Programação Estruturada Linguagem Python

Professor Adjenor Cristiano Queiroz
FAPAM

Termos de Uso

Todo conteúdo desta apresentação é para uso exclusivo em sala de aula. Este material não poderá ser reproduzido, copiado, publicado, transmitido, vendido ou distribuído e nem sequer modificado, seja ou não para fins comerciais, sem prévia autorização expressa e formal da empresa.

Aula 8


HORA DA
CORREÇÃO!

Exercícios:

1 - Altere o programa anterior de modo que ele receba os valores para preencher a lista, a cada valor digitado o sistema ofereça ao usuário a opção de imprimir a lista com seus índices ([0] - 4). O programa também deve oferecer ao usuário a opção de adicionar mais um valor ou excluir um valor. Ao finalizar o sistema deve apresentar na tela **todos os números digitados, o maior, menor e a média.**

Exercicio1 - Lista add del.py - F:\Fapam\Programação Estruturada\Aulas\Aula 13 - Aula 8 Python\Exercicio1 - Lista add del.py (3.8.0)

File Edit Format Run Options Window Help

```
1 Lista = []
2 maior = 0
3 menor = 0
4 total = 0
5 menu = "O que desesa Fazer?\n"
6 menu += "[A] - Add um Valor \n"
7 menu += "[R] - Remover um Valor \n"
8 menu += "[V] - Visualizar a Lista \n"
9 menu += "[E] - Encerrar \n"
```

```
10 while(True):
11     opcao = input(menu)
12 
13     if(opcao.upper()=="A"):
14         nun = input("Digite o número que deseja add:")
15         Lista.append(int(nun))
16 
17     elif(opcao.upper()=="V"):
18         for cont, item in enumerate(Lista):
19             print ("[",cont,"] - ",item)
20 
```

```
21      elif(opcao.upper()=="R"):
22          for cont, item in enumerate(Lista):
23              print ("[",cont,"] - ",item)
24          remover = int(input("Digite o índice que deseja remover"))
25
26          if(remover<len(Lista)):
27              del Lista[remover]
28          else:
29              print("Índice Inválido!")
30      elif(opcao.upper()=="E"):
31          break
32
```

```
33 for cont, item in enumerate(Lista):
34     if((item > maior)or(cont==0)):
35         maior=item
36     if((item < menor)or(cont==0)):
37         menor=item
38     total +=item
39     print ("[",cont,"] - ",item)
40 print()
41 print("Maior: ",maior)
42 print("Menor: ",menor)
43 print("Total: ",total)
44 print("Média: ",total/(cont+1))
```

Exercícios:

2 - Desenvolva um programa para separar os valores da lista [9,8,7,12,0,13,21,35,6,11,1] em duas listas, uma com os pares e outra com os ímpares.

```
1  Lista = [9,8,7,12,0,13,21,35,6,11,1]
2  ListaImpar=[]
3  ListaPar=[]
4
5  for cont, item in enumerate(Lista):
6      if(item%2==0):
7          ListaPar.append(item)
8      else:
9          ListaImpar.append(item)
10
11 print("Lista Par: ", ListaPar)
12 print("Lista Impar: ", ListaImpar)
13
```

Exercícios:

3 - Altere o software do exercício anterior de forma que ele receba os dados do usuário para criar na lista de compras. O sistema deve oferecer ao usuário a possibilidade de exibir a lista de compras, adicionar e remover itens, além de mostrar os totais (valor total da compra e quantidade total de ítens).

```
1 Lista = []
2 menu = "O que desesa Fazer?\n"
3 menu += "[A] - Add um Item \n"
4 menu += "[R] - Remover um Item \n"
5 menu += "[V] - Visualizar a Lista \n"
6 menu += "[E] - Encerrar \n"
```

```
while(True):
    opcao = input(menu)

    if(opcao.upper()=="A"):
        itemadd = []
        item = input("Digite o item que deseja add:")
        itemadd.append(item)
        qtd = float(input("Digite a quantidade que deseja add:"))
        itemadd.append(qtd)
        valor = float(input("Digite o valor para add:"))
        itemadd.append(valor)
        Lista.append(itemadd)
```

```
20    elif(opcao.upper()=="V"):
21        qtdTotal = 0
22        vlrTotal = 0
23        print("-----------------------------------")
24        for Linha in Lista:
25            print("Produto: %s" % Linha[0])
26            print("Quantidade: %5.2f" % Linha[1])
27            print("Preço: R$%5.2f" % Linha[2])
28            print("-----------------------------------")
29            qtdTotal+= Linha[1]
30            vlrTotal+= Linha[2]
31        print()
32        print("*****************************************")
33        print("Qtd Itens: %5.2f" % qtdTotal)
34        print("Vlr Total: R$ %5.2f" % vlrTotal)
35        print("*****************************************")
```

```
36    elif(opcao.upper()=="R"):
37       for cont, item in enumerate(Lista):
38          print ("[",cont,"] - ",item[0])
39       remover = int(input("Digite o índice que vc deseja remover"
40
41       if(remover<len(Lista)):
42          del Lista[remover]
43       else:
44          print("Índice Inválido!")
45    elif(opcao.upper()=="E"):
46       break
```

```
48 qtdTotal = 0
49 vlrTotal = 0
50 print("-----------------------------------")
51 for Linha in Lista:
52     print("Produto: %s" % Linha[0])
53     print("Quantidade: %5.2f" % Linha[1])
54     print("Preço: R$ %5.2f" % Linha[2])
55     print("-----------------------------------")
56     qtdTotal+= Linha[1]
57     vlrTotal+= Linha[2]
58 print()
59 print("*****************************************")
60 print("Qtd Itens: %5.2f" % qtdTotal)
61 print("Vlr Total: R$ %5.2f" % vlrTotal)
62 print("*****************************************")
```

