



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS

Escola Politécnica - Curso de Engenharia de Software

12413 - ALGORITMOS DE PROGRAMAÇÃO, PROJETOS E COMPUTAÇÃO

ASSUNTOS:

- correção exercício
- estrutura de dados: lista

Profa. Angela de Mendonça Engelbrecht

angel@puc-campinas.edu.br

profa.angela@gmail.com

Considerações:

A partir desse ponto, considerando que já possuímos conceitos básicos de programação e de uma ferramenta básica, para a construção dos algoritmos e as soluções para os problemas, vamos utilizar **a ideia** da *Metodologia Ativa – Aprendizado Baseado em Problemas*.

Digo a ideia, uma vez que não vamos avançar em áreas da Engenharia de Software que envolvam conhecimentos que apenas obterão nos próximos semestres. O Projeto Integrador trabalha essa convergência de conhecimentos.

Passaremos a **olhar um problema** e buscar alternativas para a solução e aprender novas possibilidades, com **FOCO** na **estruturação dos dados**, para a organização e armazenamento das informações, pertinentes ao problema, BASE FUNDAMENTAL para a criação de um algoritmo.

Partir do seguinte problema:

Construir um programa que faz a leitura das **notas** de **N alunos**. Calcula a **média** da classe e **imprime** a média da classe.

Análise do problema:**1. Estrutura dos dados - variáveis e seus tipos:**

- **N**: para a quantidade de alunos – *inteiro*
- **nota**: para a leitura da nota - *real*
- **soma**: para acumular as notas lidas – *real*
- **media**: receber o resultado do cálculo da média - *real*

4. Sobre o fluxo do programa:

- para ler **N** notas, **precisamos** de **processo repetitivo** – por ser quantidade, usar **for**;

6. sobre os comandos dentro do processo repetitivo:

- leitura de cada **NOTA**;
- efetuar a **soma** da **NOTA** lida, acumulando na variável **soma**;

2. Entrada de Dados:

- **N**: número de notas
- **nota** de **N** alunos

3. Cálculos necessários:

- Somatório das **N** notas: **soma = soma + nota**
- depois de calcular a soma de **TODAS** as notas – calcular a média: **media = soma/N**

5. validações necessárias:

- o valor de **N** deve ser positivo
- a nota digitada deve estar entre **0** (zero) e **10** (dez)
(*) **no exercício não fiz a validação da nota**

6. resultado esperado:

- a impressão da média da classe calculada

File Edit Format Run Options Window Help

```
''' calcula media da classe e imprime notas e média'''

''' leitura do NÚMERO de notas que serão digitadas'''
while True:
    N = int(input('Número de notas (Maior do que ZERO):'))
    if N>0:
        break
    else: print('Quantidade de Notas Inválida')

'''iniciando a variável que acumulará as somas
com ZERO - elemento neutro da soma'''
soma =0

for i in range(N):
    nota = float(input('Nota: '))
    soma+=nota

Média = soma/N

print(f'Média da Classe: {Média:.1f}')
```

```
===== RESTART:
Número de notas (Maior do que ZERO):4
Nota: 5
Nota: 6
Nota: 7
Nota: 8.5
Média da Classe: 6.6
>>>
```

PROBLEMA – NIVEL I:

Construir um programa que faz a leitura das **notas** de **N alunos**. Calcula a **média** da classe e imprime **a lista de notas** e a média da classe.

Análise do problema:

1. Estrutura dos dados - variáveis e seus tipos:

- **N**: para a quantidade de alunos – *inteiro*
- guardar as **N notas** lidas – precisamos de uma **estrutura que guarde** um conjunto de informações → **para o exercício** vamos usar estrutura de dados **LISTA**
- **soma**: para acumular as notas lidas – *real*
- **media**: receber o resultado do cálculo da média - *real*

antes de continuar, vamos examinar a estrutura de dados LISTA e suas características – com esse conhecimento, poderemos empregá-la na solução do problema.

listas - *list*

junto com outros tipos, são caracterizadas por armazenar coleções de itens.

