

PRÁTICA LABORATORIAL 01

Objetivos:

- Classes
- Objectos

EXERCÍCIOS

1. Escreve um programa onde crias uma classe chamada "Aluno" com atributos de nome, número de estudante e média. Cria três instâncias da classe "Aluno", define os seus atributos usando o construtor e imprime os dados de cada aluno formatados (por exemplo: "Aluno: João Silva | Nº: 12345 | Média: 15.5").
2. Escreve um programa onde crias uma classe chamada "Gato" com um nome (String), cor do pelo (String) e um som (String) que por defeito deve ser "Miau".
 - Deve conter um método miar() que imprime na consola o respetivo som.
 - Deve conter um método brincar() que imprime "[nome do gato] está a brincar!".
 - Cria duas instâncias da classe "Gato", define os seus atributos usando o construtor.
 - Modifica o som de um dos gatos usando o setter (por exemplo, para "Mrrr") e invoca os métodos miar() e brincar() para ambos.
3. Escreve um programa onde crias uma classe chamada "Triangulo" com atributos base e altura.
 - Instancia dois triângulos com valores diferentes.
 - Cria métodos para calcular a área do triângulo.
 - Compara qual dos dois triângulos tem maior área e imprime o resultado.
4. Escreve um programa onde crias uma classe chamada "Cilindro" com atributos raio e altura.
 - Instancia um cilindro.
 - Cria métodos para calcular o volume ($\pi \times \text{raio}^2 \times \text{altura}$) e a área da superfície lateral ($2 \times \pi \times \text{raio} \times \text{altura}$) do cilindro.
 - Imprime os resultados formatados com 2 casas decimais.
5. Cria uma classe chamada "Telemovel" com os seguintes atributos: marca, modelo, bateria (percentagem, valor entre 0-100) e ligado (boolean).
 - Cria um método "ligar()" que verifica se há bateria (>0%) e altera o estado para ligado, imprimindo mensagens apropriadas.
 - Cria um método "desligar()" que altera o estado para desligado.
 - Cria um método "usarTelemovel(int minutos)" que consome 1% de bateria por cada 10 minutos de uso (só funciona se estiver ligado).
 - Cria dois objetos da classe "Telemovel" e testa todos os métodos, simulando diferentes cenários de uso.

6. Cria uma classe sem atributos chamada "OperacoesMatematicas".
 - Elabora métodos para potência (recebe base e expoente), raiz quadrada (recebe um número) e média (recebe três números). Estes métodos devem retornar o resultado do cálculo.
 - Cria um objeto dessa classe e usa os métodos para realizar algumas operações matemáticas, imprimindo os resultados na consola (deve realizar, pelo menos, uma de cada).
7. Cria uma classe chamada "Filme" com os seguintes atributos: título, realizador, género, duração (em minutos) e classificação etária. Em seguida, cria um método na classe "Filme" chamado "mostrarInformacao" que imprime na consola todos os atributos do filme de forma formatada. Cria também um método "podeVerFilme" que recebe a idade do espetador como parâmetro e retorna se pode ou não ver o filme. De seguida, cria dois objetos da classe "Filme" e invoca os métodos para cada um deles.
8. Cria uma classe ContaBancaria que tenha os atributos titular, numeroConta, saldo e ativa (boolean).
 - Cria um construtor que receba o titular e número de conta como parâmetros, inicialize o saldo a 0 e a conta como ativa.
 - Cria um método chamado estadoConta que retorne "Conta Positiva" se o saldo for maior que 0, "Conta a Zero" se for igual a 0, ou "Conta Negativa" caso contrário.
 - Cria um método depositar que receba um valor e aumente o saldo (só funciona se a conta estiver ativa).
 - No método main, cria um objeto dessa classe, faz um depósito de 500€ e apresenta o estado da conta.
9. Cria uma classe Restaurante que tenha os atributos nome, capacidadeMaxima e lugaresOcupados (deve ser inicializada a 0), e os métodos reservarMesa e libertarMesa, que recebam o número de lugares desejados como parâmetro e atualizem os lugares ocupados.
 - Para reservar uma mesa, deve analisar se existe capacidade suficiente. Caso exista, deve atualizar os lugares ocupados. Caso não exista capacidade suficiente, deve informar o utilizador que a reserva não é possível por lotação esgotada.
 - No método main, cria um objeto dessa classe Restaurante com capacidade para 50 pessoas.
 - Invoca o método para reservar 20 lugares e imprime os lugares ocupados.
 - Invoca o método para libertar 5 lugares.
 - Invoca o método para reservar 40 lugares (deve dar erro de lotação).
10. Cria uma classe Jogador que tenha os atributos nome, pontos, nivel e equipa, e os métodos ganharPontos e mostrarEstatisticas.
 - O método ganharPontos deve receber uma quantidade de pontos como parâmetro e adicioná-los ao total. Sempre que os pontos atingirem múltiplos de 100, o nível deve aumentar 1 (por exemplo: 100 pontos = nível 1, 200 pontos = nível 2, etc.).