Ordenação

Até agora, os elementos de nossas listas apresentaram a mesma ordem em que foram digitados, sem qualquer ordenação. Para ordenar uma lista, realizaremos uma operação semelhante à da pesquisa, mas trocando a ordem dos elementos quando necessário.

Um algoritmo muito simples de ordenação é o “Bubble Sort”, ou método de bolhas, fácil de entender e aprender.

A ordenação pelo método de bolhas consiste em comparar dois elementos a cada vez. Se o valor do primeiro elemento for maior que o do segundo, eles trocarão de posição. Essa operação é então repetida para o próximo elemento até o fim da lista.

Esse método tem outra propriedade, que é posicionar o maior elemento na última posição da lista, ou seja, sua posição correta. Isso permite eliminar um elemento do fim da lista a cada passagem completa.

```
1  L=[7,4,3,12,8]
2  fim=5
3  while fim > 1:
4      trocou=False
5      x=0
6      while x<(fim-1):
7          if L[x] > L[x+1]:
8              trocou=True
9              temp=L[x]
10             L[x]=L[x+1]
11             L[x+1]=temp
12         x+=1
13     if not trocou:
14         break
15     fim-=1
16 for e in L:
17     print(e)
```

Algoritmos de ordenação comuns:

- Selection Sort (ordenação por seleção)
- Insertion Sort (ordenação por inserção)
- Merge Sort (ordenação por intercalação)
- Quick Sort (ordenação rápida)
- Heap Sort
- Counting Sort
- Radix Sort
- Bucket Sort

Algoritmos de ordenação comuns:

Trabalho AVALIATIVO - Valor 8 pontos:

Em grupo de 4 ou 5 Alunos (só pode haver algum grupo de 5 caso tenham mais de 32 alunos), cada grupo deve estudar um dos algoritmos de ordenação, desenvolver uma apresentação para sala e apresentar a ordenação por este utilizando a linguagem Python. Também deve explicar suas vantagens, desvantagens e principais usos.

Algoritmos de ordenação comuns:

Trabalho AVALIATIVO - Valor 8 pontos:

O trabalho deve ser apresentado no dia 06/06 por todos o membros do grupo. cada grupo terá entre 10 e 15 minutos para sua apresentação. Serão avaliados conteúdo e domínio da matéria de todos os membros do grupo e a avaliação será individualizada.

Funções

Podemos definir nossas próprias funções em Python. Sabemos como usar várias funções, como len, int, float, print e input. Neste capítulo, veremos como declarar novas funções e utilizá-las em programas.

Para definir uma nova função, utilizaremos a instrução def. Vejamos como declarar uma função de soma que recebe dois números como parâmetros e os imprime na tela (Listagem 8.1).

▶ Listagem 8.1 – Definição de uma nova função

```
def soma(a,b): ❶
    print(a+b) ❷
soma(2,9) ❸
soma(7,8)
soma(10,15)
```

Funções são especialmente interessantes para isolar uma tarefa específica em um trecho de programa. Isso permite que a solução de um problema seja reutilizada em outras partes do programa, sem precisar repetir as mesmas linhas. O exemplo anterior utiliza dois parâmetros e imprime sua soma. Essa função não retorna valores como a função len ou a int. Vamos reescrever essa função de forma que o valor da soma seja retornado (Listagem 8.2).

▶ Listagem 8.2 – Definição do retorno de um valor

```
def soma(a,b):
    return(a+b) ❶
print(soma(2,9))
```

Veja que agora utilizamos a instrução return ❶ para indicar o valor a retornar.

