Lista de Exercícios 01 - LPOO

01 - Defina com suas palavras o processo de resolução de problemas (Princípios de Pólya).

O método de Polya consiste em três etapas: Compreender o problema, Designar um plano, Executar o plano e Retrospecto do problema.

Referência

02 - Quais são os tipos de erros possíveis no processo de criação dos Algoritmos?

- 01 - Não declarar variáveis utilizadas dentro do programa
- 02 - Utilizar uma variável de um modo incompatível com o seu tipo declarado
- 03 - Fazer comparações com uma variável do tipo CHAR e um caractere e não colocar este caractere entre aspas simples
- 04 - Usar atribuições no lugar de comparações ou vice-versa
- 05 - Operar variáveis tipo CHAR com números
- 06 - Certifique-se que você não está usando uma palavra reservada do Python (Python keyword) como nome de variável
- 07 - Verifique que você colocou um dois-pontos `:` no final do cabeçalho de cada comando composto (compound statement), incluindo laço `for`, laço `while`, comandos de seleção `if`, `elif` e `else` e declaração de funções `def`
- 08 - Verifique se a tabulação é consistente. Você pode tabular com espaços ou tabs mas não é aconselhável misturá-los, Cada nível deve ter a mesma quantidade de espaços ou tabs.
- 09 - Laços infinitos
- 10 - Recursão infinita

Referência

Referência

03 - Escreva três formas de representação de um algoritmo e explique-os.

Os três tipos mais utilizados de algoritmos são a descrição narrativa, o fluxograma e o pseudocódigo ou portugol.

- Descrição Narrativa: consiste em analisar o enunciado do problema e escrever, utilizando uma linguagem natural (por exemplo, a língua portuguesa), os passos que devem ser seguidos para a resolução do problema.
- Fluxograma: consiste em analisar o enunciado do problema e escrever, utilizando símbolos gráficos predefinidos, os passos que devem ser seguidos para a resolução do problema.
- Pseudocódigo ou Portugol: consiste em analisar o enunciado do problema e escrever, utilizando regras predefinidas, os passos a serem seguidos para a resolução do problema.

Referência

04 - Em Python, a codificação é interpretada, possibilitando a compreensão das instruções pela máquina. A interpretação de código é diferente da compilação. Quais as diferenças

entre Compilação e Interpretação? Demonstre.

Ao utilizar o método de compilação, o compilador irá ler o código, fazer todas as análises sintáticas e demais processos, para, por fim, gerar um arquivo código-objeto ou um arquivo executável. Já na interpretação, nenhum arquivo ou código é gerado, e sim uma tradução instantânea, em tempo de execução.

Referência

05 - Quais são os benefícios da utilização da Integrated Development Enviroment (IDE)?

Simplifica os processos e unifica diversas ferramentas a uma única interface.

Dessa maneira, as aplicações são criadas de forma mais ágil, já que a IDE é eficiente ao analisar o código que será escrito, apontando falhas e erros de digitação, por exemplo. Entre as principais funcionalidades da plataforma, podemos citar:

- Editor de código-fonte: aplicado na escrita de comandos, que seguem sua devida linguagem de programação;
- Automação de compilação local: função que automatiza tarefas repetitivas em um desenvolvimento, como compilação de código binário, execução de testes e criação de pacotes de códigos;
- Preenchimento inteligente: uma das funções mais interessantes do IDE, traz o autopreenchimento de trechos de código, tornando o desenvolvimento mais rápido;
- Debbuger: realiza o teste de diversos programas, exibindo graficamente a posição do bug no código,
- Refatoração: essa função trabalha continuamente na otimização do código-fonte, aplicando testes automáticos para a eliminação completa de erros.

Referência

06 - Disserte sobre a importância da sequência dos comandos em Python, a endentação e comentários.

Em Python, os comandos/instruções em um programa são executados um após o outro na sequência escrita (de cima para baixo) do início ao fim.

A indentação é uma característica importante no Python, pois além de promover a legibilidade é essencial para o bom funcionamento do código. Em outras palavras, se a indentação não estiver adequada o programa pode se comportar de forma inesperada ou até mesmo não compilar.

