Relatório: Trabalho 3

João Davi

Técnicas de Programação

Problema 1: Número par ou ímpar

➜ Programa que leia um número inteiro e verifique se ele é par ou ímpar. Além disso, verifique se o número está no intervalo entre 10 e 20 (inclusive).

➜ Input: Entrada do usuário com um número inteiro.

➜ Uso do if/else: Verificar se é par ou ímpar e se está no intervalo entre 10 e 20.

➜ Output: Diz se o número é par ou ímpar e se está no intervalo.

Implementação

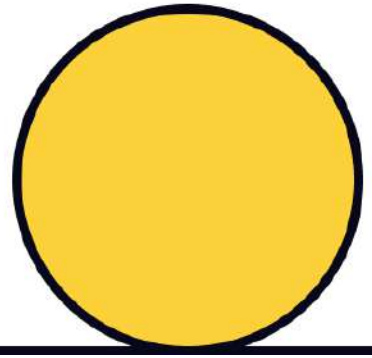
Verifica-se o resto da divisão do número da entrada num do usuário é zero.

Caso seja, o programa informará que o número é par, caso contrário informará que é ímpar.

Se o número for maior que 10 e menor que 20 o programa informará que está no intervalo entre 10 e 20.

```python
num = int(input("Digite um numero inteiro: "))

if num % 2 == 0:
    print(f"O numero {num} eh par")
else:
    print(f"O numero {num} eh impar")
if num > 10 and num < 20:
    print(f"O numero {num} esta no intervalo entre 10 e 20")
```



Problema 2: Soma maior que 100

➜ Programa que leia números do usuário até que a soma desses números seja maior que 100. Mostre quantos números foram lidos.

➜ Input: Entrada do usuário com um número inteiro.

➜ Uso do while: Repetição da entrada de dados até a condição ser falsa.

➜ Output: Quantos números foram lidos.

Implementação

O laço de repetição while continua até o valor soma seja maior que 100.

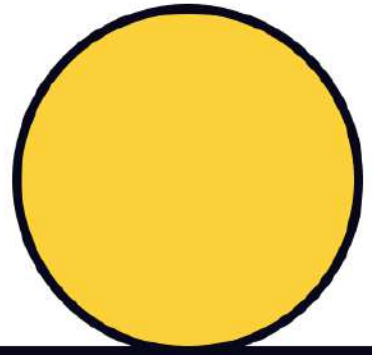
Recebe-se o valor num e soma ao valor soma e incrementa count o contador até que a condição while do seja falsa.

Imprime-se a soma atual após cada iteração e a quantidade de números lidos no finaç do programa.

```
soma = 0
count = 0

while soma <= 100:
    num = int(input("Digite um numero: "))
    soma += num
    count += 1
    print(f"Soma atual = {soma}.")

print(f"Foram lidos {count} numeros.")
```



Problema 3: Lista - Múltiplos de 3

➜ Programa que leia uma lista de 10 números inteiros e imprima os números que são múltiplos de 3.

➜ Input: Entrada do usuário com um 10 números a serem adicionados na lista criada.

➜ Uso do for: Adicionar os números na lista e imprimir os múltiplos de 3.

➜ Uso do if: Verificar se o número é divisível por 3.

➜ Output: Mútiplos de 3 presentes na lista.

Implementação

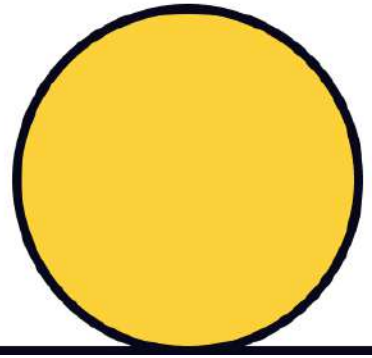
Usa-se o list() para criar uma lista vazia e adiciona 10 números inteiros usando a estrutura de repetição for.

