*

*Risco de loop infinito.*

*For: Contagem automática do número de repetições.*



# EXERICIO

**Faça um programa que determina a soma de todos os números pares desde 100 até 200. (Usando For ao invés de While)**

# EXERICIO

**Faça um programa que determina a soma de todos os números pares desde 100 até 200. (Usando For ao invés de While)**

```
def somaPares():

    Soma = 0
    for Par in range(100,202,2) :
        Soma = Soma + Par
    return Soma
```

# ESTRUTURAS DE REPETIÇÃO

## VAMOS A APRENDER COMO TRABALHAR COM LOOPS! PARTE 3

# ES AS
# DI ÃO

**BREAK THE LOOP!**

# BREAK E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

```
# Tente descobrir o que faz esta função
# int → int
def soma(numero):
    soma = 0
    contador = 0
    while contador < numero:
        if contador == 5:
            break
        soma = soma + contador
        contador = contador + 1
    return soma
```

**Qual a saída desta função se a chamada for soma(10)?**

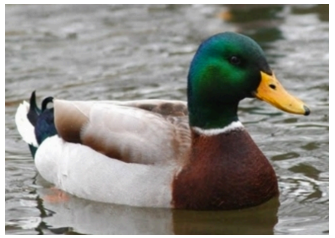
# BREAK E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

```
# Tente descobrir o que faz esta função
# int → int
def soma(numero):
    soma = 0
    contador = 0
    while contador < numero:
        if contador == 5:
            break
        soma = soma + contador
        contador = contador + 1
    return soma
```

**Qual a saída desta função se a chamada for soma(10)?**

**O comando break interrompe o “loop” quando contador == 5**



# CONTINUE E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

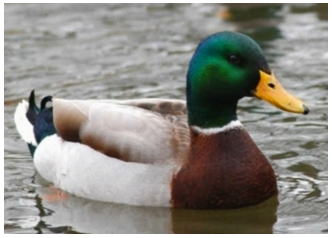
```
# Tente descobrir o que faz esta função
# int → int
def soma1(numero):
    soma = 0
    contador = 0
    while contador < numero:
        contador = contador + 1
        if contador == 5:
            continue
        soma = soma + contador
    return soma
```

**Qual a saída desta função se a chamada for soma1(10)?**

# CONTINUE E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

```
# Tente descobrir o que faz esta função
# int → int
def soma1(numero):
    soma = 0
    contador = 0
    while contador < numero:
        contador = contador + 1
        if contador == 5:
            continue
        soma = soma + contador
    return soma
```



**Qual a saída desta função se a chamada for soma1(10)?**

**O comando continue pula para a próxima execução do "loop" quando contador == 5, ou seja, não acumula a soma quando contador == 5 !**

# CONTINUE E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

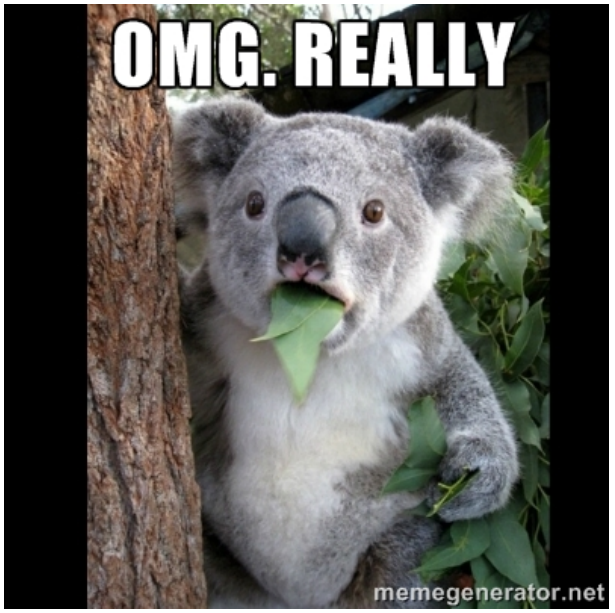
```
# Tente descobrir o que faz esta função
# int → int
def soma2(numero):
    soma = 0
    contador = 0
    while contador < numero:
        if contador == 5:
            continue
        soma = soma + contador
        contador = contador + 1
    return soma
```

**Qual a saída desta função se a chamada for soma2(10)?**

# CONTINUE E WHILE

**Permite que o programador especifique uma alteração na estrutura de repetição**

```
# Tente descobrir o que faz esta função
# int → int
def soma2(numero):
    soma = 0
    contador = 0
    while contador < numero:
        if contador == 5:
            continue
        soma = soma + contador
        contador = contador + 1
    return soma
```




