

Programação Estruturada Linguagem Python

Professor Adjenor Cristiano
Queiroz
FAPAM - 2023

Aula 4



HORA DA
CORREÇÃO!





Exercício

Desenvolva um software que leia dois números e que pergunte qual operação você deseja realizar. Você deve poder calcular a soma (+), subtração (-), multiplicação (*) e divisão (/). Exiba o resultado da operação solicitada.

```
1 nun1 = float(input("Digite o Primeiro Número:\n"))
2 sinal = input("Digite o sinal da Operação:\n")
3 nun2 = float(input("Digite o Segundo Número:\n"))
4 if(sinal=="+" ):
5     print("%.2f + %.2f = %.2f"%(nun1, nun2, nun1+nun2))
6 elif(sinal=="-"):
7     print("%.2f - %.2f = %.2f"%(nun1, nun2, nun1-nun2))
8 elif(sinal=="*"):
9     print("%.2f * %.2f = %.2f"%(nun1, nun2, nun1*nun2))
10 elif(sinal=="/"):
11     if(nun2!=0):
12         print("%.2f / %.2f = %.2f"%(nun1,nun2,nun1/nun2))
13     else:
14         print("Não é possível efetuar divisão por 0")
15 else:
16     print("Sinal da operação Inválido!")|
```



Exercício

Desenvolva um software para aprovar o empréstimo bancário para compra de uma casa. O programa deve perguntar o valor da casa a comprar, o salário e a quantidade de anos a pagar. O valor da prestação mensal não pode ser superior a 30% do salário. Calcule o valor da prestação como sendo o valor da casa a comprar dividido pelo número de meses a pagar.

```
1 valor = float(input("Digite o Valor do Imóvel:\n"))
2 salario = float(input("Digite seu Salário:\n"))
3 anos = int(input("Digite quantos anos quer pagar:\n"))
4 prestacao = valor/(anos*12)
5 prestacaoMax = salario*0.3
6 prestacaoReal = str("%.2f"%prestacao).replace(".",",")
7 prestMaxReal = str("%.2f"%prestacaoMax).replace(".",",")
8 if (prestacaoMax>=prestacao):
9     print("Financiamento Aprovado!")
10    print("Valor Prestação: %s"%prestacaoReal)
11    print("Valor Máximo Prestação Permitido:", end="")
12    print("%s"%prestMaxReal)
13 else:
14    print("Financiamento Reprovado!")
15    print("Valor Prestação: %s"%prestacaoReal)
16    print("Valor Máximo Prestação Permitido: ", end="")
17    print("%s"%prestMaxReal)
```



Exercício

Desenvolva um software para analisar se um aluno foi aprovado. O software deve receber 4 notas (entre 0 e 25) e ao final atestar as seguintes possibilidades:

- Se o total for maior ou igual 80 - ALUNO APROVADO - EXCELENTE
- Se o total for entre 60 e 79 - ALUNO APROVADO
- Se o total for entre 40 e 59 - ALUNO EM RECUPERAÇÃO
- Se o total for menor que 40 - ALUNO REPROVADO
- Se alguma das notas estiver fora do intervalo (entre 0 e 25) o software deve exibir a mensagem de “Valor inválido” e encerrar.



```
1 print ("\n*****Boletim Escolar*****\n")
2 n1 = int(input("Primeiro Nota: "))
3 n2 = int(input("Segunda Nota: "))
4 n3 = int(input("Terceiro Nota: "))
5 n4 = int(input("Quarta Nota: "))
6 total = n1+n2+n3+n4
7
8 if total >= 80:
9     print ("Nota: %i - ALUNO APROVADO - EXCELENTE!"%total)
10 elif total >= 60:
11     print ("Nota: %i - ALUNO APROVADO"%total)
12 elif total >= 40:
13     print ("Nota: %i - ALUNO EM RECUPERAÇÃO"%total)
14 else:
15     print ("Nota: %i - ALUNO REPROVADO"%total)
```

```
1 print ("\n*****Boletim Escolar*****\n")
2 n1 = int(input("Primeiro Nota: "))
3 if (n1<=25) and (n1>0):
4     n2 = int(input("Segunda Nota: "))
5     if (n2<=25) and (n2>0):
6         n3 = int(input("Terceiro Nota: "))
7         if (n3<=25) and (n3>0):
8             n4 = int(input("Quarta Nota: "))
9             if (n4<=25) and (n4>0):
10                 total = n1+n2+n3+n4
11                 if total >= 80:
12                     print ("Nota: %d - ALUNO APROVADO - EXCELENTE!"%total)
13                 elif total >= 60:
14                     print ("Nota: %d - ALUNO APROVADO"%total)
15                 elif total >= 40:
16                     print ("Nota: %d - ALUNO EM RECUPERAÇÃO"%total)
17                 else:
18                     print ("Nota: %d - ALUNO REPROVADO"%total)
19             else:
20                 print("Nota Inválida")
21         else:
22             print("Nota Inválida")
23     else:
24         print("Nota Inválida")
25 else:
26     print("Nota Inválida")
```




