Cotuca – Unicamp

Mariana Marietti da Costa - 24140

Rafaelly Maria Nascimento da Silva – 24153

1º ano de Informática

GALERIA VIRTUARTE

PROJETO I - Relatório

1. Introdução e objetivo:

Este projeto é um programa escrito por meio do Visual Studio Code em Python que utiliza a biblioteca Tkinter para selecionar arquivos para gerenciar uma galeria de arte virtual, oferecendo opções para cadastrar, listar e criar uma página web com base nas obras de arte, além de gerar o Triângulo de Pascal, permitindo criar uma galeria de arte virtual que exiba o ano, título, estilo, artista, valor e imagem da obra.

O fluxo do programa consiste em:

1. Atividade 1 (Cadastro de obras de arte):

O usuário escolhe a opção 1, seleciona um arquivo para armazenar os dados das obras, digita eles (ano, mês, autor, nome, estilo, valor estimado e URL da foto) e os grava no arquivo.

Iniciamos o projeto dia 08/04/2024 às 08h20, na monitoria, tentando interpretar primeiramente essa opção. A partir disso produzimos o resto.

2. Atividade 2 (Listagem de obras de arte):

O usuário escolhe a opção 2 e seleciona o arquivo que as obras foram cadastradas. Logo, o programa lê as obras do arquivo e as exibe na tela, de maneira organizada, somando o total de obras cadastradas e o valor total de todas.

3. Atividade 3 (Página web de obras de arte):

O usuário escolhe a opção 3, seleciona um arquivo que contém as obras, e o programa gera uma página HTML com as informações das obras, salvando-as em um arquivo.

Começamos a desenvolver a tabela no dia 10, fazendo pesquisas sobre como colocar html em python e chamar um navegador.

4. Atividade 4 (Triângulo de Pascal):

O usuário escolhe a opção 4, digita o número de linhas desejado para o Triângulo de Pascal, e o programa gera e exibe-o.

5. Terminar a Execução:
O usuário escolhe a opção 0 e o programa se encerra.

2. Auxílio da monitoria:

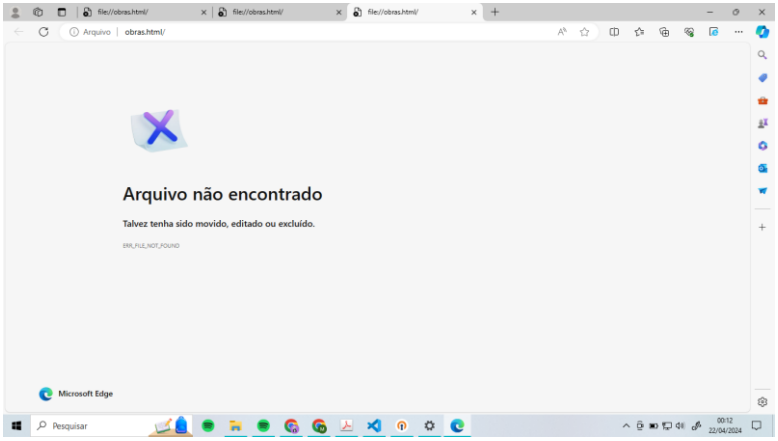
A participação da monitoria foi crucial em diversas frentes do projeto. Além de ajudar na interpretação do projeto, desempenhou um papel fundamental na identificação e correção dos erros.

Ademais, os monitores compartilharam conhecimentos que não estavam plenamente estabelecidos, o que enriqueceu significativamente o processo de desenvolvimento.

