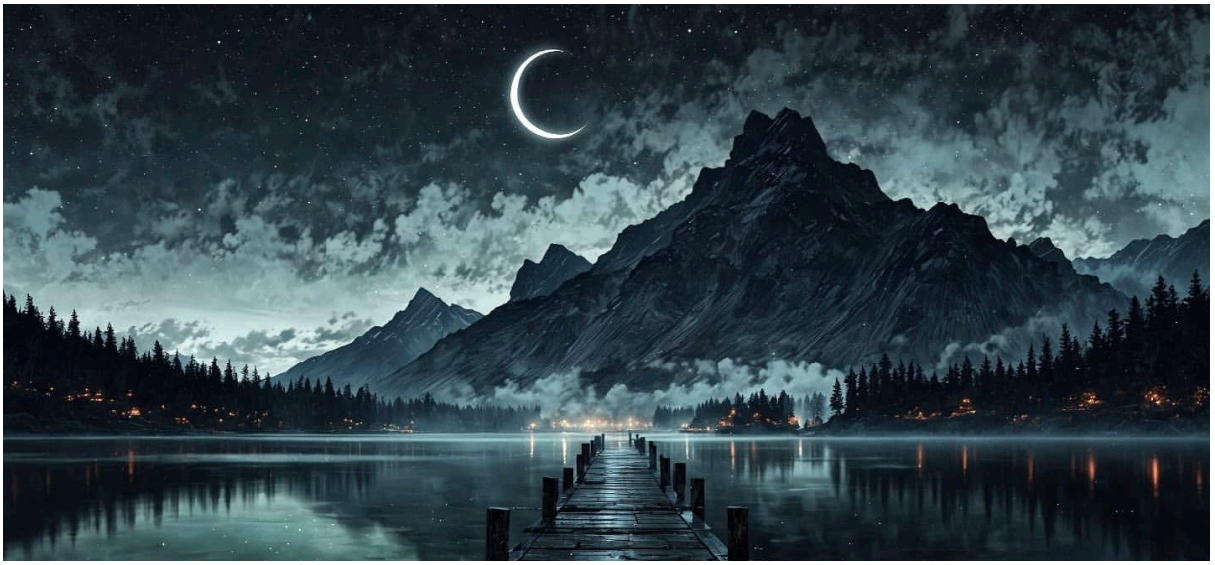

Versionamento com Git

1. Python Desktop Simples

Desafio 1

Código com erro:

```
def dividir(a, b):
    return a / b

a = 10
b = 0
resultado = dividir(a, b)
print(f"O resultado da divisão é: {resultado}")
```

Código corrigido:

```
def dividir(a, b):
    if b == 0:
        return "Erro: divisão por zero não é permitida!"
    return a / b
```

```
a = 10
b = 0
resultado = dividir(a, b)
print(f"O resultado da divisão é: {resultado}")
```

Erro Localizado:

O código tentava dividir um número por zero (`b = 0`), o que causa um erro de execução (`ZeroDivisionError`). Para corrigir isso adicionei uma verificação antes da divisão para evitar o erro, retornando uma mensagem caso o divisor seja zero.

Desafio 2

Código com erro:

```
nome_usuario = "Professor(a)"
print("Olá, " + nome_usuar)
```

Código corrigido:

```
nome_usuario = "Professor(a)"
print("Olá, " + nome_usuario)
```

Erro Localizado:

A variável foi escrita incorretamente no `print` (`nome_usuar` em vez de `nome_usuario`), o que causa `NameError.`Corrigi o nome da variável para que seja o mesmo usado na declaração.

Desafio 3

Código com erro:

```
frutas = [maçã, "banana", "laranja"]
print(f"Minhas frutas favoritas: {frutas}"
```

Código corrigido:

```
frutas = ["maçã", "banana", "laranja"]
print(f"Minhas frutas favoritas: {frutas}")
```

Erro Localizado:

A palavra `maçã` não estava entre aspas dentro da lista, o que fazia o Python pensar que era uma variável inexistente. Coloquei `"maçã"` entre aspas para que o Python entenda como uma string.