**Qual a saída desta função se a chamada for soma2(10)?**

**Nenhuma!! Fica num loop infinito!!!**

# EXERCÍCIO

**Considere a função que gera números aleatórios entre 1 e 10 e calcula a soma destes números até que seja gerado o número 5.**

```
from random import randint
# função que soma números gerados aleatóriamente

# sem parâmetro → int
def somaAleatoria():
    soma = 0
    numero = randint(1,10)
    while numero != 5:
        soma = soma + numero
        numero = randint(1,10)
    return soma
```

# EXERCÍCIO

**Considere a função que gera números aleatórios entre 1 e 10 e calcula a soma destes números até que seja gerado o número 5.**

**Complete a função com essa nova condição:**

```
from random import randint
# função que soma números gerados aleatóriamente

# sem parâmetro → int
def somaAleatoria():
    soma = 0
    while True: # True indica um loop infinito

        COMPLETE A FUNÇÂO

    return soma
```

Nova condição!

# EXERCÍCIO

**Considere a função que gera números aleatórios entre 1 e 10 e calcula a soma destes números até que seja gerado o número 5.**

```
from random import randint
# função que soma números gerados aleatóriamente

# sem parâmetro → int
def somaAleatoria():
    soma = 0
    while True: # True indica um loop infinito
        numero = randint(1,10)
        if numero ==5:
            break # Interrompe o loop infinito
        soma = soma + numero
    return soma
```

# SE A GENTE VIU FOR, PORQUE SÓ NO WHILE?

# UÉ, CLARO QUE PODEMOS USAR TAMBÉM NO FOR!

**Olha aqui, qual a saída?**

```
# Tente descobrir o que faz esta função
# sem parâmetro → int
def Exemplo1():
    lista = [ ]
    for x in range(1, 11):
        if x ==5:
            break
        lista += [x]
    return lista
```

# BREAK E FOR

```
# Tente descobrir o que faz esta função
# sem parâmetro → int
def Exemplo1():
    lista = [ ]
    for x in range(1, 11):
        if x ==5:
            break
        lista += [x]
    return lista
```

**[1,2,3,4]**

# CONTINUE E FOR

**E qual a saída aqui?**

```
# Tente descobrir o que faz esta função
# sem parâmetro → int
def Exemplo2():
    lista = [ ]
    for x in range(1, 11)
        if x ==5:
            continue
        lista += [x]
    return lista
```

# CONTINUE E FOR

**E qual a saída aqui?**

```
# Tente descobrir o que faz esta função
# sem parâmetro → int
def Exemplo2():
    lista = [ ]
    for x in range(1, 11)
        if x ==5:
            continue
        lista += [x]
    return lista
```

