

Lista de Exercícios 4 - Algoritmos e Programação
Vetores/Listas

1. Dado um n inteiro e uma sequência de n números inteiros (um por linha), imprima a sequência em ordem inversa à de leitura.
2. Dado um inteiro n que representa uma quantidade de dias e uma sequência de n valores (um por linha) representando as temperaturas desses dias, responda quais dias tiveram temperatura acima da média.
3. Uma prova consta de 6 questões, cada uma com cinco alternativas identificadas pelas letras A, B, C, D e E. Dado o cartão gabarito da prova (em uma linha) e o cartão de respostas de n estudantes (um por linha), computar o número de acertos de cada um dos estudantes.

Exemplo de entrada:

```
B A C D A E
3
A A C D B E
A A A A A A
B A C D B E
```

Exemplo de saída:

```
4
2
5
```

4. Tentando descobrir se um dado era viciado, um dono de cassino o lançou n vezes. Dados os n resultados dos lançamentos (um por linha), determinar o número de ocorrências de cada face.

Exemplo: se $n = 8$ e os resultados dos lançamentos são 6, 5, 3, 1, 6, 1, 2, 4, então a saída deve ser 2, 1, 1, 1, 1, 2.

5. Um jogador interessado em cassinos deseja fazer um levantamento estatístico simples sobre uma roleta. Para isso, ele fez n lançamentos nesta roleta. Sabendo que uma roleta contém 37 números (de 0 a 36), calcular a frequência de cada número desta roleta nos n lançamentos realizados.

Exemplo: se $n = 10$ e os resultados dos lançamentos são 32, 12, 6, 0, 9, 12, 11, 7, 28, 21, então a saída deve ser 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0.

6. Calcule o valor do polinômio $p(x) = a_0 + a_1x + \dots + a_nx^n$ em k pontos distintos. São dados os valores de n (grau do polinômio), com $1 \leq n \leq 100$, de a_0, a_1, \dots, a_n (coeficientes reais do polinômio), de k e dos pontos x_1, x_2, \dots, x_k .
7. Dada uma sequência de números inteiros em uma única linha, encontre e escreva o menor valor (**NÃO utilize a função min**).
8. Dada uma sequência de números inteiros em uma única linha, imprimi-la em ordem crescente de seus valores em uma única linha (**NÃO utilize a função min ou a função sort**).
9. Dada uma sequência de números inteiros em uma única linha, escreva essa sequência sem repetições em uma única linha.
10. Dada uma sequência de números inteiros em uma única linha, escreva os elementos repetidos em uma única linha.
11. Dizemos que uma sequência de n elementos, com n par, é *balanceada* se as seguintes somas são todas iguais:
 - a soma do maior elemento com o menor elemento;
 - a soma do segundo maior elemento com o segundo menor elemento;
 - a soma do terceiro maior elemento com o terceiro menor elemento;
 - e assim por diante ...

Exemplo:

2 12 3 6 16 15 é uma sequência balanceada,
pois $16 + 2 = 15 + 3 = 12 + 6$.

Dada uma sequência de números inteiros em uma única linha, verificar se essa sequência é balanceada. Suponha que a quantidade de números digitados sempre será **par**.

12. Dado uma sequência de números inteiros em uma única linha, determinar a quantidade de vezes que cada um deles ocorre na própria sequência.

Sequência: -1 3 0 5 0 5 -1 2 -5 0

Saída:

-1 ocorre 2 vez(es)
3 ocorre 1 vez(es)
0 ocorre 3 vez(es)
5 ocorre 2 vez(es)
2 ocorre 1 vez(es)
-5 ocorre 1 vez(es)

Você não deve escrever a saída mais de uma vez para o mesmo valor. Dica: utilize uma lista auxiliar para armazenar para quais valores você já escreveu a saída.

13. Escreva uma função com a seguinte interface:

```
def subconjunto(A, B):
```

que receba duas listas A e B , ambas representando conjuntos, devolvendo **True** se A está contido em B ($A \subset B$) e **False** caso contrário. Escreva um programa que leia dois conjuntos com números inteiros e diga se os conjuntos são iguais (obs: os elementos podem estar repetidos e em ordens diferentes).

14. Escreva uma função com a seguinte interface:

```
def remrep(lista):
```

que receba uma lista e remova elementos repetidos. Escreva um programa que leia um conjunto de números inteiros e escreva o conjunto lido sem repetições de valores.

15. Escreva uma função com a seguinte interface:

```
def intersec(A, B):
```

que dadas duas listas com números inteiros A e B que representam conjuntos, construa e devolva uma nova lista contendo a intersecção dos conjuntos formados pelos elementos de A e B ($A \cap B$). Escreva um programa que leia dois conjuntos com números inteiros e escreva a intersecção dos dois conjuntos.