Por exemplo, no caso de um if, o que determina se um código está dentro da condicional é o fato dele ter sido indentado. O Código 7 apresenta um exemplo onde o comando print só será executado se o valor da variável x for maior que 8.

Referência

Referência

07 - Quais são os tipos de dados disponíveis em Python? Como ocorre a declaração e a atribuição das variáveis e seus tipos?

- Tipo Inteiro (int): `x = 1`, `print(type(x))`

- Ponto Flutuante ou Decimal (float): `y = 3.1415`, `print(type(y))`
- Complexo (complex): `z = 5+2j`, `print(type(z))`
- String (str): `name = 'John Doe'`, `print(type(name))`
- Boolean (bool): `is_active = True`, `print(type(is_active))`
- Listas (list): `names = ['John Doe', 'Mary Doe']`, `print(type(names))`
- Tuplas (tuple): `ages = (42, 38)`, `print(type(ages))`
- Dicionários (dict): `people_age = {'John': 42, 'Mary': 38}`, `print(type(people_age))`

Referência

08 - O que significa Tipagem Dinâmica e Tipagem Forte em Python? Disserte.

Python é uma linguagem de tipagem forte e dinâmica. Na tipagem dinâmica, você pode alterar o tipo de uma variável durante a execução do código e o Python não lançará nenhuma `Exception`. Linguagens de tipagem fraca, permite que você faça operações algumas operações, sem a necessidade da realização do cast. Como por exemplo:

```
name = '11'

print(1 + name)
```

Isso não é o caso do Python, ao realizar isso. Ele retorna o seguinte traceback:

```
Traceback (most recent call last):
  File "type_error.py", line 3, in <module>
      print(1 + name)
      TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Referência

09 - Quais são as regras para declarações de variáveis em Python? Descreva.

- Os nomes das variáveis devem começar com uma letra ou um underline
- O restante do nome da variável pode consistir em letras, números e underlines
- Os nomes diferenciam maiúsculas de minúsculas

Referência

10 - Como ocorre a entrada de dados em Python? Demonstre através de exemplos.

Em Python, fazemos isso utilizando a função `input()`, que é literalmente 'entrada' em inglês.

A função `input()` recebe como parâmetro uma `String` que será mostrada como auxílio ao usuário, geralmente o informando que tipo de dado o programa está aguardando receber.

```
input("Escreva seu nome: ")
```

Referência

11 - Efetue a alteração do tipo de uma variável inteiro para o tipo decimal. Demonstre através de algoritmos e apresente a saída da operação.

```
x = 2
print('x é do tipo', type(x))
x = float(x)
print('x é do tipo', type(x))
```

12 - Construa um algoritmo em Python que solicite uma entrada e apresente o valor inserido.

```
valor = input('Digete alguma coisa:\n')
print(f'Você digitou => {valor}')
```

13 - Faça um programa que leia o nome, a idade, a altura, o peso e a nacionalidade do usuário e escreva essas informações na forma de um parágrafo de apresentação

```
print("Olá humano 🖐\n")

while True:
    try:
        input_nome = input("Qual o seu nome?\n✏️  ")
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_idade = int(input("Qual a sua idade?\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_altura = float(input("Qual a sua altura?\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_peso = float(input("Qual o seu peso?\n✏️  "))
        break
```

```
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"Olá {input_nome}, você tem {input_idade} anos, sua altura é
{input_altura}, seu peso é {input_peso} e seu IMC é {input_peso /
(input_altura * input_altura)}")
```

14 - Faça um programa que exiba o perímetro de uma circunferência a partir do seu raio

```
import math

print("Olá humano 🖐\n")

while True:
    try:
        input_raio = float(input("Digite o raio da circuferência que direi
o seu perímetro:\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O perímetro da circuferência é {2 * math.pi * input_raio}")
```

15 - Faça um programa que leia dois pontos num espaço bidimensional e calcule a distância entre esses pontos

