Mundo 3 - Tratamento de erros

1) Reescreva a função LeiaInt() que fizemos nos desafios anteriores, incluindo agora a possibilidade da digitação de um número de tipo inválido. Aproveite e crie também uma função leiaFloat() com a mesma funcionalidade.

Guanabara)

Vamos entender por partes.

LeiaInt

1. definimos a função onde a mesma recebe um parâmetro chamada msg
2. empacotamos agora um segmento do código com um loop infinito
3. Agora entramos numa estrutura de tratamento de erro, iniciando em try
4. variável n recebe o int input de msg (sendo o parâmetro inicial da função)
5. se der erro, ou seja se ocorrer um except do tipo valueerroe ou typeerror ele printa uma mensagem informando que está errado e continua para voltar ao inicio do loop
6. se der um except de interrupção pelo teclado ele printa que o usuário interrompeu e retorna 0 como valor
7. senão ele retorna o valor de n em int input

LeiaFloat

1. Funciona basicamente igual à estrutura anterior, a única diferença é o input ser do tipo float (número real)

```
def leiaInt(msg):
    while True:
        try:
            n = int(input(msg))
        except (ValueError,TypeError):
            print('\033[31mErro: por favor, digite um número inteiro válid
o.\033[m')
            continue
        except KeyboardInterrupt:
            print('\033[31mEntrada de dados interrompida pelo usuário\033
[m')
            return 0
        else:
            return n

def leiaFloat(msg):
    while True:
        try:
            n = float(input(msg))
```

```
        except (ValueError, TypeError):
            print('\033[31mErro: por favor, digite um número real válido.\0
33[m')
            continue
        except KeyboardInterrupt:
            print('\033[31mUsuário preferiu não digitar este número\033[m')
            return 0
        else:
            return n


n1 = leiaInt('Digite um número Inteiro: ')
n2 = leiaFloat('Digite um número Real: ')
print(f'O valor inteiro digitado foi {n1} e o real foi {n2}')
```

2) Crie um Python que teste se o site Pudim está acessível pelo computador usado

Não funcionou o método do Guanabara)

Erro:

<urlopen error [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (_ssl.c:1000)>

```
import urllib
import urllib.request


try:
    site = urllib.request.urlopen('http://www.pudim.com.br')
except Exception as erro:
    print('Deu erro')
    print(erro)
else:
    print('Deu certo')
```

a)

A forma que eu descobri que desse certo foi essa, com a biblioteca requests.

Basicamente ela funciona junto com a urllib3 e a certifi

1. Colocamos um try onde a variável url recebe um input para o usuário digitar o site
2. site recebe a função requests.get que ele faz uma requisitação ao site indicado
3. variável status traz a informação de qual valor do status a requisição teve

4. Se der erro ele printa deu ruim e mostra o erro
5. Se der certo ele informa e mostra o valor do status

```
import requests


try:
    url = input('Digite o site: ')
    site = requests.get(url,timeout=5)
    status = site.status_code
except Exception as erro:
    print('O site não está acessível')
    print(erro)
else:
    print('O site está acessível')
    print(f'Status de conexão {status}')
```

3) Crie um pequeno sistema modularizado que permita cadastrar pessoas pelo seu nome e idade em um arquivo de texto simples. O sistema só vai ter 2 ações: cadastrar uma nova pessoa e listar todas as pessoas cadastradas

Este é o último projeto do curso e preciso mostrar em partes pois ele é composto por:

1. sistema main que é o core do projeto
2. lib interface para a parte gráfica
3. lib arquivo para a parte de manipulação de texto

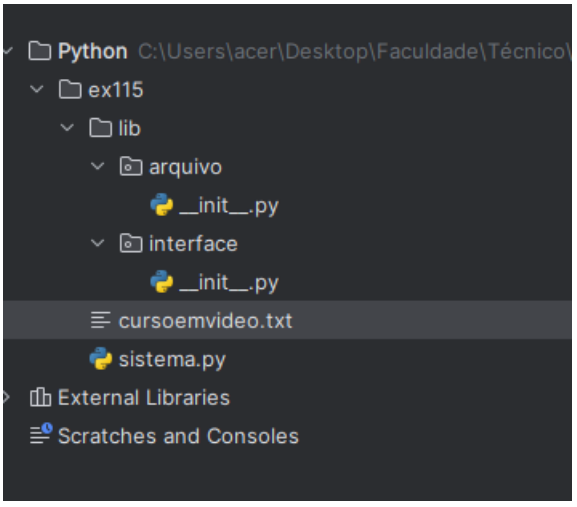
Vou iniciar explicando a interface primeiramente, após isso o sistema de manipulação de texto para juntarmos tudo na main do sistema.

