

Aula Teórica 07

Laço de Repetição **for**

Prof. Gustavo

Método de cálculo das notas de BCC 701

Ler as notas P1, P2, A1 e A2, valendo de 0 a 10 cada uma. Em uma função, calcular a média da seguinte forma: $AV1 = 30\% \text{ da Prova1} + 15\% \text{ da Atividade1}$,

$$AV2 = 40\% \text{ da Prova2} + 15\% \text{ da Atividade2}$$

Média = $AV1 + AV2$. Abaixo temos o programa para calcular a média de um aluno.

```
def calc_media(N1, N2, at1, at2):  
    av1 = 0.3*N1 + 0.15*at1  
    av2 = 0.4*N2 + 0.15*at2  
    M = av1 + av2  
    return M  
  
print("Início do programa")  
print("Aluno 1")  
P1 = float(input("Prova 1: "))  
P2 = float(input("Prova 2: "))  
a1 = float(input("Atividade 1: "))  
a2 = float(input("Atividade 2: "))  
media = calc_media(P1, P2, a1, a2)  
print("Média: {media:7.2f}")  
    If media >= 6.0:  
        print("Aprovado")  
    else:  
        print("Exame")  
print("Fim do programa")
```

Método de cálculo das notas de BCC 701

Ler as notas P1, P2, A1 e A2, valendo de 0 a 10, calcular a média da seguinte forma:

$AV1 = 30\% \text{ da Prova1} + 15\% \text{ da Atividade1}$, $AV2 = 40\% \text{ da Prova2} + 15\% \text{ da Atividade2}$

Média = $AV1 + AV2$. Abaixo temos o programa para calcular a média de um aluno.

```
def calc_media(N1, N2, at1, at2):  
    av1 = 0.3*N1 + 0.15*at1  
    av2 = 0.4*N2 + 0.15*at2  
    M = av1 + av2  
    return M  
  
print("Início do programa")  
print("Aluno 1")  
P1 = float(input("Prova 1: "))  
P2 = float(input("Prova 2: "))  
a1 = float(input("Atividade 1: "))  
a2 = float(input("Atividade 2: "))  
media = calc_media(P1, P2, a1, a2)  
print("Média: {media:7.2f}")  
    If media >= 6.0:  
        print("Aprovado")  
    else:  
        print("Exame")  
print("Fim do programa")
```

E se precisarmos
processar **a nota de 30
alunos**

Devemos repetir o
mesmo código por 30
vezes! Isso pode ser
feito com um comando
while ou com um for,
como veremos a seguir

Laço de Repetição **for**



- **Permite que uma operação seja executada várias vezes;**
- **Uma estrutura deste tipo também é chamada de laço (*loop* em inglês);**
- **Vamos estudar dois comandos de repetição:**
 1. **Laço controlado logicamente: **while** (enquanto)**
 2. **Laço controlado por contador : **for** (para)**

Em um laço controlado por contador, ou *laço contado*, os comandos contidos no corpo do laço são repetidos um número predeterminado de vezes: **for**

Sabe-se de antemão o número de vezes que o laço será repetido

No laço controlado logicamente, ou *laço indeterminado*, os comandos no corpo do laço são repetidos enquanto uma dada expressão lógica for verdadeira: **while**

Não se sabe, de antemão, quantas vezes o laço será repetido

Denomina-se iteração a repetição de um conjunto de comandos:

- Cada execução do corpo do laço, juntamente com a avaliação da condição do laço, é uma iteração

Fazer um programa para calcular das notas de BCC 701 de uma turma com 25 alunos:

```
print("Início do programa")
def calc_media(N1, N2, at1, at2):
    av1 = 0.3*N1 + 0.15*at1
    av2 = 0.4*N2 + 0.15*at2
    return av1 + av2
```

Repita bloco abaixo por 25 vezes:

```
print(f"Aluno {contador}:")          # o contador controla o
P1 = float(input("Prova 1: "))      # num. de repetições
P2 = float(input("Prova 2: "))
a1 = float(input("Atividade 1: "))
a2 = float(input("Atividade 2: "))
media = calc_media(P1, P2, a1, a2)
print(f"Média: {media:7.2}")
If media >= 6.0:
    print("Aprovado")
else:
    print("Exame")
```

Fim das repetições

```
print("Fim do programa")
```

Com esta estrutura podemos repetir o bloco dentro do laço quantas vezes for necessário.