16. Uma sequência de $n > 0$ números inteiros é chamada de **cheia** se os valores absolutos das diferenças entre os elementos consecutivos representam todos os possíveis valores entre 1 e $n - 1$.

Exemplo: 1 4 2 3 é uma sequência cheia, porque os valores absolutos das diferenças entre seus elementos consecutivos são 3, 2 e 1, respectivamente.

Observe que esta definição implica que qualquer sequência contendo exatamente um número inteiro é uma sequência cheia.

Escreva uma função com a seguinte interface:

```
def uns(contadores):
```

que receba uma lista de números inteiros chamada *contadores*, devolvendo *True* se todos elementos da lista são iguais a 1 ou *False* caso contrário. Escreva um programa que leia uma sequência de números inteiros e diga se a sequência é cheia ou não.

17. O **crivo de Eratóstenes** é um método para encontrar números primos até um certo valor limite. Segundo a tradição, foi criado pelo matemático grego Eratóstenes, o terceiro bibliotecário-chefe da Biblioteca de Alexandria. A ideia do método é começar com todos os inteiros no intervalo $[2, n]$ e eliminar, em cada iteração, os múltiplos próprios de um número primo considerado. O primeiro número primo a ser considerado é 2 e o último é $\lfloor \sqrt{n} \rfloor$. Dados dois números inteiros a e b , dizemos que a é **múltiplo próprio** de b se a é múltiplo de b e $a > b$.

Exemplo:

Se $n = 25$, consideramos a lista C com os $n - 1$ elementos a seguir

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25.

Eliminando os múltiplos próprios de 2 restam

2, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25 .

Eliminando os múltiplos próprios de 3 restam

2, 3, 5, 7, 11, 13, 17, 19, 23, 25 .

Eliminando os múltiplos próprios de 5 restam

2, 3, 5, 7, 11, 13, 17, 19, 23 .

Não há necessidade de verificar os múltiplos próprios de 7, 11, 13, 17, 19 e 23, já que $\lfloor \sqrt{25} \rfloor = 5$.

Escreva uma função com a seguinte interface:

```
def elimina(C, i):
```

que receba uma lista C com números inteiros e um índice i , e então elimine desta lista os múltiplos próprios do valor em $C[i]$ a partir da posição seguinte a esse valor, ou seja, a partir da posição $i + 1$ de C . Lembre-se que para remover o elemento na posição j de uma lista C podemos utilizar `del C[j]`. Escreva um programa que receba um número inteiro n , com $2 \leq n \leq 1000$, e imprima os números primos de 2 a n .

18. [BEECROWD - nº 1174] Faça um programa que leia um vetor $A[100]$. No final, mostre todas as posições do vetor que armazenam um valor menor ou igual a 10 e o valor armazenado em cada uma das posições.

A entrada contém 100 valores, podendo ser inteiros, reais, positivos ou negativos.

Para cada valor do vetor menor ou igual a 10, escreva " $A[i] = x$ ", onde i é a posição do vetor e x é o valor armazenado na posição, com uma casa após o ponto decimal.

Entrada:

0

-5

63

-8.5

...

Saída:

A[0] = 0.0

A[1] = -5.0

A[3] = -8.5

...

19. [BEECROWD - nº 1175] Faça um programa que leia um vetor N[20]. Troque a seguir, o primeiro elemento com o último, o segundo elemento com o penúltimo, etc., até trocar o 10º com o 11º. Mostre o vetor modificado.

A entrada contém 20 valores inteiros, positivos ou negativos.

Para cada posição do vetor N, escreva "N[i] = Y", onde i é a posição do vetor e Y é o valor armazenado naquela posição.

Entrada:

0

-5

...

63

230

Saída:

N[0] = 230

N[1] = 63

...

N[18] = -5

N[19] = 0

20. [BEECROWD - nº 1177] Faça um programa que leia um valor T e preencha um vetor N[1000] com a sequência de valores de 0 até T-1 repetidas vezes, conforme exemplo abaixo. Imprima o vetor N.

A entrada contém um valor inteiro T ($2 \leq T \leq 50$).

Para cada posição do vetor, escreva "N[i] = x", onde i é a posição do vetor e x é o valor armazenado naquela posição.

Entrada:

3

Saída:

N[0] = 0

N[1] = 1

N[2] = 2

N[3] = 0

N[4] = 1

N[5] = 2

N[6] = 0

N[7] = 1

N[8] = 2

...