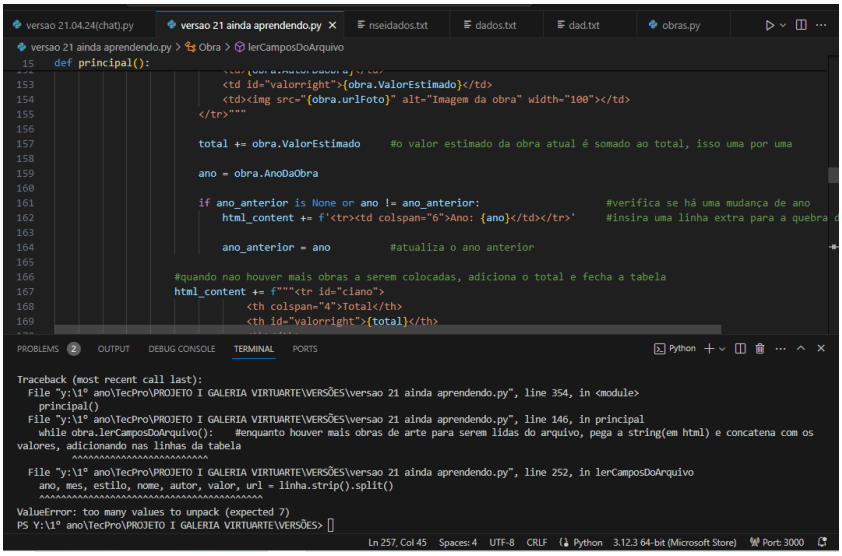
Esse engajamento construtivo foi essencial para aprimorar a compreensão do projeto, alcançar resultados satisfatórios, e até mesmo otimizar a eficácia do trabalho em dupla. Sua presença ativa e dedicada foi essencial para garantir a qualidade e precisão em cada etapa do processo.

3. Erros e problemas encontrados:

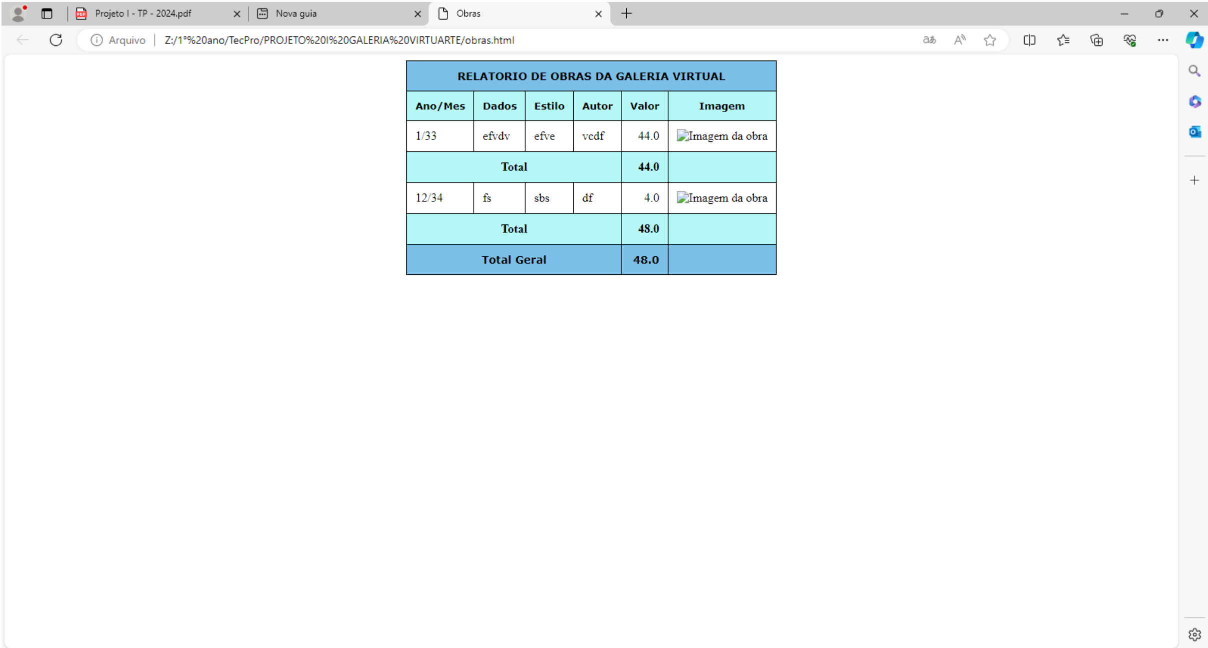
1. O arquivo do exercício 3 não era encontrado.