Separei o código em 3 páginas:

Interface do sistema

Arquivo

Main


Python C:\Users\acer\Desktop\Faculdade\Técnico\
ex115
lib
arquivo
__init__.py
interface
__init__.py
cursoemvideo.txt
sistema.py
External Libraries
Scratches and Consoles

Interface do sistema

Como temos várias funções, vou explicar uma por uma e no final mostro como ficou completa.

LeiaInt

Trouxemos a mesma função explicada anteriormente, mas basicamente:

1. definimos a função onde a mesma recebe um parâmetro chamada msg
2. empacotamos agora um segmento do código com um loop infinito
3. Agora entramos numa estrutura de tratamento de erro, iniciando em try
4. variável n recebe o int input de msg (sendo o parâmetro inicial da função)
5. se der erro, ou seja se ocorrer um except do tipo valueerroe ou typeerror ele printa uma mensagem informando que está errado e continua para voltar ao inicio do loop
6. se der um except de interrupção pelo teclado ele printa que o usuário interrompeu e retorna 0 como valor
7. senão ele retorna o valor de n em int input

```
def leiaInt(msg):
    while True:
        try:
            n = int(input(msg))
        except (ValueError, TypeError):
            print('\033[31mErro: por favor, digite um número inteiro válid
o.\033[m')
            continue
        except KeyboardInterrupt:
            print('\033[31mEntrada de dados interrompida pelo usuário\033
[m')
            return 0
        else:
            return n
```

Linha

Esta função é super simples, ela recebe um parâmetro chamado tamanho que será o tamanho da linha onde por padrão é 42 se não for especificado, após isso ela retorna o hífen multiplicado pelo valor do tamanho ou seja 42 se não for especificado.

```
def linha(tam = 42):
    return '-'*tam
```

Cabeçalho

Literalmente serve para mostrar na tela algum texto como cabeçalho entre 2 linhas sendo estas puxadas pela função anterior, então ele recebe um parâmetro chamado tx printa linha() depois o txt

centralizado em 42 espaços e abaixo outra linha.

```
def cabeçalho(txt):
    print(linha())
    print(txt.center(42))
    print(linha())
```

Menu

Por fim temos o menu principal, ele vai ficar dando loop e mostrando a cada comando que o usuário pedir igual qualquer sistema que tenha somente uma tela.

Funciona da seguinte forma:

1. A função recebe um parâmetro em lista
2. chamamos o cabeçalho como ‘Menu Principal’
3. declaramos uma variável c valendo 1 para servir de contador
4. na estrutura for fazemos: para cada item na lista, printa o contador e o item
5. c recebe c + 1 para iterar
6. printa linha() no final
7. variável opc recebe o leiaInt() para verificar se realmente é um valor inteiro ou não que o usuário está digitando
8. retorna opc

```
def menu(lista):
    cabeçalho('MENU PRINCIPAL')
    c = 1
    for item in lista:
        print(f'\033[33m{c}\033[m - \033[34m{item}\033[m')
        c+=1
    print(linha())
    opc = leiaInt('\033[32mSua Opção: \033[m')
    return opc
```

Tudo junto ficaria

```
def leiaInt(msg):
    while True:
        try:
            n = int(input(msg))
        except (ValueError, TypeError):
            print('\033[31mErro: por favor, digite um número inteiro válid
o.\033[m')
            continue
        except KeyboardInterrupt:
```

```
            print('\033[31mEntrada de dados interrompida pelo usuário\033
[m')
            return 0
        else:
            return n


def linha(tam = 42):
    return '-'*tam


def cabeçalho(txt):
    print(linha())
    print(txt.center(42))
    print(linha())


def menu(lista):
    cabeçalho('MENU PRINCIPAL')
    c = 1
    for item in lista:
        print(f'\033[33m{c}\033[m - \033[34m{item}\033[m')
        c+=1
    print(linha())
    opc = leiaInt('\033[32mSua Opção: \033[m')
    return opc
```

Arquivo

Este módulo serve para manipularmos os arquivos txt que vamos usar para salvar os usuários cadastrados.