- representação – seus elementos são **indexados** a partir do ZERO;
- **usados []** para seus elementos, separados por vírgulas;
- **mutável**;
- pode conter repetição de seus valores;
- construtor: *list()*
- ***type()*** -> ***<class 'list'>***;

Exemplos:

```
>>> # [] --> caracteriza uma lista vazia
>>> L = []
>>> type(L)
<class 'list'>
>>> len(L)
0
```

listas - *list*

Exemplos:

```
>>> ListaNumeros= [1,2,3,4,5,6]
>>> ListaNumeros
[1, 2, 3, 4, 5, 6]
>>> type(ListaNumeros)
<class 'list'>
>>> len(ListaNumeros)
6
>>> ListaNumeros[0]
1
>>> ListaNumeros[len(ListaNumeros)-1]
6
>>> ListaNumeros[len(ListaNumeros)]
Traceback (most recent call last):
  File "<pyshell#63>", line 1, in <module>
    ListaNumeros[len(ListaNumeros)]
IndexError: list index out of range
```


pode ser uma lista de nomes

```
>>> ListaNomes=["Maria", "João", "José", "Paulo"]
>>> type(ListaNomes)
<class 'list'>
>>> len(ListaNomes)
4
>>> ListaNomes[2:]
['José', 'Paulo']
>>> |
```

partes de uma lista

pode ser uma lista de tipos variados

listas dentro de listas

```
>>> ListaSubLista = [1, 2, [10, 20], "Casa", 45.54]
>>> type(ListaSubLista)
<class 'list'>
>>> len(ListaSubLista)
5
>>> for elemento in ListaSubLista:
    print(elemento)

1
2
[10, 20]
Casa
45.54
```

iterável → pode estar num processo iterativo - repetitivo

acessando valores em sublistas

dois valores na posição [2]

[0] [1] [2] [3] [4] [5]

```
>>> lista = ['apontador', 'régua', ['caneta azul', 'caneta vermelha'], 'lápis', 'caderno', 'caderno']
```

```
>>> lista[0]
```

```
'apontador'
```

```
>>> lista[2]
```

```
['caneta azul', 'caneta vermelha']
```

```
>>> lista[2][0]
```

```
'caneta azul'
```

```
>>> lista[2][1]
```

```
'caneta vermelha'
```

```
>>> lista [len(lista)]
```

```
Traceback (most recent call last):
```

```
File "<pyshell#72>", line 1, in <module>
```


```
lista [len(lista)]
```

```
IndexError: list index out of range
```

```
>>>
```


Lista Mutável -> podemos mudar valor de algum item

string – visto
anteriormente, é
IMUTÁVEL



```
>>> str = "casa"
>>> str[0]
'c'
>>> str[0]='C'
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    str[0]='C'
TypeError: 'str' object does not support item assignment
>>> |
```


list – é
MUTÁVEL



```
>>> lista = ['a', 'b', 'd', 'd']
>>> lista
['a', 'b', 'd', 'd']
>>> lista[2]='c'
>>> lista
['a', 'b', 'c', 'd']
```


loops -> as listas em processos iterativos

```
>>> #loop e as listas
>>> # pelos elementos da lista
>>> lista=["casa", "mesa", "janela", "porta"]
>>> for elem in lista:
    print(elem, end=' ')
```



casa mesa janela porta

```
>>> #loop e as listas
>>> # pelos indices da lista
>>> lista=["casa", "mesa", "janela", "porta"]
>>> for indice in range(len(lista)):
    print(lista[indice], end = ' ')
```



casa mesa janela porta

Métodos : ações a serem aplicadas em listas

Método importante:

append() -> adicionar novos item(s) no **final** da lista – a lista precisa **existir** – mesmo que vazia

Observação: não se pode ocupar uma posição que ainda não existe na lista.

Outros Métodos Úteis são apresentados nos slides finais.

Métodos:

append() -> **adicionar** novos item(s) no **final** da lista – a lista precisa existir – mesmo que vazia

Exemplo 1:

```
>>> #lista vazia
>>> lista=[]
>>> lista
[]
>>> #append() - insere no final se existir
>>> lista.append("caderno")
>>> lista
['caderno']
```

Exemplo 2:

```
>>> #lendo valor e inserindo no final
>>> lista.append(input("Material escolar: "))
Material escolar: borracha
>>> lista
['caderno', 'borracha']
```

Voltando ao Problema:

PROBLEMA – NIVEL I:

Construir um programa que faz a leitura das **notas** de **N alunos**. Calcula a **média** da classe e imprime **a lista de notas** e a média da classe.

