MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PIAUÍ
CENTRO DE CIÊNCIAS DA NATUREZA- CCE
Campus Universitário Ministro Petrônio Portella, Bloco 06 - Bairro Ininga,
CEP 64049-550 – Teresina-Piauí-Brasil – Fone: (86) 3215-5565 – Fax: (86) 3215-5560

Curso: Ciência da Computação
Disciplina: Laboratório de Programação
Professor: Armando
Aluno: Otávio C. França

Relatório Referente a Primeira Lista

Introdução

O presente relatório descreve uma série de programas escritos em Python para resolver os problemas da primeira lista de exercícios referente a primeira avaliação. A linguagem utilizada é o "Python" que é uma linguagem de programação de alto nível conhecida por sua sintaxe simples e legibilidade, adequada para uma variedade de aplicações, desde desenvolvimento web até ciência de dados. Os códigos foram desenvolvidos utilizando a IDE "Visual Studio Code", que oferece uma gama de recursos robustos para o desenvolvimento em Python, incluindo realce de sintaxe, depuração e controle de versão. Este relatório, e apenas este relatório, foi construído com o auxílio da inteligência artificial "ChatGPT".

Objetivos

Os objetivos dos códigos fornecidos são os seguintes:

- Programa 1 (Número, Dobro e Triplo(inteiro): Solicitar ao usuário um número inteiro e exibir o número, seu dobro e triplo.
- Programa 2 (Calculadora Simples): Criar uma calculadora simples que realiza operações básicas (soma, multiplicação, subtração e divisão) com dois números inseridos pelo usuário.
- Programa 3 (Número, Dobro e Triplo(Real): Solicitar ao usuário um número inteiro e exibir o número, seu dobro e triplo.
- Programa 4 (Saudação Personalizada): Solicitar ao usuário seu nome e exibir uma saudação personalizada, juntamente com o comprimento do nome digitado.
- Programa 5 (Números Pares): Solicitar ao usuário um número inteiro e exibir todos os números pares de 0 até o número digitado.
- Programa 6 (Números Ímpares): Solicitar ao usuário um número inteiro e exibir todos os números pares de 0 até o número digitado.
- Programa 7 (Inversão de Frase): Solicitar ao usuário uma frase e exibir a frase invertida.
- Programa 8 (Fatorial de um Número): Solicitar ao usuário um número inteiro e exibir o fatorial desse número.

- Programa 9 (Sequência Fibonacci): Solicitar ao usuário um número inteiro e exibir a sequência Fibonacci até esse número.

Metodologia

Os códigos foram desenvolvidos de maneira modular, divididos em funções que realizam tarefas específicas. A metodologia utilizada inclui:

- Solicitação de entrada do usuário para números e strings.
- Validação de entrada para garantir que os valores inseridos sejam do tipo esperado.
- Implementação de funções para realizar cálculos, processamento de strings e operações matemáticas.
- Estruturas de controle, como loops e condicionais, para direcionar o fluxo do programa.
- Reutilização de funções sempre que possível para promover a legibilidade e a manutenção do código.

Reviw Code

1. Número, Dobro e Triplo(inteiro)

```
'''
1. Escreva um programa que solicita ao usuário para digitar um número inteiro e, em
seguida, exibe esse número, o dobro e o triplo do valor desse número na tela.
'''
def valid_number(number):
    while 1:
        if number.isdigit():
            break
        else:
            number = input('invalid character, please type again!')
    return int(number)

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def process_number(number):
    double_number = number*2
    triple_number = number*3
    return double_number,triple_number

def show_numbers(number1, double, triple):
    print(f'The number entered was: {number1}\nTwice the number is: {double}\nTriple the number is: {triple}')

if __name__ == '__main__':
    number = request_number()
    double_number, triple_number = process_number(number)
    show_numbers(number, double_number, triple_number)
```

Este programa solicita ao usuário um número inteiro, calcula seu dobro e triplo e exibe os resultados.

Funcionalidades:

- valid_number(): Valida se a entrada é um número inteiro positivo.
- request_number(): Solicita um número inteiro ao usuário e valida se é um número.
- process_number(): Calcula o dobro e o triplo do número fornecido.

- show_numbers(): Exibe o número digitado pelo usuário, seu dobro e triplo na tela.