Vamos utilizar a função cabeçalho, sendo assim precisamos importar:

```
from ex115.lib.interface import cabeçalho
```

Partindo daí vamos ver as 4 funções que criamos.

arquivoExiste

Esta função serve para verificar se o arquivo txt existe ou não no diretório que estamos mexendo, se não existir o python precisa criar, então:

1. Definimos a função onde vamos passar um nome sendo o nome do arquivo que depois vamos indicar na main do sistema
2. Damos um try onde a variável a recebe um comando open tentando abrir o arquivo com o nome do parâmetro com o comando rt que é um READ TEXT, literalmente só vai tentar ler
3. Feito isso damos um a.close() que fecha o arquivo
4. Se não conseguirmos abrir ou fechar pois deu uma exceção FileNotFoundError, retornamos False
5. Senão retornamos True

```
def arquivoExiste(nome):
    try:
        a = open(nome,'rt')
        a.close()
    except FileNotFoundError:
        return False
    else:
        return True
```

criarArquivo

1. Caso não encontremos o arquivo, precisamos criamos, então definimos a função incialmente passando o nome do arquivo como parâmetro
2. Damos um try onde a recebe o open nome do arquivo com o comando WT+ ou seja WRITE TEXT +, ele vai tentar escrever e se não existir um arquivo o + indica a solicitação de criação
3. Se der alguma exceção por algum motivo o python vai printar o erro
4. Senão ele vai informar que o arquivo nome foi criado

```
def criarArquivo(nome):
    try:
```

```
        a = open(nome,'wt+')
        a.close()
    except:
        print('Houve um erro na criação do arquivo')
    else:
        print(f'Arquivo {nome} criado com sucesso')
```

LerArquivo

1. Esta função serve para que possamos mostrar na tela todas as pessoas cadastradas no txt então inicialmente passamos o parâmetro nome do arquivo
2. Damos um try onde a recebe o comando open do arquivo com RT, READ TEXT
3. Se der alguma exceção ele informa que foi impossível
4. Senão ele printa cabeçalho Pessoas Cadastradas
5. Iniciamos um for linha em a
6. variável dado recebe linha.split com o ; sendo o delimitador, onde em dado salvamos uma lista
7. dado[1] recebe dado[1].replace('\n' por '') ou seja retiramos o espaço em dado[1]
8. Feito isso ele faz o print do dado posição 0 e dado posição 1
9. finalmente ele fecha o arquivo

```
def lerArquivo(nome):
    try:
        a = open(nome,'rt')
    except:
        print('Erro ao ler o arquivo')
    else:
        cabeçalho('Pessoas Cadastradas')
        for linha in a:
            dado = linha.split(';')
            dado[1] = dado[1].replace('\n','')
            print(f'{dado[0]:<30}{dado[1]:>3} anos')
    finally:
        a.close()
```

Cadastrar

1. primeiramente declaramos como `def cadastrar(arq, nome='desconhecido', idade=0):`, ou seja recebe o arq, o nome que por padrão é desconhecido e idade por padrão é 0
2. damos um try com a recebendo open o arquivo com o parâmetro at sendo APPEND TEXT
3. se der uma exceção informamos que houve um erro
4. Senão, aí colocamos outro try, sendo a.write ou seja escrevemos as informações
5. se der uma exceção informamos que deu erro
6. senão informamos que o registro de nome foi feito e fechamos o arquivo.

```
def cadastrar(arq, nome='desconhecido', idade=0):
    try:
        a = open(arq, 'at')
    except:
        print('Houve um erro na abertura do arquivo')
    else:
        try:
            a.write(f'{nome};{idade}\n')
        except:
            print('Houve um erro na hora de escrever os dados')
        else:
            print(f'Novo registro de {nome} adicionado')
            a.close()
```