**[1,2,3,4,6,7,8,9,10]**

# EXERCÍCIO

**Diga o que é retornado pela função abaixo para os seguintes valores de entrada: 501, 745, 384, 2, 7 e 1.**

**O que faz a função?**

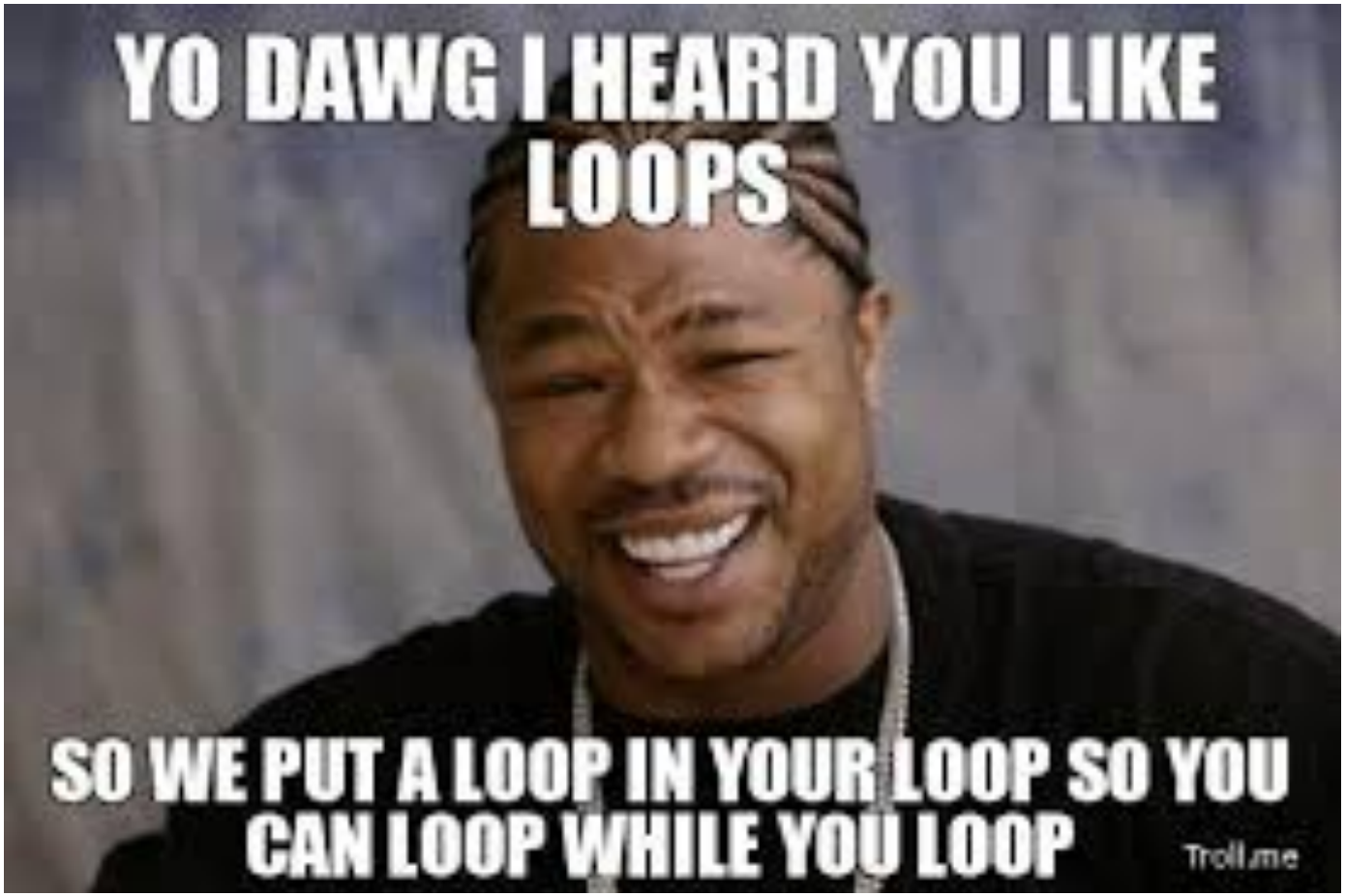
```
# Tente descobrir o que faz esta função
# int → list
def contagemcedulas(valor):
    cedulas = 0
    atual = 50
    apagar = valor
    contagem = []
    while True:
        if atual <= apagar:
            apagar -= atual
            cedulas += 1
        else:
            contagem += [(cedulas,atual)]
            if apagar <= 1:
                break
            if atual == 50:
                atual = 20
            elif atual == 20:
                atual = 10
            elif atual == 10:
                atual = 5
            elif atual == 5:
                atual = 2
            cedulas = 0
    return contagem
```

# EXERCÍCIO

**Modifique a função para considerar cédulas de 100.**

**Modifique a função para retornar uma mensagem de erro caso o valor não possa ser completamente pago apenas por cédulas.**

```
# Tente descobrir o que faz esta função
# int → list
def contagemcedulas(valor):
    cedulas = 0
    atual = 50
    apagar = valor
    contagem = []
    while True:
        if atual <= apagar:
            apagar -= atual
            cedulas += 1
        else:
            contagem += [(cedulas,atual)]
            if apagar <= 1:
                break
            if atual == 50:
                atual = 20
            elif atual == 20:
                atual = 10
            elif atual == 10:
                atual = 5
            elif atual == 5:
                atual = 2
            cedulas = 0
    return contagem
```


YO DAWG I HEARD YOU LIKE LOOPS
SO WE PUT A LOOP IN YOUR LOOP SO YOU CAN LOOP WHILE YOU LOOP
Troll.me

# EXERCÍCIO

**Podemos combinar mais de uma estrutura de repetição de forma a obter resultados interessantes.**
**Exemplo: Gerar as tabuadas de multiplicação de 1 a 10.**