```
import math

print("Olá humano 🖐\n")

while True:
    try:
        input_xa = float(input("Digite a coordenada x para o ponto A:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_ya = float(input("Digite a coordenada y para o ponto A:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_xb = float(input("Digite a coordenada x para o ponto B:\n✏️
```

```
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_yb = float(input("Digite a coordenada y para o ponto B:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f'A distância entre os pontos A e B é {math.sqrt((xb - xa) ** 2 + (yb
- ya) ** 2)}')
```

16 - Faça um programa para, a partir de um valor informado em centavos, indicar a menor quantidade de moedas que representa esse valor. Considere moedas de 1, 5, 10, 25 e 50 centavos, e 1 real. Exemplo: para o valor 290 centavos, a menor quantidade de moedas é 2 moedas de 1 real, 1 moeda de 50 centavos, 1 moeda de 25 centavos, 1 moeda de 10 centavos e 1 moeda de 5 centavos

```
print("Olá humano 🖐\n")

while True:
    try:
        input_valor = float(input("Digite um valor que direi a quantidade
mínima de moedas (1 real, 50 centavos, 25 centavos, 10 centavos, 5 centavos
e 1 centavo) representa esse valor:\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

moedas_utilizadas = {1: 0, 0.5: 0, 0.25: 0, 0.1: 0, 0.05: 0, 0.01: 0}

def minimo_moedas(valor):
    moedas = list(moedas_utilizadas.keys())
    quantidade = 0
    for moeda in moedas:
        while valor >= moeda:
            moedas_utilizadas[moeda] += 1
            valor -= moeda
            quantidade += 1
    return quantidade

print(f"A quantidade mínima de moedas que representa o valor {input_valor}
é {minimo_moedas(input_valor)}")

print("\nMoedas utilizadas:\n")

for moeda, quantidade in moedas_utilizadas.items():
```

```
    if quantidade > 0:
        print(f"{quantidade} moeda(s) de {moeda} {'centavo(s)' if moeda < 1
else 'real'}")
```

17 - Descreva a funcionalidade de cada um dos operadores aritméticos abaixo e dê exemplos:

- `(x)`: Parênteses têm a mais alta precedência e podem ser usados para forçar uma expressão a ser avaliada na ordem que você quiser.
- `**`: Exponenciação => Retorna o resultado da elevação da potência pelo outro.
- `+=`: Equivalente a => `x = x + 1`
- `-x`: Equivalente a => `x = x - 1`
- `*`: Multiplicação => Realiza a multiplicação de ambos operandos.
- `/`: Divisão => Realiza a Divisão de ambos operandos.
- `//`: Divisão inteira => Realiza a divisão entre operandos e a parte decimal de ambos operandos.
- `%`: Módulo => Retorna o resto da divisão de ambos operandos.
- `+`: Adição => Realiza a soma de ambos operandos.
- `-`: Subtração => Realiza a subtração de ambos operandos.

Referência

Referência

18 - Resolva

```
x = 320
y = 7.1 - (x // 5 - 22 / 3) + 21 * 0.8 + 5

print(f"O valor de y é {y}") # output: O valor de y é -27.766666666666666
```

19 - Escreva um algoritmo que calcule a distância entre dois pontos em um plano cartesiano. Apresente as variáveis em um teste de mesa. Dica: utilize funções para realizar o algoritmo.

```
import math


print("Olá humano 🖐\n")

while True:
    try:
        input_xa = float(input("Digite a coordenada x para o ponto A:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
```

```
    try:
        input_ya = float(input("Digite a coordenada y para o ponto A:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_xb = float(input("Digite a coordenada x para o ponto B:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_yb = float(input("Digite a coordenada y para o ponto B:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

def distancia(xa, ya, xb, yb):
    return math.sqrt((xb - xa) ** 2 + (yb - ya) ** 2)

print(f'A distância entre os pontos A e B é {distancia(input_xa, input_ya,
input_xb, input_yb)}')
```

20 - Escreva um algoritmo que realize o sorteio de um número dentro de um intervalo inserido pelo usuário.