2. Web (HTML, CSS e JavaScript)

Desafio 4

Código com erro:

```
<!DOCTYPE html>
<html>
<head>
   <title>Bug de Tag</title>
   <link rel="stylesheet" href="style.css">
</head>
```

```
<body>
    <h1 class="titulo">Página com Bug de Estrutura</h1>
    <p>Este é o primeiro parágrafo. </pa>
    <div id="status">Verificando JS...</div>
    <script src="script.js"></script>
</body>
</html
```

Código corrigido:

```
<!DOCTYPE html>
<html>
<head>
    <title>Bug de Tag</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <h1 class="titulo">Página com Bug de Estrutura</h1>
    <p>Este é o primeiro parágrafo. </p>
    <div id="status">Verificando JS...</div>
    <script src="script.js"></script>
</body>
</html>
```

Erro Localizado:

A tag `<p>` estava sendo fechada incorretamente com `</pa>`, e o `</html>` estava sem o `>` final.
Corrigi o fechamento da tag de parágrafo e adicionei o `>` que faltava no final do arquivo.

Desafio 5

Código com erro:

```
titulo {
 font-size: 24px;
 color: blue
 background-color: yellow;
 padding: 10px;
 border: 1px solid black;
}
```

Código corrigido:

```
.titulo {
    font-size: 24px;
    color: blue;
    background-color: yellow;
    padding: 10px;
    border: 1px solid black;
}
```

Erro Localizado:

Faltava o ponto e vírgula (`;`) após a linha `color: blue`, o que impede o CSS de aplicar corretamente os estilos seguintes.

Desafio 6

Código com erro:

```
let valor = 5;
let elemento = document.getElementById("status");
```

```
if (valor = 10) {
  elemento.innerText = "A condição é sempre verdadeira! (Bug de Lógica)";
} else {
  elemento.innerText = "A condição é falsa.";
}
```

Código corrigido:

```
let valor = 5;
let elemento = document.getElementById("status");

if (valor === 10) {
   elemento.innerText = "A condição é verdadeira!";
} else {
   elemento.innerText = "A condição é falsa.";
}
```

Erro Localizado:

Foi usado o operador de atribuição (`=`) em vez do operador de comparação (`==` ou `===`) dentro do `if`, o que fazia a condição ser sempre verdadeira.

3. Django (Python Web Framework)

Desafio 7

Código com erro:

```
# views.py no app Django
# Tarefa: Descomentar a linha de importação para que HttpResponse funci
on
```

```
e.
# from django.http import HttpResponse
def saudacao(request):
    return HttpResponse("Olá do Django!")
```

Código corrigido:

```
from django.http import HttpResponse

def saudacao(request):
    return HttpResponse("Olá do Django!")
```

Erro Localizado:

A linha que importava `HttpResponse` estava comentada, então o Django não reconhecia a função usada na view.

Desafio 8

Código com erro:

```
from django.urls import path
from . import views
def outra_view(request):
    pass
urlpatterns = [
    path('ola/', views.saudacao)
    path('outra/', views.outra_view),
]
```

Código corrigido:

```
rom django.urls import path
from . import views

def outra_view(request):

    pass
urlpatterns = [
    path('ola/', views.saudacao)
    path('outra/', views.outra_view),
]
```

Erro Localizado:

- Faltava uma vírgula entre os dois `path()` dentro da lista `urlpatterns`.
- A view `outra_view` não retornava nenhuma resposta, o que geraria erro ao acessá-la.

4. Kivy (Mobile com Python)

Desafio 9

Código com erro:

```
# Tarefa: Corrigir o nome da classe 'Ap' para 'App' na linha 1 e 5.
from kivy.app import Ap
from kivy.uix.label import Label
class MinhaApp(Ap):
    def build(self):
```

```
        return Label(text='Kivy App com Bug de Importação')
MinhaApp().run()
```

Código corrigido:

```
from kivy.app import App
from kivy.uix.label import Label

class MinhaApp(App):
    def build(self):
        return Label(text='Kivy App com Bug de Importação')

MinhaApp().run()
```

Erro Localizado:

A classe `App` foi escrita incorretamente como `Ap`, o que faz o import falhar.

Desafio 10

Código com erro:

```
from kivy.app import App
from kivy.uix.button import Button
class OutraApp(App):
    def build(self)
        Button(text='Corrija o Retorno!').run()
OutraApp().run()
```

Código corrigido:

```
from kivy.app import App
from kivy.uix.button import Button

class OutraApp(App):
    def build(self):
        return Button(text='Corrija o Retorno!')

OutraApp().run()
```

Erro Localizado:

- Faltava os dois pontos (`:`) após `def build(self)` .
- O código chamava `.run()` no botão, o que está incorreto, pois quem deve executar o método `run()` é o aplicativo (`App`).
- O `build()` não retornava nada.