O comando **for** pode ser definido da seguinte forma:

```
for variável in range(<início>, <fim + 1>, <passo>) :  
    <conjunto de comandos>  
<bloco após o for>
```

- **<conjunto de comandos>** comandos a serem executadas,
denominado **corpo do laço**
recuado à direita : usamos 4 espaços
- **variável in range(<início>, <fim+1>, <passo>)** *<intervalo>*
 1. <variável> recebe <início>
 2. Ao final de cada iteração o valor da <variável> é incrementada na quantidade <passo>.
 3. O laço termina quando o valor na <variável> chegar em <fim + 1>
- se <passo> = 1, ele pode ser omitido **var in range(<ini>, <fim>)**
- **for**, **in** e **range** são palavras reservadas da linguagem.

Fazer um programa para calcular das notas de BCC 701 de uma turma com 25 alunos:

```
print("Início do programa")
def calc_media(N1, N2, at1, at2):
    av1 = 0.3*N1 + 0.15*at1
    av2 = 0.4*N2 + 0.15*at2
    return av1 + av2
for cont in range(1, 26): # var inteira cont percorre o inter. [1, 26)
    print(f"Aluno {cont}:")      # cont controla o número de repetições
    P1 = float(input("Prova 1: "))
    P2 = float(input("Prova 2: "))
    a1 = float(input("Atividade 1: "))
    a2 = float(input("Atividade 2: "))
    media = calc_media(P1, P2, a1, a2)
    print(f"Média: {media:7.2}")
    If media >= 6.0:
        print("Aprovado")
    else:
        print("Exame")
```

Fim das repetições

#aqui já estamos fora do laço, fora da indentação do for.

```
print("Fim do programa")
```

Esta estrutura permite repetir o bloco dentro do laço quantas vezes for necessário. 26 -> valor desejado.

Faça um programa que imprima os 10 primeiros números Naturais (range)

```
print("Impressão dos 10 primeiro números Naturais")
for num in range(0, 10): # mesmo que range(10)
    print(f" {num}", end = ' ')
print("\nFim do programa")
```

Faça um programa que imprima os 10 primeiros números Naturais

```
print("Impressão dos 10 primeiro números Naturais")
for num in range(0, 10):
    print(f" {num}", end = '')
print("\nFim do programa")
```

Saída:

```
Impressão dos 10 primeiros números Naturais
0  1  2  3  4  5  6  7  8  9
Fim do programa
```

Faça um programa que imprima os 10 primeiros números Naturais

```
print("Impressão dos 10 primeiro números Naturais")
for num in range(0, 10):
    print(f" {num}", end = ' ')
print("\nFim do programa")
```

Saída:

```
Impressão dos 10 primeiros números Naturais
0  1  2  3  4  5  6  7  8  9
Fim do programa
```

Observe que o último número impresso foi o 9 e não o 10

Este programa tem uma característica indesejável, ele sempre imprimirá os 10 primeiros números inteiros

Isso pode ser flexibilizado pedindo que o usuário digite o total de números naturais impressos, de acordo com sua necessidade

Elabore um programa que gere e imprima os números Naturais até um dado valor k, k digitado pelo usuário.



```
print("Impressão dos k primeiro números Naturais")
k = int(input("Digite o valor limite: "))
for i in range(0, k):
    _print(f" {i}",end="") # end="" fica na msm linha
print("\nFim do programa")
```

Saída:

```
Impressão dos k primeiros números Naturais
Digite o total de inteiros: 12
0  1  2  3  4  5  6  7  8  9  10  11
Fim do programa
```

Elabore um programa que gere e imprima os números Naturais até um dado valor k, k digitado pelo usuário.



```
print("Impressão dos k primeiro números Naturais")
k = int(input("Digite o valor limite: "))
for i in range(0, k):
    print(f" {i}",end="") # end="" fica na msm linha
print("\nFim do programa")
```

Saída:

```
Impressão dos k primeiros números Naturais
Digite o total de inteiros: 12
0  1  2  3  4  5  6  7  8  9  10  11
Fim do programa
```

Saída:

```
Impressão dos k primeiros números Naturais
Digite o total de inteiros: 8
0  1  2  3  4  5  6  7
Fim do programa
```

Exemplos **for ... in range**

Codificando um laço de 10 iterações.