Observe também que retiramos o print da função. Isso é interessante porque a soma e a impressão da soma de dois números são dois problemas diferentes.

Nem sempre vamos somar e imprimir a soma, por isso, vamos deixar a função realizar apenas o cálculo. Se precisarmos imprimir o resultado, podemos utilizar a função print, como no exemplo.

Vejamos outro exemplo, uma função que retorne verdadeiro ou falso, dependendo se o número é par ou ímpar (Listagem 8.3).

▶ Listagem 8.3 – Retornando se valor é par ou não

```
def épar(x):
    return(x%2==0)
print(épar(2))
print(épar(3))
print(épar(10))
```

Imagine agora que precisamos definir uma função para retornar a palavra par ou ímpar.

```
def épar(x):
    return(x%2==0)
def par_ou_ímpar(x):
    if épar(x): ❶
        return "par" ❷
    else:
        return "ímpar" ❸
print(par_ou_ímpar(4))
print(par_ou_ímpar(5))
```

Em ❶ chamamos a função épar dentro da função par_ou_ímpar.

Observe que não há nada de especial nessa chamada: apenas repassamos o valor de x para a função épar e utilizamos seu retorno no if.

Se a função retornar verdadeiro, retornaremos “par” em ❷; caso contrário, “ímpar” em ❸. Utilizamos dois return na função par_ou_ímpar.

A instrução return faz com que a função pare de executar e que o valor seja retornado imediatamente ao programa ou função que a chamou. Assim, podemos entender a instrução return como uma interrupção da execução da função, seguido do retorno do valor. As linhas da função após a instrução return são ignoradas **de forma similar à instrução break dentro de um while ou for.**

Exercício 8.1 Escreva uma função que retorne o maior de dois números.

Valores esperados:

máximo(5,6) == 6

máximo(2,1) == 2

máximo(7,7) == 7

Exercício 8.2 Escreva uma função que receba dois números e retorne `True` se o primeiro número for múltiplo do segundo.

Valores esperados:

múltiplo(8,4) == `True`

múltiplo(7,3) == `False`

múltiplo(5,5) == `True`

Exercício 8.3 Escreva uma função que receba o lado (l) de um quadrado e retorne sua área ($A = lado^2$).

Valores esperados:

área_quadrado(4) == 16

área_quadrado(9) == 81

Exercício 8.4 Escreva uma função que receba a base e a altura de um triângulo e retorne sua área (A = (base x altura)/2).

Valores esperados:

área_triângulo(6, 9) == 27

área_triângulo(5, 8) == 20

▶ Listagem 8.5 – Pesquisa em uma lista

```
def pesquise(lista, valor):
    for x,e in enumerate(lista):
        if e == valor:
            return x ❶
    return None ❷
L=[10, 20, 25, 30]
print(pesquise(L, 25))
print(pesquise(L, 27))
```

▶ Listagem 8.6 – Cálculo da média de uma lista

```
def soma(L):
    total=0
    for e in L:
        total+=e
    return total
def media(L):
    return(soma(L)/len(L))
```

▶ Listagem 8.7 – Soma e cálculo da média de uma lista

```
def média(L):
    total=0
    for e in L:
        total+=e
    return total/len(L)
```

▶ Listagem 8.8 – Como não escrever uma função

```
def soma(L):
    total=0
    x = 0
    while x<5:
        total+=L[x]
        x+=1
    return total
L=[1,7,2,9,15]
print(soma(L)) ❶
print(soma([7,9,12,3,100,20,4])) ❷
```

Dica: Python tem funções para calcular a soma, o máximo e o mínimo de uma lista.

```
>>> L=[5,7,8]
>>> sum(L)
20
>>> max(L)
8
>>> min(L)
5
```

Atividades:

1 – Faça uma função que retorne o reverso de um número inteiro informado. Por exemplo: 127 -> 721.

2 – Faça uma função que informe a quantidade de dígitos de um determinado número inteiro informado. Exemplo: 127 -> 3

3 – Faça uma função que computa a potência ab para valores a e b (assuma números inteiros) passados por parâmetro (não use o operador **).

4 – Faça uma função que converta da notação de 24 horas para a notação de 12 horas. Por exemplo, o programa deve converter 14:25 em 2:25 P.M.

5 – Faça uma função que converta uma data em formato brasileiro para americano e uma que converta data americana para brasileira. (Data BR = DD/MM/AAAA – Data AM = YYYY-MM-DD)

6 – Faça uma função que converta um valor para reais. Ex: 10 -> R$ 10,00 / 12.682 -> R$ 12,68.

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/
- https://www.freecodecamp.org/portuguese/news/algoritmos-de-ordenacao-explicados-com-exemplos-em-python-java-e-c/