Programação Orientada a Objetos em Python

1. No contexto de Programação Orientada a Objetos em Python, qual é a principal finalidade do método __init__ dentro de uma classe?

a) É um método comum que precisa ser chamado manualmente para criar um objeto.

b) Serve para destruir um objeto e liberar memória quando ele não é mais necessário.

c) É o construtor da classe, responsável por inicializar os atributos de uma nova instância (objeto).

d) É utilizado para definir métodos que serão compartilhados por todas as instâncias da classe.

2. Na classe Musica do material, total_musicas é um atributo de classe. Qual das seguintes afirmações descreve corretamente a diferença entre um atributo de classe e um atributo de instância (como self.titulo)?

a) Atributos de classe são únicos para cada objeto, enquanto atributos de instância são compartilhados por todos.

b) Atributos de classe são compartilhados por todas as instâncias da classe, enquanto cada instância tem sua própria cópia dos atributos de instância.

c) Atributos de classe só podem ser números inteiros, e atributos de instância podem ser de qualquer tipo.

d) Atributos de classe não podem ser modificados após a criação da classe, ao contrário dos atributos de instância.

3. O que acontece se você tentar chamar um método que foi adicionado à definição de uma classe depois que um objeto dessa classe já foi instanciado?

a) O Python atualiza automaticamente o objeto existente, e o método funciona normally.

b) O programa gera um erro de sintaxe (SyntaxError), pois a classe foi modificada.

c) O objeto já existente não terá o novo método, e a chamada resultará em um AttributeError.

d) O método será executado, mas não terá acesso a nenhum dos atributos do objeto.

4. Qual é o papel da palavra-chave self como primeiro parâmetro dos métodos de uma classe em Python?

a) É uma referência à própria classe, usada para acessar atributos de classe.

b) É uma variável global que armazena todos os objetos criados a partir da classe.

c) É opcional e só é necessário em métodos que modificam os atributos do objeto.

d) É uma referência à instância específica do objeto que está chamando o método, permitindo acessar seus próprios atributos e métodos.

5. Na classe Playlist, o método adicionar_musica verifica se uma música já existe antes de adicioná-la. Como essa verificação de duplicatas é implementada?

a) Comparando os objetos de música diretamente com o operador ==.

b) Verificando se o título e o artista da nova música (ignorando maiúsculas/minúsculas) já correspondem a alguma música na lista.

c) Usando o método count() da lista para ver se a música já aparece nela.

d) Verificando apenas o atributo duracao para identificar músicas idênticas.

6. Qual das seguintes afirmações sobre a criação de atributos em Python, com base no exemplo do notebook, é verdadeira?

a) Todos os atributos de um objeto devem ser declarados previamente dentro do método __init__.

b) É possível adicionar um novo atributo a uma única instância de objeto em tempo de execução, sem que outras instâncias da mesma classe o possuam.

c) Se você adicionar um atributo a um objeto, todos os outros objetos da mesma classe receberão automaticamente esse atributo com um valor padrão None.

d) A criação dinâmica de atributos é considerada uma má prática e gera um aviso (warning) em Python.

7. No método listar_musicas da classe Playlist, a linha musicas_ordenadas = sorted(self.musicas, key=lambda m: getattr(m, ordenar_por, "")) é usada para ordenar a lista. O que a função getattr(m, ordenar_por, "") faz nesse contexto?

a) Define um novo atributo no objeto m com o nome contido na variável ordenar_por.

b) Gera um erro se o atributo especificado por ordenar_por não existir no objeto m.

c) Obtém o valor do atributo do objeto m cujo nome é o mesmo que o valor da string na variável ordenar_por.

d) Verifica se o objeto m possui um atributo chamado ordenar_por.

8. O material apresenta um comparativo da classe Musica em Python e em Java. Qual das diferenças apontadas NÃO é correta?

a) Java exige a declaração explícita do tipo de cada atributo (tipagem estática), enquanto Python não (tipagem dinâmica).

b) Em Java, o construtor tem o mesmo nome da classe, enquanto em Python, ele se chama __init__.

c) Em Java, a palavra-chave this tem o mesmo papel que self em Python, referindo-se à instância atual.

d) Tanto em Python quanto em Java é obrigatório usar a palavra-chave new para instanciar objetos.

Gabarito:

1. c
2. b
3. c
4. d
5. b
6. b
7. c
8. d