Percorre-se, usando for, a lista, em busca de elementos divisíveis por 3 e imprime eles.

```
lista = list()

for i in range(10):
    num = int(input("Digite um numero inteiro: "))
    lista.append(num)

for num in lista:
    if num % 3 == 0:
        print(num)
```



Problema 4: Lista de nomes

➜ Programa que leia 5 nomes e os armazene em uma lista. Em seguida, imprima os nomes em ordem alfabética.

➜ Input: Entrada dos nomes a serem adicionados na lista.

➜ Uso do for: Adicionar os nomes na lista criada.

➜ Uso do sort(): Ordenar a lista em ordem alfabética.

➜ Output: Nomes em ordem alfabética.

Implementação

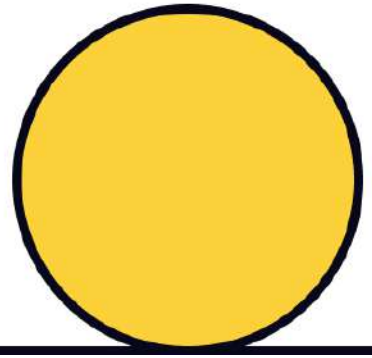
Usa-se o `list()` para criar uma lista vazia e adiciona 5 nomes usando a estrutura de repetição `for`.

Usa-se `sort()` para ordenar a `lista` criada em ordem alfabética e imprime ela no final do programa.

```
lista = list()

for i in range(5):
    nome = input("Digite um nome: ")
    lista.append(nome)

lista.sort()
print(lista)
```



Problema 5: Média Aritmética

➜ Função que receba dois números e retorne a média aritmética deles. Verifique na função se os números são válidos (inteiros ou flutuantes).

➜ Input: Dois números inteiros ou flutuantes.

➜ Uso do `if/else` e do `isinstance()`: Verificar se os números pertencem ao tipo de dado inteiro ou flutuante.

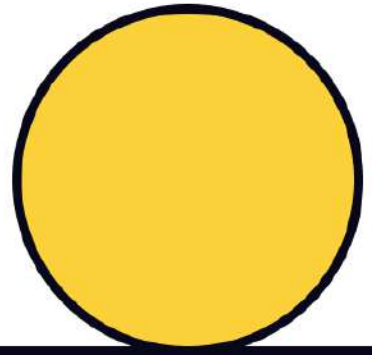
➜ Output: Média aritmética dos números ou informa que os números não são válidos.

Implementação

Verifica-se, usando o isinstance() e o for, se os números são válidos.

Caso sejam, retorna a média aritmética dos dois números de entrada.

```python
def media_aritmetica(num1, num2):
    if isinstance(num1, (int, float)) and isinstance(num2, (int, float)):
        return (num1 + num2) / 2
    else:
        return "Os números não são válidos"
```



Problema 6: Soma dos ímpares

➜ Função que receba uma lista de números e retorne a soma de todos os números ímpares da lista. Utilize um loop for para iterar pela lista.

➜ Uso do if: Verificar se o número é ímpar.

➜ Uso do for: Percorrer a lista criada.

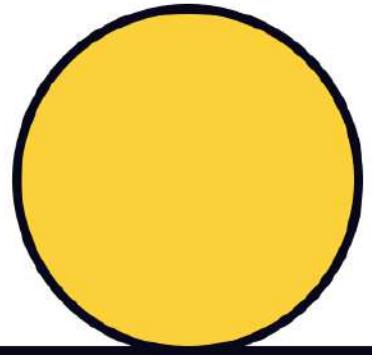
➜ Output: Soma dos números ímpares da lista.

Implementação

Usa-se da estrutura de repetição `for` para percorrer a `lista`.

Caso o resto da divisão do número `num` presente na `lista` por 2 não seja zero, incrementa-se a `soma` e retorna ela no final do programa

```
def soma_impares(lista):
    soma = 0
    for num in lista:
        if num % 2 != 0:
            soma += num
    return soma
```



Problema 7: Números pos., neg. e zeros

➜ Programa que leia uma lista de 10 números e conte quantos são positivos, negativos e zeros. Além disso, verifique se a lista contém pelo menos um número primo.