Análise do problema:

1. Estrutura dos dados - variáveis e seus tipos:

- **N**: para a quantidade de alunos – *inteiro*
- guardar as **N notas** lidas – precisamos de uma **estrutura que guarde** um conjunto de informações → **para o exercício** vamos usar estrutura de dados **LISTA**
- **soma**: para acumular as notas lidas – *real*
- **media**: receber o resultado do cálculo da média - *real*

Representação lógica:

Exemplo: supor **N = 8** e **notas=[]** → iniciar a lista como vazia

notas	[6.5,	7,	8,	10,	5.5,	3.5,	8,	10]
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]

PROBLEMA – NIVEL I**Análise do problema - continuação:****2. Entrada de Dados:**

- **N**: número de notas
- **nota** de **N** alunos

3. Cálculos necessários:

- Somatório das **N** notas: **soma = soma + nota**
- depois de calcular a soma de **TODAS** as notas – calcular a média: **media = soma/N**

4. Sobre o fluxo do programa:

- para ler **N** notas, **precisamos** de **processo repetitivo** – por ser quantidade, usar **for**;
- para imprimir as **N** notas, **precisamos** de outro **processo repetitivo**

5. sobre os comandos dentro do processo repetitivo:

- **1º. processo repetitivo:**
 - leitura de cada **nota**;
 - **guardar** a nota na **LISTA** e,
 - efetuar a **soma** da **nota** lida, acumulando na variável **soma**;
- **2º. processo repetitivo:**
 - percorrer a **lista** e imprimir **as notas lidas**

6. resultado esperado:

- a lista de notas e a média da classe calculada

7. validações necessárias:

- o valor de **N** deve ser positivo
- a nota digitada deve estar entre **0** (zero) e **10** (dez)
(*) **no exercício não fiz a validação da nota**

PROBLEMA – NIVEL I

```
File Edit Format Run Options Window Help

''' calcula media da classe e imprime notas e média'''

''' leitura do NÚMERO de notas que serão digitadas'''
while True:
    N = int(input('Número de notas (Maior do que ZERO):'))
    if N>0:
        break
    else: print('Quantidade de Notas Inválida')

'''iniciando a variável que acumulará as somas
com ZERO - elemento neutro da soma'''
soma =0
NOTAS =[] #iniciar a lista vazia
for i in range(N):
    nota = float(input('Nota: '))
    NOTAS.append(nota)
    soma+=nota

Média = soma/N

print(f'Média da Classe: {Média:.1f}')

#percorrer a lista para imprimir as notas
print('<<< LISTA DE NOTAS >>>')

for nota in NOTAS:
    print(f'Nota: {nota:.1f}')
```

```
===== RESTART: C
Número de notas (Maior do que ZERO):4
Nota: 5
Nota: 7
Nota: 4
Nota: 3
Média da Classe: 4.8
<<< LISTA DE NOTAS >>>
Nota: 5.0
Nota: 7.0
Nota: 4.0
Nota: 3.0
>>>
```

percorrendo a lista com as notas

PROBLEMA – NIVEL II: construir um programa que faz a leitura **DO NOME** e das **notas** de **N** alunos, armazenando-as numa **lista**. Calcula a **média** da classe e **imprime: a média da classe** e o **NOME** e a **NOTA** apenas dos alunos com nota maior ou igual à média da classe

Análise do problema:

1. Como estruturar os dados? Que variáveis e que tipo de estrutura usar?

destaque:

- **N:** para a quantidade de alunos – *inteiro*
- guardar: **NOME** e **NOTA** de **N** alunos – precisamos de uma estrutura que aceita um conjunto de informações, com mesmo padrão: *nome, nota* → para o exercício vamos usar **LISTA**
- **soma:** para acumular as notas lidas – *real*
- **media:** receber o resultado do cálculo da média - *real*

PROBLEMA – NIVEL II**destaque:**

guardar: **NOME** e **NOTA** de **N** alunos – precisamos de uma estrutura que aceita um conjunto de informações, com mesmo padrão: *nome, nota* → para o exercício vamos usar **LISTA**

Vamos examinar duas formas básicas para representar:

I - **Lista simples** contendo os pares NOME e NOTA, como elementos, não agrupados:

ALUNO	[João,	7.5,	Maria,	9.5,	Mara,	9,	Sérgio,	6.5,	Flávia,	10]
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

II - Lista onde os pares: NOME e NOTA, são agrupados em sub-listas:

ALUNO	[[João,	7.5],	[Maria,	9.5],	[Mara,	9],	[Sérgio,	6.5],	[Flávia	10]]
		[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	

PROBLEMA – NIVEL II

destaque:

guardar: **NOME** e **NOTA** de **N** alunos – precisamos de uma estrutura que aceita um conjunto de informações, com mesmo padrão: *nome, nota* → para o exercício vamos usar **LISTA**

I - **Lista simples** contendo os pares NOME e NOTA, como elementos, não agrupados:

ALUNO	[João,	7.5,	Maria,	9.5,	Mara,	9,	Sérgio,	6.5,	Flávia,	10]
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

para inserir
elementos na lista:

```
for i in range(N):  
    nome = input('Nome: ')  
    nota = float(input('Nota: '))  
    ALUNO.append(nome)  
    ALUNO.append(nota)
```

para percorrer e
imprimir:

```
for i in range(0, 2*N, 2):  
    if ALUNO[i+1] >= Média:  
        print(f'Nome: {ALUNO[i]} Nota: {ALUNO[i+1]:.1f}')
```

PROBLEMA – NIVEL II

destaque:

guardar: **NOME** e **NOTA** de **N** alunos – precisamos de uma estrutura que aceita um conjunto de informações, com mesmo padrão: *nome, nota* → para o exercício vamos usar **LISTA**

II - Lista onde os pares: NOME e NOTA, são agrupados em sub-listas:

ALUNO

[[João,	7.5],	[Maria,	9.5],	[Mara,	9],	[Sérgio,	6.5],	[Flávia	10]]
		[0]			[1]			[2]			[3]			[4]		

para inserir
elementos na lista:

```
for i in range(N):  
    nome = input('Nome: ')  
    nota = float(input('Nota: '))  
    ALUNO.append([nome, nota])
```

para percorrer e
imprimir:

```
for aluno in ALUNO:  
    if aluno[1] >= Média:  
        print(f'Nome: {aluno[0]} Nota: {aluno[1]:.1f}')
```

Análise do problema(continuação)

2. Entrada de Dados:

- **N**: número de notas
- par: **Nome** e **Nota** de **N** alunos

3. Cálculos necessários:

- Somatório das **N** notas: **soma = soma + nota**
- depois de calcular a soma de **TODAS** as notas – calcular a média: **media = soma/N**

4. Sobre o fluxo do programa:

- para ler **N** pares de NOME e NOTA, **precisamos** de **processo repetitivo** – por ser quantidade, usar **for**;
- para imprimir os **N** pares de **NOME** e **NOTA**, **precisamos** de outro **processo repetitivo**

6. resultado esperado:

- a lista de notas e a média da classe calculada

7. validações necessárias:

- o valor de **N** deve ser positivo
- a nota digitada deve estar entre **0** (zero) e **10** (dez)

5. o que será repetido? o que estará dentro de cada processo repetitivo?

- **1º. processo repetitivo:**
 - leitura de cada par: **NOME** e **NOTA**;
 - **guardar** o par de dados lido na **LISTA** e,
 - efetuar a **soma** da **NOTA** lida, acumulando na variável **soma**;
- **2º. processo repetitivo:**
 - percorrer a lista, testar se a nota é maior ou igual à média da classe. Se sim, imprimir **NOME** e **NOTA** daquele aluno. Se não, não fazer nada.

listas - *list*

```
''' calcula media da classe e imprime acima ou igual a média'''
''' leitura e validação do NÚMERO de notas que serão digitadas'''
print(' <<< CALCULO DA MÉDIA E LISTA DE ALUNOS ACIMA DA MÉDIA >>>')

print('\n <<< ENTRADA DE DADOS >>>\n')

while True:
    N = int(input('Número de notas (Maior do que ZERO):'))
    if N>0:
        break
    else: print('Quantidade de Notas Inválida')

'''iniciando a variável que acumulará as somas
com ZERO - elemento neutro da soma'''
Soma =0

ALUNO =[]# criando a lista vazia

for i in range(N):
    nome = input('Nome: ')
    while True:
        nota = float(input('Nota(de 0 a 10): '))
        if 0 <= nota <= 10:
            break
        else: print('Nota Inválida')

    ALUNO.append([nome, nota])
    Soma+=nota

Média = Soma/N
print(' <<< RELATÓRIO >>>')
print(f'\n Média da Classe: {Média:.1f}\n')

for aluno in ALUNO:
    if aluno[1]>=Média:
        print(f'Nome: {aluno[0]} Nota: {aluno[1]:.1f}')
```

```
<<< CALCULO DA MÉDIA E LISTA DE ALUNOS ACIMA DA MÉDIA >>>

<<< ENTRADA DE DADOS >>>

Número de notas (Maior do que ZERO):2
Nome: Ana Maria
Nota(de 0 a 10): 8
Nome: João Antônio
Nota(de 0 a 10): 4
<<< RELATÓRIO >>>

Média da Classe: 6.0

Nome: Ana Maria Nota: 8.0
```