Exercício

Desenvolva um software de lanchonete que apresente para o cliente um menu com ao menos 4 opções de salgado para que ele escolha, depois apresente ao menos 4 opções de bebidas. Ao final, o software deve mostrar na tela o valor total da compra e os itens comprados.



```
1 opcao = 0
2 total = 0
3 pedido = ''
4
5 menu = "Menu:\n1 - Coxinha = R$ 6,00 \n"
6 menu += "2 - Pão de Queijo = R$ 4,00"
7 menu += "\n3 - Pastel = R$ 4,50"
8 menu += "\n4 - Empada = R$5,00 "
9 menu += "\n5 - Nenhuma Opção\n"
10
11 opcao = int(input(menu))
12
13 if(opcao==1):
14     total+=6
15     pedido+="1 - Coxinha - R$ 6,00  \n"
16 elif(opcao==2):
17     total+=4
18     pedido += "1 - Pão de Queijo = R$ 4,00 \n"
19 elif(opcao==3):
20     total+=4.5
21     pedido += "1 - Pastel = R$ 4,50 \n"
22 elif(opcao==4):
23     total+=5
24     pedido += "1 - Empada = R$ 5,00 \n"
```



```
26 menu = "Bebidas:\n1 - Coca Cola = R$ 6,00"
27 menu += "\n2 - Suco Laranja = R$ 8,00"
28 menu += "\n3 - Suco Caju = R$ 8,50"
29 menu += "\n4 - Cerveja = R$ 10,00 "
30 menu += "\n5 - Nenhuma Opção\n"
31
32 opcao = int(input(menu))
33
34 if(opcao==1):
35     total+=6
36     pedido+="1 - Coca Cola = R$ 6,00  \n"
37 elif(opcao==2):
38     total+=8
39     pedido += "1 - Suco Laranja = R$ 8,00 \n"
40 elif(opcao==3):
41     total+=8.5
42     pedido += "1 - Suco Caju = R$ 8,50\n"
43 elif(opcao==4):
44     total+=10
45     pedido += "1 - Cerveja = R$ 10,00  \n"
46
47 totalReal = str("%.2f"%total).replace(".",",")
48 print("Seu pedido: \n\n%s"%pedido)
49 print("O total do seu pedido é R$ %s"%totalReal)
```

Estruturas de Repetição

Estruturas de Repetição

Estruturas de Repetição

Estruturas de Repetição

Estruturas de Repetição

Estruturas de Repetição

Estruturas de Repetição




JÁ ARRUMOU
SUA CAMA?

JÁ LIMPOU
O QUARTO?

JÁ COLOCOU O
LIXO PRA FORA?

JÁ LAVOU A
LOUÇA?





Repetições representam a base de vários programas. São utilizadas para executar a mesma parte de um programa várias vezes, normalmente dependendo de uma condição.


Por exemplo, para imprimir três números na tela, poderíamos escrever um programa como o apresentado na listagem 5.1.

► Listagem 5.1 – Imprimindo de 1 a 3

```
print(1)
```

```
print(2)
```

```
print(3)
```



Podemos imaginar que para imprimir três números, começando de 1 até o 3, devemos variar `print(x)`, onde `x` varia de 1 a 3. Vejamos outra solução para o problema na listagem 5.2.

► Listagem 5.2 – Imprimindo de 1 a 3 usando uma variável

```
x=1
```


```
print(x)
```

```
x=2
```

```
print(x)
```

```
x=3
```

```
print(x)
```



Outra solução seria incrementar o valor de x após cada print. Vejamos essa solução na listagem 5.3.

► Listagem 5.3 – Imprimindo de 1 a 3 incrementando

```
x=1
```


```
print(x)
```

```
x=x+1
```


```
print(x)
```

```
x=x+1
```

```
print(x)
```



Porém, se o objetivo fosse escrever 100 números, a solução não seria tão agradável, pois teríamos que escrever pelo menos 200 linhas! A estrutura de repetição aparece para nos auxiliar a resolver esse tipo de problema.



Uma das estruturas de repetição do Python é o `while`, que repete um bloco enquanto a condição for verdadeira. Seu formato é apresentado na listagem 5.4, onde condição é uma expressão lógica, e bloco representa as linhas de programa a repetir enquanto o resultado da condição for verdadeiro.

Vamos Praticar:

```
x=1 ❶  
while x<=3: ❷  
    print(x) ❸  
    x = x + 1 ❹
```

EXERCÍCIOS:

Exercício 5.1 Modifique o programa para exibir os números de 1 a 100.

Exercício 5.2 Modifique o programa para exibir os números de 50 a 100.

Exercício 5.3 Faça um programa para escrever a contagem regressiva do lançamento de um foguete. O programa deve imprimir 10, 9, 8, ..., 1, 0 e Fogo! na tela.



Interrompendo a repetição:

Embora muito útil, a estrutura `while` só verifica sua condição de parada no início de cada repetição. Dependendo do problema, a habilidade de terminar `while` dentro do bloco a repetir pode ser interessante.

