UNIVERSIDADE UNIGRANRIO

GABRIEL FONSECA CORREIA 0701028

Algoritmo e Estrutura de Dados AP1

SUMÁRIO

1. INTRODUÇÃO

Escrever um programa que leia 30 números e que, para cada número lido, verifique e codifique de acordo com as regras abaixo:
a. Se o número for par, empilhe em uma lista chamada PAR;
b. Se o número for ímpar, empilhe em uma lista chamada IMPAR;
c. Se for zero, desempilhe um elemento de cada pilha. Caso a pilha esteja vazia, exiba na tela uma mensagem informando que a lista está vazia.
d. Ao final do programa, desempilhe todos os elementos das duas listas exibindo-os na tela.

O programa foi desenvolvido em linguagem Python, (conforme aprovado pelo professor Oswaldo Borges em conversa no Fórum da Disciplina), utilizando Programação Orientada a Objetos. Segue abaixo o código.

2. CÓDIGO

```
#!/usr/local/bin/python
#
# -*- coding: utf-8 -*-

from time import sleep

class Node:
    def __init__(self, data=0, last_node=0):
        self.data = data
        self.last = last_node

    def __repr__(self):
        return f'{self.data} -> {self.last}'


class Pile:
    def __init__(self):
        self.top = None

    def __repr__(self):
        return f"[{str(self.top)}]"

    def is_even(self, num):
        if num % 2 == 0:
            return True
        else:
            return False

    def insert_data(self, new_data):
        new_node = Node(new_data)
        new_node.last = self.top
        self.top = new_node

    def remove_data(self):
        if self.top == None:
              print("Não é possível remover um valor de uma pilha
vazia.")
        else:
            self.top = self.top.last
```

```
    def line(tipo):
        print(tipo * 30 )


    ########## Main ############

    even = Pile()
    uneven = Pile()

    for i in range(30):
        try:
            line('~=')
            num = int(input(f'Digite o {i + 1}° número: '))
            where_to_upload = even.is_even(num)
            if num == 0:
                even.remove_data()
                uneven.remove_data()
                print(even)
                print(uneven)
            else:
                if even.is_even(num) and num != 0:
                    even.insert_data(num)
                        print(f"Insere valor {num} no topo da pilha
PARES: {even}")
                else:
                    uneven.insert_data(num)
                        print(f"Insere valor {num} no topo da pilha
ÍMPARES: {uneven}")
        except ValueError as error:
            print(f"Somente números são permitidos. {error}")


    #### CONCLUSÃO ####

    line("==")
    print("Visualizando a Pilha PARES completa: ", even)
    line("~=")

    while even.top != None:
        even.remove_data()
         print("Removendo elemento que está no topo da pilha PARES: ",
even)
        sleep(0.5)
```

```
    line("==")
    print("Visualizando a Pilha ÍMPARES completa: ", uneven)
    line("~=")

    while uneven.top != None:
        uneven.remove_data()
         print("Removendo elemento que está no topo da pilha ÍMPARES:
", uneven)
        sleep(0.5)

    line('~~')
    print("Fim das FILAS!")
    line('~~')
```

https://github.com/gothmate/AP1-Algoritmos_pilha/blob/main/ap1_comClasses.py