```
import random

print("Olá humano 🖐\nVou sortear um número no intervalo que informar")

while True:
    try:
        input_ini = int(input("Digite o valor inicial do intervalo :\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_fim = int(input("Digite o valor final do intervalo :\n✏️  "))
        if input_fim < input_ini:
            raise ValueError("O valor final deve ser maior que o inicial")
        break
    except ValueError as e:
```

```
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"O número sorteado foi {random.randint(input_ini, input_fim)}")
```

21 - Quais são os operadores relacionais em Python? Apresente-os e utilize exemplos demonstrado o valor resultado da operação relacional.

- `==`: Igual a => Verifica se um valor é igual ao outro.
- `!=`: Diferente de => Verifica se um valor é diferente ao outro.
- `>`: Maior que => Verifica se um valor é maior que outro
- `>=`: Maior ou igual => Verifica se um valor é maior ou igual ao outro.
- `<`: Menor que => Verifica se um valor é menor que outro.
- `<=`: Menor ou igual => Verifica se um valor é menor ou igual ao outro.

Referência

22 - Quais são os operadores lógicos em Python? Descreva-os e apresente exemplos.

- `and`: Retorna True se ambas as afirmações forem verdadeiras.
- `or`: Retorna True se uma das afirmações for verdadeira.
- `not`: Retorna Falso se o resultado for verdadeiro

Referência

```
num1 = 7
num2 = 4

# Exemplo and
if num1 > 3 and num2 < 8:
    print("As Duas condições são verdadeiras")

# Exemplo or
if num1 > 4 or num2 <= 8:
    print("Uma ou duas das condições são verdadeiras")

# Exemplo not
if not (num1 < 30 and num2 < 8):
    print('Inverte o resultado da condição entre os parânteses')
```

23 - Qual o valor atribuído a variável w?

```
x = 18
y = -15
z = 1
w = x * y < z / x or x / y > z * x and z * y < x

print(f" O valor de w é {w}") # output: O valor de w é True
```

24 - O que significa `if`, `else` e `elif` em Python?

São as chamadas Estruturas Condicionais:

- `if`: Executa um bloco de código se sua condição for atendida.
- `else`: Caso uma condição não for atendida, o Python executará o bloco de código definido abaixo dessa cláusula.
- `elif`: É utilizando quando mais de uma condição `if` precisa ser testada.

Referência

25 - Faça um programa que verifique se um número é ímpar.

```
print("Olá humano 🖐\n")

while True:
    try:
        input_valor = int(input("Digite um número inteiro e direi se é
ímpar:\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O número {input_valor} {'é ímpar' if input_valor % 2 else 'não é
ímpar'}")
```

26 - Faça um programa que receba 4 valores e retorne o menor entre eles.

```
print("Olá humano 🖐\n")

while True:
    try:
        input_value = input("Digite uma lista de números inteiros separados
por ',' e no final direi qual deles é o menor:\n✏️  ")
        values_splited = input_value.split(',')
        values_int = [int(num) for num in values_splited]
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O menor número da lista é {min(values_int)}")
```

27 - Faça um algoritmo que leia a idade de uma pessoa expressa em dias e mostre-a expressa em anos, meses e dias.

```
from datetime import datetime, timedelta

print("Olá humano 🖐\n")

while True:
    try:
        input_dias = int(input("Digite quantos dias de vida você tem:\n✏️
"))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")


data_nascimento = datetime.today() - timedelta(days=input_dias)

print(f"Data de nascimento: {data_nascimento.strftime('%d/%m/%Y')}")
```

28 - Faça um programa que receba um valor que é o valor pago, um segundo valor que é o preço do produto e retorne o troco a ser dado.

