INSTITUTO FEDERAL
Brasília

BOLSA FUTURO DIGITAL
PROGRAMA DE FORMAÇÃO ACELERADA

Lista de Exercícios

1. Elabore um programa em Python orientado a objetos que leia um número n (o número de termos de uma progressão aritmética), a1 (o primeiro termo da progressão) e r (razão) e escreva todos os termos dessa progressão, bem como a soma dos elementos (Fórmulas da P.A.: an = a1 + r x (n – 1) e S = n * (a1 + an) / 2).

2. Faça um programa em Python orientado a objetos que, a partir de uma string digitada pelo usuário, imprima:
   - O número de caracteres da string;
   - A string com todas suas letras em maiúsculo;
   - A string com todas suas letras em minúsculo;
   - O número de vogais da string;
   - Se a substring “IFB” aparece no texto (ignorando maiúsculas/minúsculas).

3. Faça um programa em Python orientado a objetos que receba uma frase dada pelo usuário e a criptografe. A criptografia consistirá na troca das vogais da frase por um número correspondente: A – 4, E – 3, I – 1, O – 0, U – 8.

4. Crie um programa em Python utilizando orientação a objetos para gerenciar uma lista de produtos de um carrinho de compras. Implemente uma classe chamada Produto, com

atributos privados (__nome, __preco e __quantidade) e métodos públicos para acessar e modificar esses valores de forma segura (getters e setters). Em seguida, implemente uma classe CarrinhoDeCompras, que mantém uma lista de objetos Produto e possui métodos para adicionar um produto ao carrinho, remover um produto pelo nome, calcular o valor total da compra e listar os produtos com suas quantidades e preços individuais. O programa principal deve permitir que o usuário adicione e remova produtos, visualize o conteúdo do carrinho e o total da compra. Utilize encapsulamento para proteger os dados dos produtos e boas práticas de organização orientada a objetos.

5. Implemente uma classe Impressora com o método imprimir(Documento d), que recebe um objeto da classe Documento e imprime suas informações na tela. A classe Documento deve conter os atributos título e conteúdo. A classe Impressora apenas utiliza o documento, sem manter uma referência permanente a ele, caracterizando uma relação de dependência. Desenvolva um programa com um menu interativo no console que permita criar vários documentos, visualizar seu conteúdo por meio da impressora e encerrar o programa.

6. Desenvolva uma classe Departamento com nome e um vetor de objetos Funcionario, onde cada funcionário tem nome e salário, permitindo que funcionários existam independentemente do departamento para que possam ser adicionados ou removidos livremente (agregação). Implemente métodos no Departamento para adicionar funcionários, calcular a média salarial e listar todos os funcionários. Crie um programa com menu interativo no console que permita criar departamentos, criar funcionários, adicionar funcionários a departamentos, listar funcionários e mostrar a média salarial, além de permitir sair do programa.

7. Implemente uma classe Casa que contenha vários objetos do tipo Comodo, sendo que cada cômodo possui nome (ex: sala,

cozinha, banheiro) e área. Os cômodos devem ser implementados de forma que só existam se a casa existir (relação de composição), e não devem ser acessados ou manipulados diretamente de fora. Implemente um método na casa para calcular a área total, somando as áreas de todos os cômodos. Desenvolva um programa com menu interativo no console que permita criar uma casa, adicionar cômodos à casa, listar os cômodos da casa, calcular e exibir a área total da casa e encerrar o programa.

8. Desenvolva um sistema em Python para representar personagens de um jogo de RPG. Crie uma classe base chamada Personagem, com os atributos nome e nivel, e um método atacar() que imprime uma mensagem genérica de ataque. Em seguida, crie as subclasses Guerreiro e Mago. A classe Guerreiro deve possuir o atributo adicional forca e sobrescrever o método atacar() para exibir uma mensagem de ataque físico, enquanto a classe Mago deve ter o atributo mana e sobrescrever o método atacar() para representar um ataque mágico. Na classe principal, crie instâncias das três classes e invoque o método atacar() para demonstrar o uso de herança e polimorfismo em tempo de execução.

9. Implemente em Python um sistema para uma plataforma de cursos online que utilize herança e polimorfismo, armazenando os dados em uma lista. Crie uma classe base chamada Participante, com os atributos nome e email, e um método emitirCertificado() que retorna uma mensagem genérica. Em seguida, crie as subclasses Aluno, com o atributo curso, e Instrutor, com o atributo especialidade, ambas sobrescrevendo o método emitirCertificado() com mensagens específicas: o aluno recebe um certificado de conclusão do curso e o instrutor um certificado de participação como palestrante. O programa deve conter um menu com as opções: 1) Cadastrar participante, 2) Listar participantes, 3) Emitir certificados, e 0) Sair. O usuário deve escolher entre cadastrar um aluno ou instrutor, e os dados devem ser

armazenados em uma lista de objetos do tipo Participante. O método emitirCertificado() deve ser chamado de forma polimórfica para cada participante cadastrado.

10. Implemente um sistema em Python para representar personagens do universo dos Cavaleiros do Zodíaco, utilizando herança múltipla. Crie uma classe Personagem com os atributos nome e constelacao, e métodos como apresentar(). Crie também duas classes auxiliares: CavaleiroDeBronze com o atributo poder_de_luta e um método golpe_especial(), e CavaleiroDeOuro com o atributo casa_do_zodiaco e um método defender_casa(). Em seguida, implemente a classe CavaleiroHibrido, que herda tanto de CavaleiroDeBronze quanto de CavaleiroDeOuro, combinando os comportamentos de ambas. O programa deve oferecer um menu interativo no console com as opções de cadastrar personagens, listar todos os personagens, executar os golpes especiais ou defesas, e encerrar o programa. Explore o uso da herança múltipla e do polimorfismo para definir o comportamento de cada tipo de cavaleiro.

11. Desenvolva um sistema para gerenciar veículos de transporte público em uma cidade inteligente. Crie uma classe abstrata VeiculoTransporte, com os atributos placa e capacidadePassageiros, e um método abstrato calcularCustoOperacional() que retorna o custo por quilômetro. Crie as subclasses Onibus, com o atributo consumoPorKm (litros/km), e Metro, com consumoEnergiaPorKm (kWh/km). Cada uma deve implementar o cálculo do custo com valores fictícios: R$ 6,00 por litro de diesel e R$ 0,80 por kWh. Na função principal, permita criar objetos dos dois tipos e exibir os custos operacionais usando polimorfismo. Implemente tratamento de exceções para validar os dados de entrada: placa não pode ser vazia, e os valores numéricos devem ser positivos.

12. No universo de Dragon Ball, desenvolva um sistema em Python para simular um torneio de artes marciais, utilizando orientação a objetos. Crie uma classe totalmente abstrata

chamada Lutador, contendo apenas métodos abstratos para obter o nome do lutador, o nível de poder e realizar um ataque. Em seguida, implemente subclasses como Saiyajin, Androide e Namekuseijin, cada uma com um comportamento específico ao atacar. O sistema deve conter um menu interativo que permita cadastrar lutadores de diferentes raças, listar todos os lutadores inscritos no torneio e simular ataques, demonstrando o uso de herança, abstração total e polimorfismo. Implemente também tratamento de exceções, garantindo que os nomes não estejam vazios e que o nível de poder seja um valor numérico positivo.