2. O código não conseguia passar e nem ler os valores para a tabela.



Obs.: Apareceu, porém, as partes não foram divididas corretamente e não ficou amarelo. Também não está ordenando por ano, logo, o total geral não é calculado direito. E depois, só lia certas linhas.



RELATORIO DE OBRAS DA GALERIA VIRTUAL					
Ano/Mes	Dados	Estilo	Autor	Valor	Imagem
1/33	efvdv	efve	vcdf	44.0	Imagem da obra
Total				44.0	
12/34	fs	sbs	df	4.0	Imagem da obra
Total				48.0	
Total Geral				48.0	

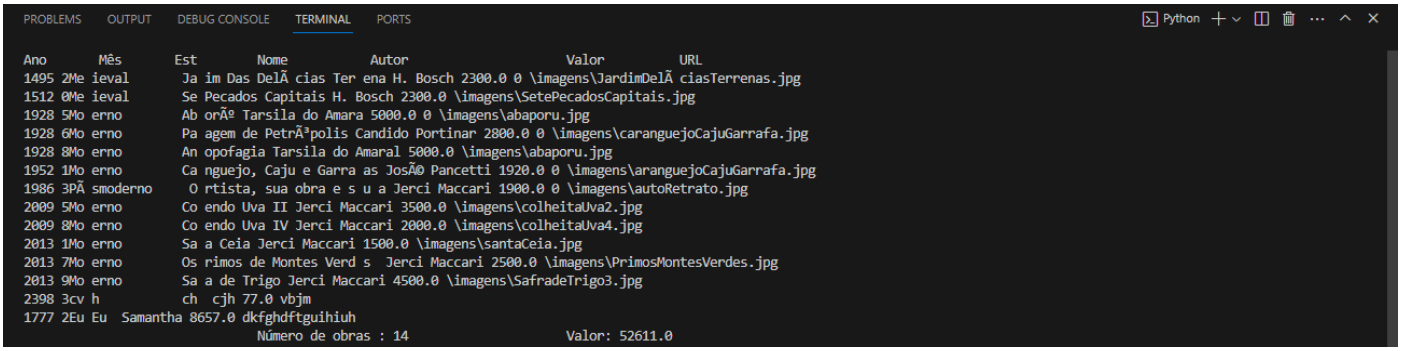
3. O valor não estava formatado corretamente:

```
File "z:\1ºano\tp\projeto\main.py", line 91, in <module>
   seletor()
 File "z:\1ºano\tp\projeto\main.py", line 78, in seletor
   opcUm()
 File "z:\1ºano\tp\projeto\main.py", line 10, in opcUm
   arquivo = filedialog.askopenfilename(title='Selecione o arquivo', multiple=False, filetypes=tipos)

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
 File "C:\Python312\Lib\tkinter\filedialog.py", line 384, in askopenfilename
   return Open(**options).show()
          ^^^^^^^^^^^^^^^^^^^^^^^^^
 File "C:\Python312\Lib\tkinter\commondialog.py", line 45, in show
   s = master.tk.call(self.command, *master._options(self.options))
       ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
_tkinter.TclError: bad Macintosh file type "texto"
```

Obs.: Esse erro apareceu quando a janela do open dialog era aberta. O que foi difícil nesse erro foi que ele se deu em um código que não foi feito, ou seja, o código foi feito automaticamente quando codificava a opção 2 no arquivo main.