2. Calculadora Simples:

```
def validNumber(number):
    while True:
        if number.isdigit():
            break
        else:
            number = input('Invalid character, please type again!\nTry again: ')
    return int(number)

def requestNum():
    num1 = input('Enter the first Number: ')
    num1 = validNumber(num1)
    num2 = input('Enter the second Number: ')
    num2 = validNumber(num2)
    return num1, num2

def soma(num1, num2):
    return num1 + num2

def produto(num1, num2):
    return num1 * num2

def subtracao(num1, num2):
    return num1 - num2

def divisao(num1, num2):
    return num1 / num2

if __name__ == '__main__':
    print('<<<<< MENU DE OPERAÇÕES >>>>>\n 1 - Soma\n 2 - Produto\n 3 - Subtração\n 4 - Divisão')
```

```
    while True:
        choice = input('Enter your choice: ').upper()

        if choice == '1':
            num1, num2 = requestNum()
            print(f'{num1} + {num2} = {soma(num1, num2)}')
        elif choice == '2':
            num1, num2 = requestNum()
            print(f'{num1} * {num2} = {produto(num1, num2)}')
        elif choice == '3':
            num1, num2 = requestNum()
            print(f'{num1} - {num2} = {subtracao(num1, num2)}')
        elif choice == '4':
            num1, num2 = requestNum()
            if num2 == 0:
                print("Error! Cannot divide by zero.")
            else:
                print(f'{num1} / {num2} = {divisao(num1, num2)}')
        else:
            print("Invalid option!")
```

Este programa implementa uma calculadora simples que realiza operações básicas (soma, multiplicação, subtração e divisão) com dois números inseridos pelo usuário.

Funcionalidades:

- valid_number(): Valida se a entrada é um número inteiro positivo.
- requestNum(): Solicita dois números inteiros ao usuário.
- Funções para realizar operações matemáticas (soma(), produto(), subtracao(), divisao()).
- Menu de operações onde o usuário pode escolher a operação desejada.

3. Número, Dobro e Triplo(Real)

```python
'''
3. Escreva um programa que solicita ao usuário para digitar um número real e, em
seguida, exiba esse número, o dobro e o triplo do valor desse número na tela.
'''
def valid_number(number):
    while 1:
        try:
            number = float(number)
            break
        except ValueError:
            number = input('Invalid input, please enter a numeric value: ')
    return number

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def process_number(number):
    double_number = number*2
    triple_number = number*3
    return double_number,triple_number

def show_numbers(number1, double, triple):
    print(f'The number entered was: {number1}\nTwice the number is: {double}\nTriple the number is: {triple}')

if __name__ == '__main__':
    number = request_number()
    double_number, triple_number = process_number(number)
    show_numbers(number, double_number, triple_number)
```

Este programa solicita ao usuário um número inteiro, calcula seu dobro e triplo e exibe os resultados.

Funcionalidades:

- valid_number(): Valida se a entrada é um número de ponto flutuante.
- request_number(): Solicita um número inteiro ao usuário e valida se é um número.
- process_number(): Calcula o dobro e o triplo do número fornecido.
- show_numbers(): Exibe o número digitado pelo usuário, seu dobro e triplo na tela.

4. Saudação Personalizada:

```
'''
4. Escreva um programa que solicita ao usuário para digitar seu nome e, em seguida,
exiba uma saudação personalizada ("Bem vindo xxx") na tela informando quantos
caracteres tem o nome digitado.

'''
def request_Name():
    name = input('Hello, Enter your name:')
    return name
def show_Menu(name):
    print(f'Welcome {name}\nYour name has {len(name)} characters.')
if __name__ == '__main__':
    name =request_Name()
    show_Menu(name)
```

Este programa solicita ao usuário seu nome e exibe uma saudação personalizada, juntamente com o comprimento do nome digitado.

Funcionalidades:

- request_Name(): Solicita o nome do usuário.
-show_Menu(): Exibe uma mensagem de boas-vindas com o nome do usuário e o comprimento do nome.

5. Números Pares:

```
'''
5. Escreva um programa que solicita ao usuário para digitar um número inteiro e, em
seguida, exiba todos os números pares de 0 até o número digitado
'''
def valid_number(number):
    while 1:
        if number.isdigit():
            break
        else:
            number = input('Invalid character, please type again:')
    return int(number)

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def process_number(number):
    print('List of even numbers:')
    for i in range(number):
        if i % 2 == 0:
            print(f'{i} ', end='')

if __name__ == '__main__':
    number = request_number()
    process_number(number)
```

Este programa solicita ao usuário um número inteiro e exibe todos os números pares de 0 até o número digitado.

- Funcionalidades:

- valid_number(): Valida se a entrada é um número inteiro positivo.
- request_number(): Solicita ao usuário um número inteiro.

- process_number(): Calcula e exibe todos os números pares até o número fornecido.