```
print("Olá humano 🖐\n")

while True:
    try:
        input_valor_pago = float(input("Digite o valor pago:\n✏️  "))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

while True:
    try:
        input_valor_produto = float(input("Digite o valor do produto:\n✏️
"))
        if input_valor_pago < input_valor_produto:
            raise ValueError("O valor pago deve ser igual ou maior que o
valor do produto.")
        break
    except ValueError as e:
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"Valor do troco: {input_valor_pago - input_valor_produto:.2f}")
```

29 - Elaborar um algoritmo que lê 3 valores a, b, c e verifica se eles formam ou não um triângulo. Vamos supor que os valores lidos são inteiros e positivos. Caso os valores formem um triângulo, informar se o triângulo é:

- Equilátero: possui os três lados com medidas iguais.

- Isósceles: possui dois lados com medidas iguais.
- Escaleno: possui os três lados com medidas diferentes.

Lembre-se que para formar um triângulo:

- Nenhum dos lados pode ser igual a zero;
- Um lado não pode ser maior do que a soma dos outros dois;

```
print("Olá humano 🖐\n")

while True:
    try:
        input_a = int(input("Digite o valor do lado a do de um
triângulo:\n✏️  "))
        if input_a < 1:
            raise ValueError("O valor deve ser positivo.")
        break
    except ValueError as e:
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

while True:
    try:
        input_b = int(input("Digite o valor do lado b do de um
triângulo:\n✏️  "))
        if input_b < 1:
            raise ValueError("O valor deve ser positivo.")
        break
    except ValueError as e:
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

while True:
    try:
        input_c = int(input("Digite o valor do lado c do de um
triângulo:\n✏️  "))
        if input_c < 1:
            raise ValueError("O valor deve ser positivo.")
        if input_c > input_a + input_b:
            raise ValueError("O valor de c deve ser menor ou igual a soma
dos valores de a e b.")
        break
    except ValueError as e:
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

lados = set([input_a, input_b, input_c])

print(f"Os lados do triângulo são: {input_a}, {input_b} e {input_c}\n")

if len(lados) == 1:
    print("Triângulo equilátero")
```

```
elif len(lados) == 2:
    print("Triângulo isósceles")
else:
    print("Triângulo escaleno")
```

30 - Faça um programa em Python que peça ao usuário para digitar um texto e informe quantos caracteres possui o texto informado pelo usuário.

```
print("Olá humano 🖐\n")

while True:
    try:
        input_texto = input("Digite alguma coisa e direi quantos caracteres
há:\n✏️  ")
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O texto digitado tem {len(input_texto)} caracteres")
```

31 - Construa um programa em Python que peça ao usuário para digitar um texto em letras maiúsculas e mostre o texto em letra minúscula, em seguida solicite-o um texto em letra minúscula e mostre-o em letra maiúsculas.

```
print("Olá humano 🖐\n")

while True:
    try:
        input_texto = input("Digite um texto em maiscúlo e vou deixá-lo em
minúsculo:\n✏️  ")
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O texto digitado em minúsculo é: {input_texto.lower()}")

while True:
    try:
        input_texto = input("Digite um texto em minúsculo e vou deixá-lo em
maiscúlo:\n✏️  ")
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"O texto digitado em maiscúlo é: {input_texto.upper()}")
```

32 - Construa um programa que solicite uma frase escrita pelo usuário. Peça ao usuário para escolhe ruma palavra da frase escrita e substituí-la por outra palavra.

```
print("Olá humano 🖐\n")

words = []
frase = ''

while True:
    try:
        input_value = input("Digite uma frase:\n✏️  ")
        frase = input_value
        values_splited = input_value.split(' ')
        for word in values_splited:
            words.append(''.join(chr for chr in word if chr.isalnum()))
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"As palavras digitadas são:\n-------------------------")
for (index, word) in enumerate(words):
    print(f"{index + 1} - {word}")

while True:
    try:
        input_idx = int(input("Digite o índice da palavra que deseja
substituir:\n✏️  "))
        index = input_idx - 1
        if index < 0 or index + 1 > len(words):
            raise ValueError("Índice inválido.")
        break
    except ValueError as e:
        print("\n🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

while True:
    try:
        input_new_word = input(f"Digite a nova palavra que irá substituir
{words[index]}:\n✏️  ")
        break
    except ValueError:
        print("🚨 Valor inválido. Tente novamente!\n")

print(f"A nova frase agora é:\n---------------------")
print(f"{frase.replace(words[index], input_new_word)}")
```

33 - Faça um programa que converta os valores do tipo inteiro, para uma única String. Depois, substitua os valores referentes para a letra correspondente e mostre a palavra gerada.

- Substituir: 1 = a, 3 = c, 4 = d, 12 = m, 14 = o, 15 = p, 17 = r, 19 = t, 20 = u
- Código da palavra: 3 – 14 – 12 – 15 – 20 – 19 – 1 – 4 – 14 – 17