4. Encontrou-se dificuldade ao achar a formatação correta para o exercício 2.



4. Respectivas soluções:

1. Chamar o arquivo para ser aberto na web corretamente.

```
html_nome_arq = "obras.html"                    #define o nome do arquivo HTML que será gerado

with open(html_nome_arq, 'w') as file:          #abre o arquivo HTML em modo de escrita/gravação ('w').
    #o arquivo é aberto dentro de um bloco with, o que garante que o arquivo será fechado corretamente após seu uso
    file.write(html_content)                    #html_content é escrito no arquivo obras.html (formata)

print(f"Arquivo HTML gerado com sucesso: {html_nome_arq}\nVocê está sendo direcionado para a página...\n")

webbrowser.open(html_nome_arq)                  #abre a url
```

2. Descobriu-se a maneira certa de se passar valores.

```
while obra.lerCamposDoArquivo():              #enquanto houver linhas para ler
    ano = obra.AnoDaObra                      #instancia

    #tudo para mostrar somente o total:

    if ano_anterior == None or ano == ano_anterior:          #se não houver obra anterior ou o ano for igual ao anterior
        total += obra.ValorEstimado                          #vai somar os valores DO ANO ESPECÍFICO

    if ano != ano_anterior and ano_anterior != None:         #verifica se há uma mudança de ano, se houver, mostra o total já dos anos que foram som
        html_content += f"""<tr id="ciano">
            <th colspan="4">Total</th>
            <th id="valorright">{total}</th>
            <th></th>
        </tr>"""
        total = obra.ValorEstimado                           #o valor estimado das obras desse ano printa!

    ano_anterior = ano                                       #atualiza o ano anterior


    #mostrar dados da obra:

    if contador % 2 == 0:                                    #se a linha for par, mostra amarelo
        html_content += f"""<tr id="yellow">
            <td >{obra.AnoDaObra}/{obra.MesDaObra}</td>
            <td >{obra.NomeDaObra}</td>
            <td >{obra.Estilo}</td>
            <td >{obra.AutorDaObra}</td>
            <td id="valorright">{obra.ValorEstimado}</td>
            <td ><img src="{obra.UrlFoto}" alt="Imagem da obra" width="100"></td>
        </tr>"""

    else:                                                    #se não, mostra branco
        html_content += f"""<tr>
            <td>{obra.AnoDaObra}/{obra.MesDaObra}</td>
            <td>{obra.NomeDaObra}</td>
            <td>{obra.Estilo}</td>
            <td>{obra.AutorDaObra}</td>
            <td id="valorright">{obra.ValorEstimado}</td>
            <td><img src="{obra.UrlFoto}" alt="Imagem da obra" width="100"></td>
        </tr>"""


    contador += 1

    total_geral += total                                     #o valor estimado da obra atual é somado ao total

ano_anterior = ano                                           #atualiza o ano anterior
```