6. Números Ímpares:

```
'''
Escreva um programa que solicita ao usuário para digitar um número inteiro e, em
seguida, exiba a soma de todos os números ímpares de 1 até o número digitado
'''
def valid_number(number):
    while True:
        if number.isdigit():
            break
        else:
            number = input('Invalid character, please type again:')
    return int(number)

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def process_number(number):
    odd_sum = 0
    print('List of odd numbers:')
    for i in range(1, number + 1):
        if i % 2 == 1:
            print(f'{i} ', end='')

if __name__ == '__main__':
    number = request_number()
    process_number(number)
```

Este programa solicita ao usuário um número inteiro e exibe todos os números ímpares de 1 até o número digitado.

- Funcionalidades:
  - valid_number(): Valida se a entrada é um número inteiro positivo.
  - request_number(): Solicita ao usuário um número inteiro.
  - process_number(): Calcula e exibe todos os números ímpares até o número fornecido.

7. Inversão de Frase:

```
'''
Escreva um programa que solicita ao usuário para digitar uma frase (máximo de 100
caracteres) e, em seguida, exiba essa frase invertida na tela
'''
def valid_frase(frase):
    while 1:
        if len(frase)<=100:
            return frase
        print('The sentence has exceeded the maximum character length!')
        frase = input('Please try again')

def request_frase():
    frase = input('Enter a sentence with a maximum of 100 characters:')
    frase = valid_frase(frase)
    return frase

def inverte_frase(frase):
    frase_invertida = frase[::-1]
    return frase_invertida

if __name__ == "__main__":
    frase = request_frase()
    frase = inverte_frase(frase)
    print('Inverted sentence: ', frase)
```

Este programa solicita ao usuário uma frase e exibe a frase invertida.

Funcionalidades:

- valid_frase(): Valida se a frase inserida tem no máximo 100 caracteres.
- request_frase(): Solicita ao usuário uma frase.
- inverte_frase(): Inverte a frase fornecida.

8. Fatorial de um Número:

```
'''
8. Escreva um programa que solicita ao usuário para digitar um número inteiro e, em
seguida, exiba o fatorial desse número.
'''
def valid_number(number):
    while True:
        if number.isdigit():
            break
        else:
            number = input('Invalid character, please type again:')
    return int(number)

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def calcular_fatorial(number):
    if number < 0:
        return "It is not possible to calculate the factorial of a negative number :("

    elif number == 0:
        return 1
    else:
        fatorial = 1
        for i in range(1, number + 1):
            fatorial *= i
        return fatorial

if __name__ == "__main__":
    number = request_number()
    resultado = calcular_fatorial(number)
    print("The factorial of", number, "is:", resultado)
```


Este programa solicita ao usuário um número inteiro e exibe o fatorial desse número.

Funcionalidades:

- valid_number(): Valida se a entrada é um número inteiro positivo.
- request_number(): Solicita ao usuário um número inteiro.
- calcular_fatorial(): Calcula o fatorial do número fornecido.

9. Sequência Fibonacci:

```python
def valid_number(number):
    while True:
        if number.isdigit():
            break
        else:
            number = input('Invalid character, please type again:')
    return int(number)

def request_number():
    number = input('Enter a number:')
    number = valid_number(number)
    return number

def fibonacci(number):
    fibonacci_seq = [0, 1]

    while fibonacci_seq[-1] + fibonacci_seq[-2] <= number:
        fibonacci_seq.append(fibonacci_seq[-1] + fibonacci_seq[-2])

    return fibonacci_seq

def show_fibonnaci(fibo):
    fibo_seq = fibonacci(fibo)

    print("Fibonacci sequence up to", fibo, ":")
    for num in fibo_seq:
        print(num, end=" ")


if __name__ == "__main__":
    number = request_number()
    show_fibonnaci(number)
```

Este programa solicita ao usuário um número inteiro e exibe a sequência Fibonacci até esse número.

Funcionalidades:

- valid_number()`: Valida se a entrada é um número inteiro positivo.
- request_number(): Solicita ao usuário um número inteiro.
- fibonacci(): Gera a sequência Fibonacci até o número fornecido.
- show_fibonnaci(): Exibe a sequência Fibonacci até o número fornecido.

Discussões

Esses códigos foram feitos para garantir clareza, legibilidade e eficiência em suas funcionalidades. Cada código resolve um problema específico de forma simples e eficaz, seguindo as melhores práticas de programação com modularização e validações de entradas. Os referentes códigos estão disponíveis no Github: https://github.com/otaviofranca/programming-laboratory. E também no replit: https://replit.com/@otaviofranca?path=folder/fist_test_otavio_franca . **Gostaria, se possível, que avaliasse os programas e sugerisse possíveis melhorias. Obs: problema numero 3 esta apesnas repetindo o enunciado**