Exemplos do uso de alguns métodos:

insert () -> inserir novos item(s) em posições específicas

remove () -> remover item(s) da lista com valor especificado

count() -> retorna o número de itens existentes na lista com o valor especificado

extend () -> adiciona os itens de uma lista (pode ser tipo iterável) no final - concatenar duas listas

index () -> retorna o **índice** do primeiro item com o valor especificado

clear() -> remove todos os item(s) da lista deixando-a vazia

copy() -> faz uma copia dos item(s) da lista

reverse () -> inverter a ordem dos item(s) da lista

pop () -> remover item(s) da posição especificada

sort () -> ordenar os item(s) da lista

Métodos:

copy() -> faz uma cópia do(s) item(s) da lista

```
>>> #copy() - faz uma cópia da lista gerando uma segunda
>>> lista2 = lista.copy()
>>> lista
['caderno', 'borracha']
>>> lista2
['caderno', 'borracha']
>>> |
```

extend () -> adiciona os itens de uma lista no final da juntando outra - concatenar duas listas

```
>>> lista
['caderno', 'borracha']
>>> #extend() - concatenar - pode ser usado '+'
>>> lista3=['lápis', 'caderno']
>>> lista.extend(lista3)
>>> lista
['caderno', 'borracha', 'lápis', 'caderno']
>>> lista3
['lápis', 'caderno']
>>> lista = lista + ['régua']
>>> lista
['caderno', 'borracha', 'lápis', 'caderno', 'régua']
\\
```

Métodos:

count() -> retorna o número de itens existentes na lista com o valor especificado

```
>>> lista
['caderno', 'borracha', 'lápis', 'caderno', 'régua']
>>> #count() - contar ocorrências
>>> # contar número de ocorrências de 'caderno'
>>> lista.count('caderno')
2
```

index() -> retorna o índice do primeiro item com o valor especificado

```
>>> lista
['caderno', 'borracha', 'lápis', 'caderno', 'régua']
>>> #index() - posição da 1ª ocorrência
>>> lista.index("caderno")
0
```

Métodos:

sort () -> ordenar o(s) item(s) da lista

```
>>> lista
['caderno', 'borracha', 'lápis', 'caderno', 'régua']
>>> #sort() - ordena
>>> lista.sort()
>>> lista
['borracha', 'caderno', 'caderno', 'lápis', 'régua']
>>> |
```

reverse () -> inverter a ordem dos itens da lista

```
>>> lista
['borracha', 'caderno', 'caderno', 'lápis', 'régua']
>>> #reverse() - inverte
>>> lista.reverse()
>>> lista
['régua', 'lápis', 'caderno', 'caderno', 'borracha']
>>> |
```

Métodos:

insert () -> inserir novo(s) item(s) em posições específicas

```
>>> lista
['régua', 'lápis', 'caderno', 'caderno', 'borracha']
>>> #insert(i,x) - insere itens em posições específicas
>>> lista.insert(0,'apontador')
>>> lista
['apontador', 'régua', 'lápis', 'caderno', 'caderno', 'borracha']
>>> lista.insert(2,['caneta azul', 'caneta vermelha'])
>>> lista
['apontador', 'régua', ['caneta azul', 'caneta vermelha'], 'lápis', 'caderno', 'caderno', 'borracha']
>>> |
```

remove () -> remover item(s) da lista com valor especificado

```
>>> lista
['apontador', 'régua', ['caneta azul', 'caneta vermelha'], 'lápis', 'caderno', 'caderno', 'borracha']
>>> #remove() - remove um determinado valor da lista - se não existir ->ValueError
>>> lista.remove('borracha')
>>> lista
['apontador', 'régua', ['caneta azul', 'caneta vermelha'], 'lápis', 'caderno', 'caderno']
>>>
```

pop () -> remover item(s) da posição especificada

```
>>> lista
['apontador', 'régua', ['caneta azul', 'caneta vermelha'], 'lápis', 'caderno', 'caderno']
>>> #pop() - remove de uma determinada posição seu valor e o retorna - se não existir ->ValueError
>>> retirado = lista.pop(2)
>>> lista
['apontador', 'régua', 'lápis', 'caderno', 'caderno']
>>> retirado
['caneta azul', 'caneta vermelha']
>>> |
```

clear() -> remove todos os itens da lista -> vazia

```
>>> lista
['apontador', 'régua', 'lápis', 'caderno', 'caderno']
>>> retirado
['caneta azul', 'caneta vermelha']
>>> #clear - remove todos os itens de uma lista
>>> lista.clear()
>>> lista
[]
>>> |
```