3 e 4. Descobrimos o lugar correto para transformar o valor estimado para float e a formatação correta.

```
def __str__(self):
    #função que concatena os atributos de dados da obra e devolve a string com essa concatenação, para leitura (la no metodo principal)
    return f"{self.AnoDaObra} {self.MesDaObra} {self.Estilo} {self.NomeDaObra} {self.AutorDaObra} {self.ValorEstimado} {self.UrlFoto}"
```

```
def preencherCampos(self, ano, mes, estilo, nome, autor, valor, url):   #receberá como parâmetros os valores dos campos de dados da obra
    #preenche os campos com espaços conforme necessário
    self.AnoDaObra = ano.ljust(4)
    self.MesDaObra = mes.ljust(2)               #seria 2, mas eu estou colocando mais dois de espaço
    self.Estilo = estilo.ljust(15)
    self.NomeDaObra = nome.ljust(28)
    self.AutorDaObra = autor.ljust(20)
    self.ValorEstimado = valor.ljust(8)
    self.UrlFoto = url.ljust(100)
    #armazenará nos atributos de Obra que descrevem uma obra

def lerCamposDoArquivo(self):

    if self._abertoParaGravacao == False:               #se abertoParaGravacao valer false
        linha = self._arquivo.readline().strip()        #será lida a próxima linha de dados do arquivo texto

        if linha:                                       #verifica se a linha não está vazia
            #separa os campos da linha pelos espaços em branco:
            self.AnoDaObra = linha[0:4]
            self.MesDaObra = linha[4:6]
            self.Estilo = linha[6:21]
            self.NomeDaObra = linha[21:49]
            self.AutorDaObra = linha[49:69]
            self.ValorEstimado = float(linha[69:77])
            self.UrlFoto = linha[77:]

            return True                 #retorna True indicando sucesso na leitura dos campos

        else:
                return False            #retorna False indicando que não há mais linhas para ler

def __init__(self, nomeArq, paraGravacao):          #construtor da classe
                                                    #nomeArq = nome do arquivo onde as obras serão armazenadas
                                                    #paraGravacao = indica se o arquivo será aberto para gravação(True) ou leitura(False)
    self.AnoDaObra : str[4]= ''                             #string
    self.MesDaObra : str[2]= ''                             #string
    self.Estilo : str[15] = ''                              #string
    self.NomeDaObra : str[28] = ''                          #string
    self.AutorDaObra : str[20] = ''                         #string
    self.ValorEstimado : float = 0.0                        #não será string pois será somado logo mais
    self.UrlFoto : str[100] = ''                            #string

    self._abertoParaGravacao = paraGravacao         #atributo booleano que indica se o arquivo está aberto para gravação ou não, no caso, é igual ao p

    self._arquivo = None                    #inicializa o atributo _arquivo com None que representa o arquivo onde as obras serão armazenadas(nomeArq)
    if paraGravacao == True:                #aberto em modo de escrita ('a') se paraGravacao for True (indica que o arquivo será aberto para gravação)
        self._arquivo = open(nomeArq, 'a')
    else:                                   #e em modo de leitura ('r') caso contrário
        self._arquivo = open(nomeArq, 'r')
```

5. Tempo gasto em cada atividade:

Atividade 1: 4h

Atividade 2: 12h

Atividade 3: 9h

Atividade 4: 3h

Total: 28 horas (esse valor não diz respeito à quantidade de horas gastas desenvolvendo o relatório do projeto).

6. Conclusão sobre o projeto e sugestões:

Apesar das inúmeras dificuldades e erros enfrentados ao longo do processo, foi possível elaborar o trabalho proposto, que representou um verdadeiro desafio. A complexidade da tarefa exigiu um esforço extra e uma abordagem meticulosa para superar os obstáculos encontrados. No entanto, com perseverança e dedicação, conseguiu-se superar cada dificuldade, contando com a valiosa assistência dos monitores e determinação própria.

Desse modo, ao refletirmos sobre o desenvolvimento deste projeto, é evidente a importância de um enunciado claro sobre as instruções. Recomenda-se que futuros enunciados de projetos sejam redigidos de forma a evitar ambiguidades e garantir uma compreensão precisa dos objetivos e requisitos, facilitando a interpretação e execução do projeto por parte dos alunos. Além disso, fornecer exemplos ou casos de uso concretos pode ajudar a ilustrar os conceitos abordados e a orientar os participantes na direção correta.

Compreendendo que a clareza é fundamental desde as fases iniciais do projeto, percebe-se que a elaboração de um enunciado preciso é o alicerce sobre o qual todo o processo é construído. Assim, ao enfrentarmos as adversidades e lidarmos com os desafios, entendemos a necessidade de uma comunicação clara, sendo essencial a comunicação entre os membros da equipe, para que a sabedoria que cada um domina possa ser encontrada no erro do outro. Logo, a partir dessa experiência, podemos fortalecer ainda mais nosso conhecimento sobre a área de programação e desempenhar um melhor rendimento durante as aulas.

7. Referências e fontes:

W3schools: https://www.w3schools.com/

Bobby hadz: https://bobbyhadz.com/blog/open-html-file-in-the-browser-using-python

Programiz: https://www.programiz.com/python-programming/methods/string/strip

Cotuca - 2024