```
print("Olá humano 🖐\n")

alfabeto = [chr for chr in "abcdefghijklmnopqrstuvwxyz"]
codes = []

while True:
    try:
        input_value = input("Digite o código da frase com números de 1 a 26
separados por '-':\n✏️  ")
        input_split = input_value.split('-')
        for code in input_split:
            if int(code) < 1 or int(code) > 26:
                raise ValueError("Código inválido.")
            else:
                codes.append(int(code))
        break
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"A frase decodificada é:\n----------------------")
print(''.join(alfabeto[code - 1] for code in codes))
print(f"As opções de codificação são:\n--------------------------------")
for (i, letra) in enumerate(alfabeto):
    print(f"{i+1} - {letra}")
```

34 - Escreva um programa que dado um valor numérico digitado pelo usuário (armazenado em uma variável inteira), imprima cada um dos seus dígitos por extenso.

```
print("Olá humano 🖐\n")

numeros = ["zero", "um", "dois", "três", "quatro", "cinco", "seis", "sete",
"oito", "nove"]

while True:
    try:
        input_value = input("Digite digite um número inteiro:\n✏️  ")
        for i in input_value:
            num = int(i)
            if num < 0:
                raise ValueError("Número inválido.")
            numeros.append(num)
        break
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")
```

```
print(f"Os algarismos do número digitado são: {', '.join([numeros[int(i)]
for i in str(input_value)])}")
```

35 - Escreva um algoritmo que solicite ao usuário a entrada de 5 números, e que exiba o somatório desses números na tela. Após exibir a soma, o programa deve mostrar também os números que o usuário digitou, um por linha.

```
print("Olá humano 🖐\n")

print("Digite cinco números:\n")
numeros = []
while len(numeros) < 5:
    try:
        input_value = float(input(f"{len(numeros) + 1}° número:\n✏️  "))
        numeros.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"Os números digitados foram:\n{', '.join([str(i) for i in
numeros])}\n--------------------------")
print(f"A soma dele é: {sum(numeros)}")
```

36 - Crie um programa que solicite a entrada de 10 números pelo usuário, armazenando-os em um vetor, e então monte outro vetor com os valores do primeiro multiplicados por 5. Exiba os valores dos dois vetores na tela, simultaneamente, em duas colunas (um em cada coluna), uma posição por linha.

```
print("Olá humano 🖐\n")

print("Digite dez números:\n")
numeros = []
while len(numeros) < 10:
    try:
        input_value = float(input(f"{len(numeros) + 1}° número:\n✏️  "))
        numeros.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"Os números digitados e sua multiplicação por 5 são:\n----------
----------")
for i in numeros:
    print(f"{i} x 5 = {i * 5}")
```

37 - Modifique o programa anterior para não aceitar a entrada do número zero, e requisitar a digitação de outro número neste caso.

```
print("Olá humano 🖐\n")

print("Digite dez números:\n")
numeros = []
while len(numeros) < 10:
    try:
        input_value = float(input(f"{len(numeros) + 1}° número:\n✏️  "))
        if input_value == 0:
            raise ValueError("Valor zero não é permitido!")
        numeros.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"Os números digitados e sua multiplicação por 5 são:\n--------------
------------")
for i in numeros:
    print(f"{i} x 5 = {i * 5}")
```

38 - Modifique novamente o programa anterior, de modo a não exibir na saída os números zero que são mostrados para todas as posições que não receberam nenhum valor durante a atribuição (e portanto estão vazias).

Não entendi. 🤔

39 - Escreva um algoritmo que lê um vetor A(10) e os escreva, imprima a posição e o elemento do vetor.