Neste caso, será armazenado na variável contadora dez valores, de acordo com a definição do comando **range**:

```
for x in range(11):  
    print(f"{x} ", end="")
```

Execução:

0 1 2 3 4 5 6 7 8 9 10

range(11) gera os valores de 0 até 10, os quais, são armazenados em x, um valor de cada vez, a cada iteração

Exemplos **for ... in range**

Codificando um laço de 10 iterações.

Neste caso, será armazenado na variável contadora dez valores, de acordo com a definição do comando **range**:

```
for x in range(11):  
    print(f"{x} ", end="")
```

Execução:

0 1 2 3 4 5 6 7 8 9 10

range(11) gera os valores de 0 até 10, os quais, são armazenados em x, um valor para cada iteração

```
for x in range(1,11):  
    print(f"{x} ", end="")
```

Execução:

1 2 3 4 5 6 7 8 9 10

range(1,11) gera os valores de 1 até 10

Exemplos **for ... in range**

Exemplos com incremento e decremento.

```
    ↙ contador    ↙ início    ↙ passo  
for i in range(1, 11, 2):  
    ↗ fim  
    print(f"{i} ", end="")
```

range(1, 11, 2) gera os valores de 1 até 10, com incrementos de 2.

Execução:

1 3 5 7 9

Exemplos **for ... in range**

Exemplos com incremento e decremento.

```
for i in range(1, 11, 2):  
    print(f"{i} ", end="")
```

range(1, 11, 2) gera os valores de 1 até 10, com incrementos de 2.

Execução:

1 3 5 7 9

```
for i in range(1, 11, 3):  
    print(f"{i} ", end="")
```

range(1, 11, 3) gera os valores de 1 até 10, com incrementos de 3.

Execução:

1 4 7 10

Exemplos **for ... in range**

Exemplos com incremento e decremento.

```
for x in range(1, 11, 2):  
    print(f"{x } ", end="")
```

range(1, 11, 2) gera os valores de 1 até 10, com incrementos de 2.

Execução:

1 3 5 7 9

```
for y in range(1, 11, 3):  
    print(f"{y } ", end="")
```

range(1, 11, 3) gera os valores de 1 até 10, com incrementos de 3.

Execução:

1 4 7 10

```
for z in range(8, 0, -2):  
    print(f"{z } ", end="")
```

range(8, 0, -2) gera os valores de 8 até 2, com exceção do 0.

Execução:

8 6 4 2

Exemplos **for ... in range**

Exemplos com incremento e decremento.

```
for x in range(1, 11, 2):  
    print(f"{x} ", end="")
```

range(1, 11, 2) gera os valores de 1 até 10, com incrementos de 2.

Execução:

1 3 5 7 9

```
for x in range(1, 11, 3):  
    print(f"{x} ", end="")
```

range(1, 11, 3) gera os valores de 1 até 10, com incrementos de 3.

Execução:

1 4 7 10

```
for x in range(8, 0, -2):  
    print(f"{x} ", end="")
```

range(8, 0, -2) gera os valores de 8 até 2, com exceção do 0.

Execução:

8 6 4 2

```
for x in range(0, 2, 0.4):  
    print(f"{x} ", end="")
```

Execução:

```
for x in range(0, 2, 0.4):  
TypeError: 'float' object cannot be  
interpreted as an integer
```

Faça um programa para calcular a soma dos 4 primeiros números naturais. Posteriormente mostrar a soma.

Faça um programa para calcular a soma dos 4 primeiros números naturais. Posteriormente mostrar a soma.

```
soma = 0          # limpando variável acumuladora
for num in range(0, 4):
    soma = soma + num #acumulando em soma (+=
num)
print(f"Soma dos {k} números naturais:
{soma}")
print("Fim do programa")
```

```
soma = 0          # limpando variável acumuladora
for num in range(0, 4):
    soma = soma + num #acumulando em soma (+= num)
print(f"Soma dos {k} números naturais: {soma}")
print("Fim do programa")
```

Simulação ou teste de mesa ou chinesinho

Primeira linha

soma = 0

Primeira iteração

num = 0;

soma = soma + num = 0 + 0 => soma = 0

Segunda iteração

num = 1;

soma = soma + num = 0 + 1 => soma = 1

Terceira iteração

num = 2;

soma = soma + num = 1 + 2 => soma = 3

Quarta iteração

num = 3;

soma = soma + num = 3 + 3 => soma = 6

Fim do laço

soma = 6 = 0 + 1 + 2 + 3

Faça um programa para calcular a soma dos números naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