21. [BEECROWD - nº 1178] Leia um valor X. Coloque este valor na primeira posição de um vetor N[100]. Em cada posição subsequente de N (1 até 99), coloque a metade do valor armazenado na posição anterior, conforme o exemplo abaixo. Imprima o vetor N.

A entrada contém um valor de dupla precisão com 4 casas decimais.

Para cada posição do vetor N, escreva "N[i] = Y", onde i é a posição do vetor e Y é o valor armazenado naquela posição. Cada valor do vetor deve ser apresentado com 4 casas decimais.

Entrada:

200.0000

Saída:

N[0] = 200.0000

N[1] = 100.0000

N[2] = 50.0000

N[3] = 25.0000

N[4] = 12.5000

...

22. [BEECROWD - nº 1180] Faça um programa que leia um valor N. Este N será o tamanho de um vetor X[N]. A seguir, leia cada um dos valores de X, encontre o menor elemento deste vetor e a sua posição dentro do vetor, mostrando esta informação.

A primeira linha de entrada contém um único inteiro N ($1 < N < 1000$), indicando o número de elementos que deverão ser lidos em seguida para o vetor X[N] de inteiros. A segunda linha contém cada um dos N valores, separados por um espaço. Vale lembrar que nenhuma entrada haverá números repetidos.

A primeira linha apresenta a mensagem “Menor valor:” seguida de um espaço e do menor valor lido na entrada. A segunda linha apresenta a mensagem “Posicao:” seguido de um espaço e da posição do vetor na qual se encontra o menor valor lido, lembrando que o vetor inicia na posição zero.

Entrada:

10

1 2 3 4 -5 6 7 8 9 10

Saída:

Menor valor: -5

Posicao: 4

23. O desvio padrão é uma medida que expressa o grau de dispersão de um conjunto de dados. Ou seja, o desvio padrão indica o quanto um conjunto de dados é uniforme. Quanto mais próximo de 0 for o desvio padrão, mais homogêneo são os dados.

Seu cálculo é dado pela fórmula:

$$DP = \sqrt{\frac{\sum_{i=1}^n (x_i - M_A)^2}{n}}$$

Onde:

A letra grega sigma maiúscula que representa o somatório. Indica que temos que somar todos os termos, desde a primeira posição (i=1) até a posição n

x_i : valor na posição i no conjunto de dados

MA: média aritmética dos dados

n: quantidade de dados

Exemplo:

Considere os dados de valores 1,70, 1,55 e 1,80.

Primeiro, calculamos a média:

$$1,70 + 1,80 + 1,55 = 5,05$$

$$5,05/3 = \text{aproximadamente } 1,68$$

Agora calculamos o desvio padrão:

$$DP = \sqrt{\frac{(1,55-1,68)^2 + (1,70-1,68)^2 + (1,80-1,68)^2}{3}}$$

$$DP = \sqrt{\frac{(0,13)^2 + (0,02)^2 + (0,12)^2}{3}} = \sqrt{\frac{0,0317}{3}}$$

$$DP = \sqrt{0,0105} = 0,1027$$

Logo, o desvio padrão é aproximadamente 0,10

Exercício: Crie uma função `desvio_padrao(dp,med)` para calcular o desvio padrão, sendo `dp` uma lista com os dados e `med` a média aritmética entre esses dados e retorne o desvio padrão (a função deve retornar um número com duas casas após o ponto decimal).

Crie um programa que leia `n`, sendo `n` a quantidade de dados que serão inseridos na lista, e o valor de todos os dados e imprima o valor do desvio padrão.

OBS: Utilize a biblioteca **math** e a função **math.sqrt()** para calcular a raiz quadrada dentro da função `desvio_padrao(dp,med)`

Entrada:

5
1
2
3
4
5

Saída:

1,41

24. Faça um programa que possua um inteiro `n` que é a quantidade de casos gerados, e `n` quantidades de strings e inteiros (uma string e um inteiro por linha), que representa o nome e a idade de pessoas, respectivamente. Liste os nomes dentro da lista `nomes[]` e as idades dentro da lista `idades[]`

Imprima a média (com uma casa após o ponto decimal) entre as idades, e diga quem está com a idade abaixo, igual ou acima da média.