```
print("Olá humano 🖐\n")

elementos = [1,2,3,'Mas o que temos aqui', 3.1415, ['Isso é outro vetor'],
('Aqui temos uma tupla'), 8, 9, 10]

for (index, elemento) in enumerate(elementos):
    print(f'''O elemento "{elemento}" do tipo {type(index)}, está na
posição {index}''')
```

40 - Leia números de matrículas de alunos e armazene-os em uma tupla até o vetor ser preenchido por 10 matrículas. Esses números são distintos, ou seja, o vetor não armazenará valores repetidos.

```
print("Olá humano 🖐\n")

print("Digite os ńumeros das matrículas:\n")
matriculas_array = []
while len(matriculas_array) < 10:
    try:
```

```
        input_value = input(f"{len(matriculas_array) + 1}° matrícula:\n✏️
")
        matriculas_array.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"As matrículas digitadas são:\n--------------------------")
matriculas_set = set(matriculas_array)
for m in sorted(matriculas_set):
    print(f"{m}")
```

41 - Fazer um algoritmo que: Preencha uma tupla com X números inteiros, em que o último número lido seja 999 (o último número não fará parte do vetor). E imprima a tupla na ordem inversa.

```
print("Olá humano 🖐\n")

print("Digite 10 números inteiros:\n")
numeros = ()
sair = False
while not(sair):
    try:
        input_value = int(input(f"{len(numeros) + 1}° número:\n✏️  "))
        if len(numeros) == 9:
            if input_value == 999:
                sair = True
            else:
                raise ValueError("O último valor deve ser 999!")
        else:
            numeros = numeros + (input_value,)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

print(f"As números digitadas em orderm reversa são:\n----------------------
----")
for num in sorted(numeros, reverse=True):
    print(f"{num}")
```

42 - Fazer um algoritmo que: Leia um vetor contendo 10 números, que correspondem a matrículas de alunos. Ler 3 matrículas e imprima uma mensagem informando se eles estão ou não presentes na lista

```
print("Olá humano 🖐\n")

matriculas = [10,20,30,40,50,60,70,80,90,100]
print(f"Matrículas: {matriculas}\n--------------------------------")
print("Digite o número de três matrículas e direi se estão presentes na
```

```
lista de alunos:\n")
numeros = []
while len(numeros) < 3:
    try:
        input_value = int(input(f"{len(numeros) + 1}° número:\n✏️  "))
        numeros.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

for num in numeros:
    if num in matriculas:
        print(f"O número {num} está presente na lista de matrículas!")
    else:
        print(f"O número {num} não está presente na lista de matrículas!")
```

43 - Fazer um algoritmo que: Preencha 3 listas, o primeiro com a nota da primeira prova, o segundo com a nota da segunda prova e o terceiro com a média das 2 primeiras notas, e imprima o resultado “APROVADO” para aqueles que obtiverem uma média igual ou acima de 6, e “REPROVADO” para quem obtiverem uma média abaixo de 6. OBS.: Saia do laço quando a primeira nota for igual a -1.

```
print("Olá humano 🖐\n")

notas_n1 = []
notas_n2 = []
medias = []
while len(notas_n1) < 3 and len(notas_n2) < 3:
    try:
        input_value = float(input(f"Nota do {len(notas_n1) + 1}° aluno da
primeira prova:\n✏️  "))
        notas_n1.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

    try:
        input_value = float(input(f"Nota do {len(notas_n2) + 1}° aluno da
segunda prova:\n✏️  "))
        notas_n2.append(input_value)
    except ValueError as e:
        print("🚨 Valor inválido. Tente novamente!\n")
        print(f"{e}\n")

medias = [(n1 + n2) / 2 for n1, n2 in zip(notas_n1, notas_n2)]

for i in range(0, len(notas_n1)):
    print(f"Aluno {i + 1} - Média: {medias[i]}\n")

    if medias[i] >= 6:
        print("Aprovado!\n-------------------------")
```

```
    else:
        print("Reprovado!\n--------------------------")
```