➜ Input: Entrada do usuário com os números a serem adicionados na lista.

➜ Uso do for: Adicionar os números na lista.

➜ Uso do if/elif/else: Verificar quantos números positivos, negativos e zeros têm na lista, além de verificar se têm algum número primo.

➜ Output: Quantidade de números positivos, negativos, zeros e se tem número primo.

Implementação

Cria-se a função ehprimo() para verificar se o número é primo. Usando o laço for, adiciona-se os números na lista.

Usa-se o if/else/elif para verificar se o número num e positivo, negativo ou zero e primo.

Imprime-se a quantidade de números positivos, negativos e zeros e se existir pelo menos um número primo na lista o programa informará.

```python
def ehprimo(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True

lista = list()
positivos = 0
negativos = 0
zeros = 0
primo = False

for i in range(10):
    num = int(input("Digite um numero: "))
    lista.append(num)
    if num > 0:
        positivos += 1
    elif num < 0:
        negativos += 1
    else:
        zeros += 1
    if ehprimo(num):
        primo = True

print(f"Positivos: {positivos}")
print(f"Negativos: {negativos}")
print(f"Zeros: {zeros}")
if primo:
    print("A lista contem pelo menos um numero primo.")
```

Problema 8: Fatorial de um número

➜ Função que receba um número e retorne seu fatorial. Utilize um loop while para calcular o fatorial. Verifique se a entrada é um número inteiro não negativo

➜ Input: Entrada do usuário com o número a ser calculado o fatorial.

➜ Uso do `if/elif/else`: Verificar se o número é válido e realizar a operação.

➜ Output: Fatorial do número.

Implementação

Verifica-se, usando if/elif/else se o número num é válido e calcula-se em cada caso o valor do fatorial.

Caso seja válido, usa a estrutura de repetição while para iterar e multiplicar o valor fatorial e retorna ele no final da função.

```
def fatorial(num):
    if num < 0:
        return "Numero invalido"
    elif num == 0:
        return 1
    else:
        fatorial = 1
        while num > 1:
            fatorial *= num
            num -= 1
        return fatorial
```

Problema 9: Lista ao cubo

➜ Função que receba uma lista de números e retorne uma nova lista com cada número elevado à terceira potência. Verifique se todos os elementos da lista são números.

➜ Input: Entrada do usuário com uma lista.

➜ Uso do if/all(): Verificação em todos os elementos da lista.

➜ Uso do isinstance(): Verificar se o elemento da lista é um número.

➜ Output: Lista alterada ou informa que os números não são válidos.

Implementação

Verifica-se, usando o if/else, all() e isinstance(), se cada elemento presente na lista é válido.

Caso sejam, retorna-se uma lista alterada, usando o list comprehension para utilizar uma operação sobre ela.

Caso não sejam, retorna-se uma mensagem dizendo que os números não são válidos.

```python
def lista_ao_cubo(lista):
    if all(isinstance(x, (int, float)) for x in lista):
        return [x**3 for x in lista]
    else:
        return "Os numeros nao sao validos."
```

Problema 10: Maior número

➜ Função que receba três números e retorne o maior deles. Utilize if/else para encontrar o maior. Adicione uma verificação para garantir que os três números são distintos.

➜ Input: Entrada do usuário com os três números.

➜ Uso do if/elif/else: Verificar qual número é maior

➜ Output: Maior número ou informa que os números não são distintos.

Implementação

Verifica-se, usando if/elif/else, qual o maior entre os três números e atualiza-se o valor maior para o maior número.

Retorna-se o maior número ou uma mensagem dizendo que os números não são distintos

```python
def maior_numero(num1, num2, num3):
    if num1 > num2 and num1 > num3:
        maior = num1
    elif num2 > num1 and num2 > num3:
        maior = num2
    elif num3 > num1 and num3 > num2:
        maior = num3
    else:
        return "Os numeros nao sao distintos!"
    return maior
```

Email e Github

Email: **joaodaviliberato@alu.ufc.br**

Github: **joaodaviliberato**

Repositório: **Atividades_Python/Atividade3**