Entrada:

5

João 35

Paulo 57

Moisés 14

Giovana 29

Solange 73

Saída:

Média: 41,6

João está abaixo da média

Paulo está acima da média

Moisés está abaixo da média

Giovana está acima da média

Solange está acima da média

25. [MEMÓRIA] Considere o código abaixo:

```
1. n = int(input())
2. v = []

3. v.append(10)
4. v.append(15)

5. i = 0
6. while i < n:
7.     v.append(float(input()))
8.     i += 1
```

Considere também o seguinte contexto:

- Cada posição de memória ocupa 1 byte;
- O processador em questão utiliza endereços de memória de 16 bits (2 bytes);
- Um valor inteiro ocupa 32 bits (4 bytes);
- Um valor float ocupa 64 bits (8 bytes).
- Todas as regiões de memória podem ser utilizadas, exceto as posições destacadas em vermelho;
- Qualquer vetor no Python recebe, inicialmente, 8 bytes de memória para armazenar valores. Assuma que o sistema operacional decidiu alocar o vetor, inicialmente, na posição 8 de memória;
- Escolha a região de memória de sua preferência para alocar a variável *n* na memória;

- Você pode realocar as variáveis do seu programa na memória sempre que houver necessidade de mais espaço.

Utilize a tabela disponibilizada neste link para representar qual será o estado da memória em dois momentos diferentes da execução:

- Logo após a execução da linha 4;
- Logo após a execução da linha 8.

26. [BEECROWD - nº 1181] Neste problema você deve ler um número, indicando uma linha da matriz na qual uma operação deve ser realizada, um caractere maiúsculo, indicando a operação que será realizada, e todos os elementos de uma matriz $M[12][12]$. Em seguida, calcule e mostre a soma ou a média dos elementos que estão na área verde da matriz, conforme for o caso. A imagem abaixo ilustra o caso da entrada do valor 2 para a linha da matriz, demonstrando os elementos que deverão ser considerados na operação

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

A primeira linha de entrada contém um número L ($0 \leq L \leq 11$) indicando a linha que será considerada para operação. A segunda linha de entrada contém um único caractere Maiúsculo T ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem os 144 valores de ponto flutuante que compõem a matriz, sendo que a mesma é preenchida linha por linha, da linha 0 até a linha 11, sempre da esquerda para a direita.

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Entrada:

2
S
0.0
-3.5
2.5
4.1
...

Saída:

12.6

27. [BEECROWD - nº 1183] Leia um caractere maiúsculo, que indica uma operação que deve ser realizada e uma matriz $M[12][12]$. Em seguida, calcule e mostre a soma ou a média considerando somente aqueles elementos que estão acima da diagonal principal da matriz, conforme ilustrado abaixo (área verde).

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

A primeira linha de entrada contém um único caractere Maiúsculo O ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem os 144 valores de ponto flutuante que compõem a matriz.

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Entrada:

S

1.0
0.0
-3.5
2.5
4.1
...

Saída:

12.6

28. [BEECROWD - nº 1184] Leia um caractere maiúsculo, que indica uma operação que deve ser realizada e uma matriz $M[12][12]$. Em seguida, calcule e mostre a soma ou a média considerando somente aqueles elementos que estão abaixo da diagonal principal da matriz, conforme ilustrado abaixo (área verde).

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

A primeira linha de entrada contém um único caractere Maiúsculo O ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem os 144 valores de ponto flutuante que compõem a matriz.

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Entrada:

S
1.0

0.0

-3.5

2.5

4.1

...

Saída:

12.6

29. [BEECROWD - nº 1185] Leia um caractere maiúsculo, que indica uma operação que deve ser realizada e uma matriz $M[12][12]$. Em seguida, calcule e mostre a soma ou a média considerando somente aqueles elementos que estão acima da diagonal secundária da matriz, conforme ilustrado abaixo (área verde).

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

A primeira linha de entrada contém um único caractere Maiúsculo O ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem os 144 valores de ponto flutuante que compõem a matriz.

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Entrada:

S

1.0

0.0

-3.5

2.5

4.1

...

Saída:

12.6

30. [BEECROWD - nº 1187] Leia um caractere maiúsculo, que indica uma operação que deve ser realizada e uma matriz $M[12][12]$. Em seguida, calcule e mostre a soma ou a média considerando somente aqueles elementos que estão na área superior da matriz, conforme ilustrado abaixo (área verde).

	0	1	2	3	4	5	6	7	8	9	10	11
0												
1												
2												
3												
4												
5												
6												
7												
8												
9												
10												
11												

A primeira linha de entrada contém um único caractere Maiúsculo O ('S' ou 'M'), indicando a operação (Soma ou Média) que deverá ser realizada com os elementos da matriz. Seguem 144 valores com ponto flutuante de dupla precisão que compõem a matriz.

Imprima o resultado solicitado (a soma ou média), com 1 casa após o ponto decimal.

Entrada:

S

1.0

330.0

-3.5

2.5

4.1

...

Saída:

12.4