- O método `mostrarEstatisticas` deve exibir na consola o nome, pontos totais, nível atual e equipa do jogador de forma formatada.
- No método `main`, cria um objeto dessa classe e invoca o método `ganharPontos` várias vezes (com valores diferentes) e depois mostra as estatísticas finais do jogador.

11. Cria um programa para fazer a gestão de carteiras digitais:

- Cria uma classe chamada `"CarteiraDigital"` com os seguintes atributos: `proprietario`, `saldo`, `moeda` (por exemplo: `"EUR"`, `"USD"`) e `ativa` (boolean).
- Cria um método `"enviarDinheiro"` que tenha como parâmetros o valor a enviar e a carteira de destino. Este método só funciona se ambas as carteiras estiverem ativas e usarem a mesma moeda.
- Cria os métodos `adicionarSaldo` (aumentar saldo), `gastar` (diminuir saldo) e `consultarSaldo`. (Exemplo de `consultarSaldo`: `"Carteira de João Silva: 450.50 EUR"`)
- O método `adicionarSaldo` deve receber um valor como parâmetro e atualizar o saldo.
- O método `gastar` deve receber um valor como parâmetro e verificar se é possível realizar a operação, considerando o saldo disponível. De seguida, atualiza o saldo se a transação for possível.
- O método `consultarSaldo` deve exibir na consola o saldo atual da carteira de forma formatada.
- No método `main`, cria um objeto da classe `CarteiraDigital` e invoca os métodos para adicionar 500€ e consultar o saldo.
- De seguida, invoca o método para gastar 75€ e exibe o saldo.
- De seguida, cria mais dois objetos da classe e atribui valores aos seus atributos. Depois, envia 100€ de uma carteira para a outra.
- No fim, deve exibir o saldo atual das três carteiras.

12. Cria um programa para gerir uma biblioteca de jogos:

- Cria uma classe chamada `"Jogo"` com os seguintes atributos: `titulo`, `plataforma` (exemplo: `"PC"`, `"PlayStation"`, `"Xbox"`), `preco`, `horasJogadas` e `classificacao` (de 0 a 10).
- Cria um construtor que receba `titulo`, `plataforma` e `preço` como parâmetros, e inicialize `horasJogadas` a 0 e `classificacao` a 0.
- Cria um método `"jogar"` que receba as horas jogadas como parâmetro e adicione ao total de horas.
- Cria um método `"avaliar"` que receba uma classificação (0-10) e a atribua ao jogo.
- Cria um método `"calcularValorPorHora"` que retorne o preço dividido pelas horas jogadas (mostra quanto custou cada hora de jogo). Se ainda não tiver jogado nenhuma hora, deve retornar uma mensagem apropriada.
- Cria um método `"mostrarDetalhes"` que exibe toda a informação do jogo formatada, incluindo o valor por hora se aplicável.
- No método `main`, cria três objetos da classe `Jogo` com jogos diferentes.
- Simula sessões de jogo (invoca o método `jogar` várias vezes para cada jogo com valores diferentes).

-
- Avalia cada jogo com classificações diferentes.
 - No final, mostra os detalhes de todos os jogos e identifica qual teve o melhor custo-benefício (menor valor por hora jogada).

Bom trabalho! ☺