Faça um programa para calcular a soma dos números naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

```
k = int(input("Digite o valor limite: "))  
soma = 0          # limpando variável acumuladora
```


Faça um programa para calcular a soma dos números Naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

```
k = int(input("Digite o valor limite: "))  
soma = 0          # limpando variável acumuladora  
for num in range(0, k):  
    soma = soma + num #acumulando em soma (+= num)
```

Faça um programa para calcular a soma dos números naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

```
k = int(input("Digite o valor limite: "))
soma = 0          # limpando variável acumuladora
for num in range(0, k):
    soma = soma + num #acumulando em soma (+= num)
print(f"Soma dos {k} números naturais: {soma}")
print("Fim do programa")
```

Faça um programa para calcular a soma dos números naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

```
k = int(input("Digite o valor limite: "))
soma = 0          # limpando variável acumuladora
for num in range(0, k):
    soma = soma + num #acumulando em soma (+= num)
print(f"Soma dos {k} números naturais: {soma}")
print("Fim do programa")
```

Saída:

```
Digite o valor limite: 10
Soma dos 10 números naturais: 55
Fim do programa
```

Faça um programa para calcular a soma dos números naturais até um certo valor k informado pelo usuário.

Posteriormente mostrar a soma.

```
k = int(input("Digite o valor limite: "))
soma = 0          # limpando variável acumuladora
for num in range(0, k):
    soma = soma + num #acumulando em soma (+= num)
print(f"Soma dos {k} números naturais: {soma}")
print("Fim do programa")
```

- **num** é a variável que percorrerá os naturais a serem somados
- **soma** é a variável onde será acumulada a soma dos naturais, portanto é chamada de **variável acumuladora**, ou **somadora**.
- a variável **soma** deve ser inicializada com valor zero (0) Elemento Neutro da Adição

Exercício: escreva um programa para ler as notas e calcular a média de uma turma de BCC701 com 40 alunos.

```
soma = 0          # zerando acumulador

for alu in range(1, 41): # percorre todos os alunos
    nota = float(input((f"Nota do aluno {alu}: "))
    soma = soma + nota

media = soma/40 #calcula média depois que termina o laço
print(f"A média da turma é: {media:5.2f}.")
```

Pode ser interessante escrever uma função valida-nota(n) para validar a nota digitada pelo usuário, que deve estar necessariamente no intervalo [0, 10]

```
def valida_notas (n):
    while n < 0 or n > 10:
        n = float(input((f"Nota {n} inválida\nDigite a nota novamente: "))
    return n
```

Exercício: escreva um programa para ler as notas e calcular a média de uma turma de BCC701 com 40 alunos.

```
def valida_nota (n):  
    while n < 0 or n > 10:  
        n = float(input((f"Nota {n} inválida\nDigite a nota  
        novamente: "))  
    return n  
  
# Programa Principal  
  
soma = 0      # zerando acumulador  
for alu in range(1, 41): # percorre todos os alunos  
    nota = float(input((f"Nota do aluno {alu}: "))  
    valida_nota(nota)  
    soma = soma + nota  
  
media = soma/40 #calcula média depois que termina o laço  
print(f"A média da turma é: {media:5.2f}.")
```

Rescreva o programa anterior, contando agora quantos alunos obtiveram nota abaixo de 6.0, na turma com 40 alunos.

```
soma = 0          # zerando acumulador
cont = 0          # zerando contador para notas < 6.0
for alu in range(1:40+1): # percorre todos os alunos
    print("Aluno: %g", alu)
    nota = input("digite a nota: ")
    soma = soma + nota
    if nota < 6.0:      # verificando nota
        cont = cont + 1 # contando notas < 6.0
media = soma/40 # estamos fora do laço for: indentação
print("A média da turma é {media}")
print("{cont} alunos obtiveram nota abaixo de 6.0")
```

Podemos melhorar este programa permitindo que o usuário defina o total de alunos da turma

Considere que temos várias turmas, devemos fazer um programa para cada turma, com diferentes <fim> do laço diferente? Não, basta ler o total de alunos no início do programa, a cada execução

```
soma = 0
cont = 0
# entrando com o total de alunos da turma
tot_alu = int(input("Entre com o total de alunos"))
for alu in range(1,tot_alu+1): #tot_alu = <fim>
    nota = float(input("Nota do aluno {alu}: "))
    soma = soma + nota
    if nota < 6.0 :
        cont = cont + 1
media = soma/tot_alu
print(f"A média da turma é {media:5.2f}.")
print("{cont} alunos obtiveram nota abaixo da média")
```