Tudo junto ficaria:

```
from ex115.lib.interface import cabeçalho

def arquivoExiste(nome):
    try:
        a = open(nome,'rt')
        a.close()
    except FileNotFoundError:
        return False
    else:
        return True

def criarArquivo(nome):
    try:
        a = open(nome,'wt+')
        a.close()
    except:
        print('Houve um erro na criação do arquivo')
    else:
        print(f'Arquivo {nome} criado com sucesso')


def lerArquivo(nome):
    try:
        a = open(nome,'rt')
    except:
        print('Erro ao ler o arquivo')
    else:
        cabeçalho('Pessoas Cadastradas')
        for linha in a:
            dado = linha.split(';')
```

```
            dado[1] = dado[1].replace('\n','')
            print(f'{dado[0]:<30}{dado[1]:>3} anos')
    finally:
        a.close()


def cadastrar(arq, nome='desconhecido', idade=0):
    try:
        a = open(arq, 'at')
    except:
        print('Houve um erro na abertura do arquivo')
    else:
        try:
            a.write(f'{nome};{idade}\n')
        except:
            print('Houve um erro na hora de escrever os dados')
        else:
            print(f'Novo registro de {nome} adicionado')
            a.close()
```

Main

Vamos agora agrupar os dois arquivos que criamos só com funções e utilizaremos em uma main ou core do sistema.

Primeiramente importamos os módulos que vamos utilizar

```
from ex115.lib.interface import *
from time import sleep
from ex115.lib.arquivo import *
```

Agora vamos passo por passo

Definindo o nome do arquivo, verificar se ele existe e solicitar criação

Criamos uma variável chamada arq com o nome do arquivo, se arquivoExiste for falso então ele chama a função de criar o arquivo.

```
arq = 'cursoemvideo.txt'

if not arquivoExiste(arq):
    criarArquivo(arq)
```

Main do código

Todo o restante do código fica nesta parte do código

1. Enquanto verdadeiro (loop infinito)
2. resposta recebe a função menu onde passamos as funções que vão aparecer em loop
3. se resposta for 1 então ele chama a opção lerarquivo
4. se a resposta for 2 ele mostra cabeçalho de um texto, informa nome, idade e chama a função cadastrar com o nome do arquivo, nome e idade informando anteriormente
5. se a resposta for 3 ele mostra o cabeçalho e quebra o código
6. senão ele informa que teve algum erro
7. No final ele dá um sleep para não embolar o terminal

```
while True:
    resposta = menu(['Ver pessoas cadastradas','Cadastrar nova Pessoas','Sa
ir do Sistema'])
    if resposta == 1:
        #opção de listar o conteúdo de um arquivo
        lerArquivo(arq)
    elif resposta == 2:
        cabeçalho('NOVO CADASTRO')
        nome = input('Nome: ')
```

```
        idade = leiaInt('Idade: ')
        cadastrar(arq, nome, idade)
    elif resposta == 3:
        cabeçalho('Saindo do sistema....Até Logo!')
        break
    else:
        print('\033[31mERRO! Digite uma opção válida!\033[m')
    sleep(1)
```

Tudo junto fica:

```
from ex115.lib.interface import *
from time import sleep
from ex115.lib.arquivo import *

arq = 'cursoemvideo.txt'

if not arquivoExiste(arq):
    criarArquivo(arq)


while True:
    resposta = menu(['Ver pessoas cadastradas','Cadastrar nova Pessoas','Sa
ir do Sistema'])
    if resposta == 1:
        #opção de listar o conteúdo de um arquivo
        lerArquivo(arq)
    elif resposta == 2:
        cabeçalho('NOVO CADASTRO')
        nome = input('Nome: ')
        idade = leiaInt('Idade: ')
        cadastrar(arq, nome, idade)
    elif resposta == 3:
        cabeçalho('Saindo do sistema....Até Logo!')
        break
    else:
        print('\033[31mERRO! Digite uma opção válida!\033[m')
    sleep(1)
```