

Roteiro de Atividade Prática

Nome: Henrique Ferreira Da Silva

Turma: 2B

Atividade 1: Sistema de Gerenciamento de Biblioteca

Descrição do exercício:

Crie uma classe Livro com atributos como título, autor e ano. Em seguida, crie uma classe Biblioteca, que armazena uma coleção de objetos Livro e inclui métodos para adicionar e listar livros.

Tempo estimado: 15 minutos

Lista de materiais

- Computador com internet;
- Caderno para anotações;
- 1 caneta.

Procedimento experimental

1. Analise o exemplo-base para a criação do código:

```
class Livro:
```

```
    def __init__(self, titulo, autor, ano):
```

```
        self.titulo = titulo
```

```
        self.autor = autor
```

```
        self.ano = ano
```

```
    def adicionar_livro(self, livro):
```

```
        self.livros.append(livro)
```

```
    def listar_livros(self):
```

```
for livro in self.livros:
```

```
    print(f"{livro.titulo}, {livro.autor}, {livro.ano}")
```

```
# Uso das classes
```

```
biblioteca = Biblioteca()
```

```
biblioteca.adicionar_livro(Livro("1984", "George Orwell", 1949))
```

```
biblioteca.listar_livros()
```

2. Partindo do princípio de que já temos a classe Livro no código, sua tarefa será a criação da classe Biblioteca para a finalização do contexto.
3. Anote o código desenvolvido nas linhas seguintes e envie por meio do AVA.

```
4. class Livro:
5.     def __init__(self, titulo, autor, ano):
6.         self.titulo = titulo
7.         self.autor = autor
8.         self.ano = ano
9.
10. class Biblioteca:
11.     def __init__(self):
12.         self.livros = []
13.
14.     def adicionar_livro(self, livro):
15.         self.livros.append(livro)
16.
17.     def listar_livros(self):
18.         for livro in self.livros:
19.             print(f'Título: {livro.titulo}, Autor: {livro.autor}, Ano:
{livro.ano}')
20.
21. biblioteca = Biblioteca()
22. biblioteca.adicionar_livro(Livro('1984', 'George Orwell', 1949))
23. biblioteca.adicionar_livro(Livro('O Senhor dos Anéis', 'J.R.R. Tolkien',
1954))
24.
25. biblioteca.listar_livros()
```

Atividade 2: Sistema de Controle de Acesso

Descrição do exercício:

Desenvolva uma classe `Usuario` com atributos privados, como nome e senha. Implemente métodos para alterar a senha, garantindo que a nova senha atenda a critérios específicos de segurança (por exemplo, tamanho mínimo).

Tempo estimado: 15 minutos

Procedimento experimental

1. Analise o exemplo-base para a criação do código:

```
class Usuario:
    def __init__(self, nome, senha):
        self.__nome = nome
        self.__senha = senha

# Uso da classe
usuario = Usuario("joao123", "senha123")
usuario.alterar_senha("novaSenha123")
```

2. Agora, a partir do código analisado, crie o método que permita a alteração da senha, conforme solicitado.
3. Anote o código desenvolvido nas linhas seguintes e envie por meio do AVA.

```
4. class Usuario:
5.     def __init__(self, nome, senha):
6.         self.__nome = nome
7.         self.__senha = senha
8.     def alterar_senha(self, nova_senha):
9.         self.__senha = nova_senha
10.
11. # Uso da classe
12. usuario = Usuario("joao123", "senha123")
13. usuario.alterar_senha("novaSenha123")
14.
```

Atividade 3: Aplicação de Herança e Polimorfismo

Descrição do Exercício:

Construa uma hierarquia de classes para representar diferentes tipos de veículos, como Carro e Motocicleta. Ambas as classes herdam de uma classe-base Veículo, mas implementam de maneira diferente um método como acelerar.

Tempo estimado: 10 minutos

Procedimento experimental

1. Analise o exemplo-base para criação do código:

```
class Veiculo:
    def acelerar(self):
        raise NotImplementedError("Método acelerar deve ser implementado.")

class Carro(Veiculo):
    def acelerar(self):
        print("Carro acelerando.")

class Motocicleta(Veiculo):
    def acelerar(self):
        print("Motocicleta acelerando.")
```

2. Agora, a partir do código analisado, temos as classes, mas suas referências não estão especificadas. Sua tarefa é viabilizar que a classe possa ser utilizada.
3. Anote o código desenvolvido nas linhas seguintes e envie por meio do AVA.

4. `class Veiculo:`

```
5.     def acelerar(self):
6.         raise NotImplementedError("Método acelerar deve ser implementado.")
7.
8. class Carro(Veiculo):
9.     def acelerar(self):
10.        print("Carro acelerando.")
11.
12. class Motocicleta(Veiculo):
13.     def acelerar(self):
14.        print("Motocicleta acelerando.")
15.
16. meu_carro = Carro()
17. minha_moto = Motocicleta()
18.
19. meu_carro.acelerar()
20. minha_moto.acelerar()
21.
```