Calculado o Somatório de Séries Matemáticas

Considera a série abaixo:






$$(1) \quad S = \frac{1}{1} + \frac{2}{3} + \frac{3}{5} + \frac{4}{7} + \frac{5}{9} + \dots + \frac{50}{99}$$

- Utilizando-se o processo de somas acumuladas, podemos calcular o somatório com programa simples.
- A variável S será a acumuladora da soma das parcelas e será iniciada com zero.
- Um contador de parcelas é utilizado para indicar qual parcela deve ser somada (acumulada). Desta forma, quando o contador for 1, soma-se a primeira parcela; quando o contador for 2, soma-se a segunda parcela e assim sucessivamente.
- Cada parcela é definida por uma fração na forma: **numerador** / **denominador**.
- Para os cálculos podemos trabalhar com as variáveis **nume** para o numerador e **deno** para o denominador.
- As variáveis **nume** e **deno** serão inicializadas e incrementadas devidamente gerando cada parcela da soma.

Calculado o Somatório de Séries Matemáticas

$$S = \frac{1}{1} + \frac{2}{3} + \frac{3}{5} + \frac{4}{7} + \frac{5}{9} + \dots + \frac{50}{99}$$

$$S = 0 + \frac{1}{1} + \frac{2}{3} + \frac{3}{5} + \frac{4}{7} + \frac{5}{9}$$

 Soma p1 1/1	 Soma p2 2/3	 Soma p3 3/5	 Soma p4 4/7	 Soma p5 5/9
---	--	---	---	---

Temos que: **nume** = 1, 2, 3, 4, 5, ...

ou seja, **nume** começa com 1 e depois **nume** = **nume** + 1

Por outro lado, **deno** = 1, 3, 5, 7, 9, ...

neste caso, **deno** começa com 1 e depois **deno** = **deno** + 2

A soma **S** começa com zero e **S** = **S** + **nume**/**deno**

Código Série 1: primeira resolução

```
s = 0
nume = 1
deno = 1
for i in range(1, 51):
    s = s + nume / deno
    nume += 1      # nume = nume + 1
    deno += 2      # deno = deno + 1
print(f"Somatório com 50 parcelas: {s:9.4f}")
```

Execução:

```
Somatório com 50 parcelas:
26.4689
```

Código Série 1: segunda resolução

Nesta resolução, será usado o próprio índice contador i para montar as frações a serem somadas.

O contador $i = 1, 2, 3, \dots$ é o próprio numerador da fração e o denominador $1, 3, 5, \dots$ pode ser gerado a partir do contador pela fórmula: **$\text{deno} = 2*i - 1$** . Assim temos a seguinte sequencia:

i	1	2	3	...
$2*i - 1$	$2*1-1 = 1$	$2*2-1 = 3$	$2*3-1 = 5$	

```
s = 0
for i in range(1, 51):
    s = s + i / (2 * i - 1)
print(f"Somatório com 50 parcelas: {s:9.4f}")
```

Exercícios

1. Escreva um programa para mostrar os 10 primeiros números inteiros positivos e pares, e a sua respectiva soma.
2. Escreva um programa para mostrar os n primeiros números ímpares positivos e o seu respectivo produto; n informado pelo usuário.
3. Escreva um programa para calcular a média das duas notas de uma turma de BCC 701, cujo total de alunos é informado pelo usuários. Contar e informar:
 - a) o total de alunos aprovados ($\text{média} \geq 6$),
 - b) o total de alunos de exame ($4.0 \leq \text{média} < 6.0$)
 - c) e o total de alunos reprovados sem exame ($\text{média} < 4.0$)

Exercícios

4. Calculado o Somatório das Séries Matemáticas:

$$\text{a)} \quad S = \frac{2}{3} + \frac{4}{7} + \frac{6}{11} + \frac{8}{15} + \frac{10}{19} + \dots + \frac{50}{99}$$

$$\text{b)} \quad S = \frac{1}{1} - \frac{2}{3} + \frac{3}{5} - \frac{4}{7} + \frac{5}{9} - \dots - \frac{50}{99}$$

Dica: criar uma variável para controlar o sinal. A cada iteração ela muda sinal de +1 para -1 e assim sucessivamente.