# -\*- coding: utf-8 -\*-

"""

===================

Amostras Analizator

===================

Um programa que permite analisar amostras e gerar a tabela de frequências e histograma

"""

print(\_\_doc\_\_)

# Bibliotecas

import math

import os

import time

# Comandos de documentação do código

\_\_author\_\_ = "Nome Completo do Aluno 1 (GRR), Nome Completo do Aluno 2 (GRR) e Nome Completo do Aluno 3 (GRR)"

\_\_license\_\_ = "GPL"

\_\_version\_\_ = "1.0"

\_\_maintainer\_\_ = "Jackson Antonio do Prado Lima" # A pessoa que mantém o código

\_\_email\_\_ = "japlima@inf.ufpr.br"

# Status tipicamentente é: Prototype, Development ou Production

\_\_status\_\_ = "Prototype"

def insere\_amostra(amostras):

'''

Função que insere amostras

'''

pass # Troque esta linha pelo seu respectivo código para esta função.

def lista\_amostra(amostras):

'''

Função que lista as amostras

'''

pass # Troque esta linha pelo seu respectivo código para esta função.

def calcula\_tabela\_frequencia(amostras):

'''

Função que calcula a tabela e o histograma de frequência

'''

pass # Troque esta linha pelo seu respectivo código para esta função.

def le\_nome\_arquivo():

'''

Função que lê o nome do arquivo informado pelo usuário.

O nome do arquivo deve obrigatoriamente terminar com .dat

Essa função é usada tanto na leitura quanto na escrita do arquivo.

'''

# Solicita que o usuário digite o nome do arquivo

arquivo = input('Nome do arquivo: ')

# Apenas arquivos que contenham extensão .dat são permitidos.

# Ex.: amostras.dat

while(not arquivo.endswith(".dat")):

print('Nome de arquivo inválido!')

arquivo = input('Forneça outro nome de arquivo:')

# Retorna o nome do arquivo

return arquivo

def salva\_amostra\_arquivo(nome\_arquivo, amostras):

'''

Função que salva as amostras em um arquivo

'''

resposta = input('Deseja salvar as alteracoes feitas? (S para sim)')

# Se o usuário desejar salvar as amostras em um arquivo

if resposta.upper() == 'S':

# Se o usuário ainda não informou um nome para o arquivo, solicita um

if nome\_arquivo == '':

nome\_arquivo = le\_nome\_arquivo()

# Salva as amostras no arquivo

with open(nome\_arquivo, 'w') as arquivo:

for amostra in amostras:

arquivo.write(str(amostra)+'\n')

def le\_amostra\_arquivo(amostras):

'''

Função que lê as amostras contidas em um arquivo

'''

"""

Criamos uma variável com o nome do arquivo, pois retornaremos

esse nome para ser utilizado no menu principal.

Assim, salvamos as alterações que realizamos no mesmo arquivo.

Caso o usuário não desejar abrir um arquivo de dados,

iremos verificar o nome do arquivo na hora de salvar para saber

se necessitamos que ele informe o nome do arquivo.

"""

nome\_arquivo = ''

resposta = input('Deseja abrir um arquivo com os dados? (S para sim)')

# Se o usuário desejar ler amostras salvas em um arquivo

if resposta.upper() == 'S':

# Solicita o noem do arquivo ao usuário

nome\_arquivo = le\_nome\_arquivo()

"""

Abre o arquivo pra leitura e salva na variável arquivo.

Ao abrir um arquivo, o programa 'bloqueia' o arquivo para ser utilizado por ele

Usando o comando "with open(nome\_arquivo, 'r') as arquivo"

permite que após a leitura o arquivo fique "desbloqueado"

para ser utilizado novamente.

"""

with open(nome\_arquivo, 'r') as arquivo:

# Lê as linhas do arquivo

for linha in arquivo:

# Como cada amostra está em uma linha, guarda a amostra lida no vetor de amostras

amostras.append(float(linha.strip('\n')))

return nome\_arquivo

def menu():

'''

Função que contém o menu do programa

'''

# Mostra as opções pro usuário na tela

print('Menu:')

print('\t1 - Insere amostras')

print('\t2 - Listar amostras')

print('\t3 - Calcular Tabela de Frequência')

print('\t4 - Sair')

# Retorna o que o usuário digitar

return int(input('Digite a opção desejada: '))

def main():

'''

Programa principal

'''

# guardaremos as amostras em um vetor/array

amostras = []

# lê as amostras salvas em um arquivo (se o usuário desejar)

nome\_arquivo = le\_amostra\_arquivo(amostras)

# Limpa a tela do terminal para ficar mais "elegante" a apresentação

os.system('cls' if os.name == 'nt' else 'clear')

# chama a função menu que mostra o menu e solicita uma opção ao usuário

opcao = menu()

# enquanto o usuário não digitar a opção: 4 - Sair

while opcao != 4:

# Se for a opção: 1 - Insere amostras

if opcao == 1:

# chama a função que insere amostras no vetor de amostras

insere\_amostra(amostras)

# Se for a opção: 2 - Listar amostras

elif opcao == 2:

# chama a função que imprime as amostras na tela

# passando o vetor que contém as amostras

lista\_amostra(amostras)

time.sleep(5)

# Se for a opção: 3 - Calcular Tabela de Frequência

elif opcao == 3:

# chama a função que calcula a tabela de frequência

# e monta o histograma

calcula\_tabela\_frequencia(amostras)

time.sleep(5)

# Se digitou uma opção de menu inválida

else:

print("\nOpção inválida! Por favor, informe uma opção válida.\n")

# Faz o programa esperar 1 segundo e depois continua

time.sleep(2)

# Limpa a tela do terminal

os.system('cls' if os.name == 'nt' else 'clear')

# Mostra ao usuário o menu novamente e aguarda ele escolher uma opção

opcao = menu()

# Antes de finalizar o programa, salva as amostras em um arquivo

salva\_amostra\_arquivo(nome\_arquivo, amostras)

if \_\_name\_\_ == "\_\_main\_\_":

'''

Quando executar o programa python esse trecho de código será executado e chamará o programa principal.

Para um melhor entendimento acesse os links abaixo:

http://www.devfuria.com.br/python/entenda-\_\_name\_\_-\_\_main\_\_/

https://pythonhelp.wordpress.com/2012/06/15/por-que-\_\_name\_\_-\_\_main\_\_/

'''

# Chama a função main

main()