A instrução `break` é utilizada para interromper a execução de `while` independentemente do valor atual de sua condição.

Vamos Praticar:

► Listagem 5.13 – Interrompendo a repetição

```
s=0
```

```
while True: ❶
```

```
    v=int(input("Digite um número a somar ou 0 para sair:"))
```

```
    if v==0:
```

```
        break ❷
```

```
    s = s+v ❸
```

```
print(s) ❹
```



Contadores:

O poder das estruturas de repetição é muito interessante, principalmente quando utilizamos condições com mais de uma variável. Imagine um problema onde deveríamos imprimir os números inteiros entre 1 e um valor digitado pelo usuário.

Vamos modificar o programa da listagem 5.5 de forma que o último número a imprimir seja informado pelo usuário. O programa já modificado é apresentado na listagem 5.6

Vamos Praticar:

► Listagem 5.6 – Impressão de 1 até um número digitado pelo usuário

```
fim=int(input("Digite o último número a imprimir:")) ❶
```

```
x = 1
```

```
while x <= fim: ❷
```

```
    print(x) ❸
```

```
    x = x + 1 ❹
```



Acumuladores:

Nem só de contadores precisamos. Em programas para calcular o total de uma soma, por exemplo, precisaremos de acumuladores.

A diferença entre um contador e um acumulador é que nos contadores o valor adicionado é constante e, nos acumuladores, variável. Vejamos um programa que calcula a soma de 10 números na listagem 5.11. Nesse caso, soma ❶ é um acumulador e n ❷ é um contador.



Vamos Praticar:

► Listagem 5.11 – Soma de 10 números

```
n = 1
soma = 0
while n <= 10:
    x = int(input("Digite o %d número:"%n))
    soma = soma + x ❶
    n = n + 1 ❷
print("Soma: %d"%soma)
```



Repetições aninhadas:

Podemos combinar vários while de forma a obter resultados mais interessantes, como a repetição com incremento de duas variáveis. Imagine imprimir as tabuadas de multiplicação de 1 a 10. Vejamos como fazer isso, lendo a listagem do programa 5.15

Vamos Praticar:

► Listagem 5.15 – Impressão de tabuadas

```
tabuada=1
```

```
while tabuada <= 10: ❶
```

```
    número = 1 ❷
```

```
    while número <= 10: ❸
```

```
        print("%d x %d = %d" % (tabuada, número, tabuada * número))
```

```
        número+=1 ❹
```

```
    tabuada+=1 ❺
```




Tabuada com apenas um While

► Listagem 5.16 – Impressão de tabuadas sem repetições aninhadas

```
tabuada=1
```

```
número=1
```

```
while tabuada <= 10:
```

```
    print("%d x %d = %d" % (tabuada, número, tabuada * número))
```

```
    número+=1
```

```
    if número == 11:
```

```
        número = 1
```

```
        tabuada+=1
```



EXERCÍCIOS:

- 1 - Desenvolva um software que receba do usuário uma quantidade e imprima na tela todos os números pares de 0 até o número digitado.
- 2 - Altere o software do exercício anterior para que o sistema receba também um tipo (P para Pares e I para Impares) e imprima apenas os números de acordo com a escolha.
- 3 - Valide a opção digitada, caso o usuário digite um valor diferente se P ou I, solicite a ele que digite novamente. DICA: Utilize repetições aninhadas.



EXERCÍCIOS:

4 - Escreva um programa que leia um número e verifique se é ou não um número primo. Para fazer essa verificação, calcule o resto da divisão do número por 2 e depois por todos os números ímpares até o número lido. Se o resto de uma dessas divisões for igual a zero, o número não é primo. Observe que 0 e 1 não são primos e que 2 é o único número primo que é par.



EXERCÍCIOS:

5 - Desenvolva um software de restaurante que apresente ao menos 4 tipos de produtos e a opção de finalizar o pedido. O software deve perguntar ao usuário se ele deseja adicionar outro produto. Ao final ele deve mostrar na tela todos os produtos que o cliente comprou e o total do pedido.

DICA: Utilize uma variável String para armazenar os produtos escolhidos e uma variável float para armazenar o total do pedido. Apresente o total formatado para Real.



Bibliografia

- MENEZES, Nilo Ney Coutinho - Introdução à Programação com Python: Algoritmos e Lógica de Programação Para Iniciantes, 3ª Edição – 2019, Editora: Novatec Editora, ISBN-10: 8575227181
- SHAW, Zed A – Aprenda Python 3 do jeito certo, 1ª Edição – 2019, Editora: Alta Books, ISBN: 978-85-508-0473-6.
- <https://docs.python.org/pt-br/3/>
- https://www.ime.usp.br/~leo/mac2166/2017-1/introducao_estrutura_basica_c_python.html
- <http://python42.com.br/?p=176>
- <https://www.youtube.com/@CursoemVideo>
- <https://panda.ime.usp.br/cc110/static/cc110/>
-