# EXERCÍCIO

**Podemos combinar mais de uma estrutura de repetição de forma a obter resultados interessantes.**
**Exemplo: Gerar as tabuadas de multiplicação de 1 a 10.**

```
# função tabuadas que gera as tabuadas
# de multiplicação de 1 a 10
# sem parâmetro → int
def tabuadas():
    pivo = 1
    lista = []
    while pivo <= 10:
        tabuada = []
        numero = 1
        while numero <= 10:
                tabuada += [str(pivo) + '*' + str(numero) + '=' + str(pivo*numero)]
                numero += 1
        pivo += 1
        lista.append(tabuada)
    return lista
```

# EXERCÍCIO

**Reescreva a função tabuadas usando for.**

```
# função tabuadas que gera as tabuadas
# de multiplicação de 1 a 10
# sem parâmetro → int
def tabuadas():
    pivo = 1
    lista = []
    while pivo <= 10:
        tabuada = []
        numero = 1
        while numero <= 10:
                tabuada += [str(pivo) + '*' + str(numero) + '=' + str(pivo*numero)]
                numero += 1
        pivo += 1
        lista.append(tabuada)
    return lista
```

# EXERCÍCIO

1) Faça uma função que calcule o valor de N!, onde N é passado como parâmetro.

(Sem usar o factorial do módulo Math).

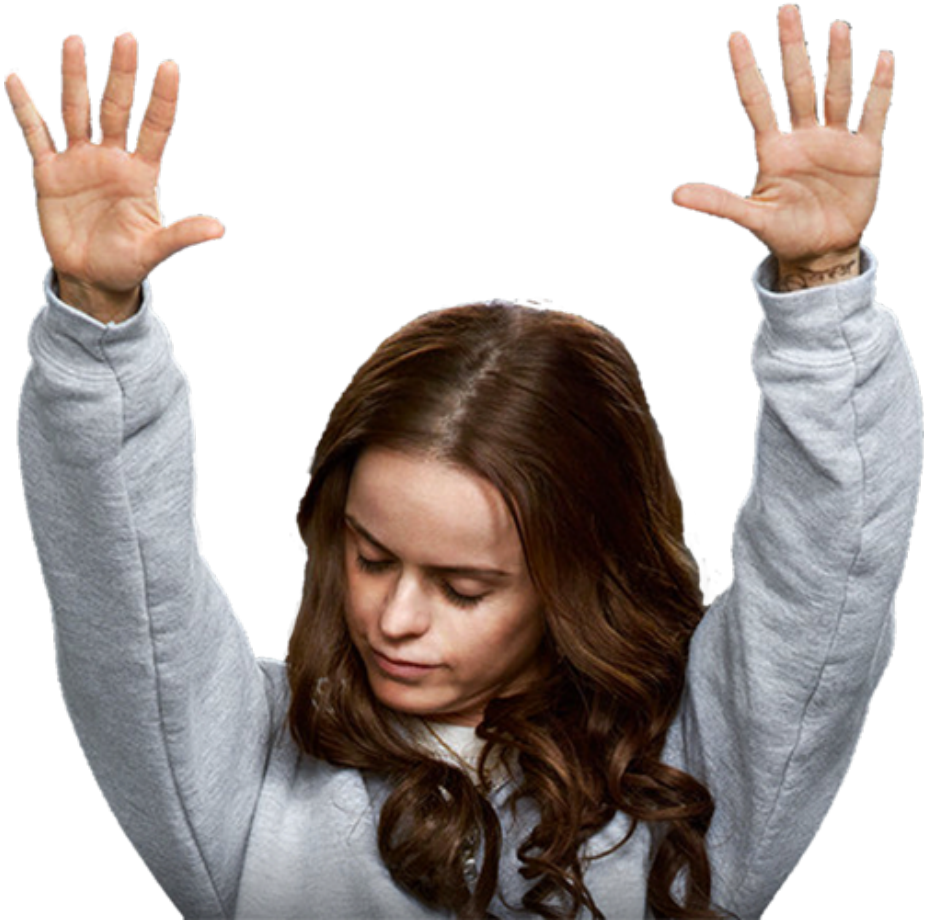
2) Faça uma função que calcule e retorne o valor de

S = (1/1)+(3/2)+(5/3)+(7/4)+....+(99/50)

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

S = (1/1)-(2/4)+(3/9)-(4/16)+....-(10/100)

# LIBERADOS!



# COMPUTAÇÃO 1 – PYTHON
# AULA 8 TEÓRICA

SLIDES BASEADOS NOS TRABALHOS:
AULAS TEÓRICAS DO DCC UFRJ
AULA DO CLAUDIO ESPERANÇA DO PESC