Lista de Exercícios

# Arquivos Texto

1. Escreva um programa Java que crie um arquivo de texto chamado "meuarquivo.txt" e escreva a mensagem "Olá, mundo!" nele.
2. Crie um programa Java que leia o arquivo "meuarquivo.txt" criado na pergunta anterior e exiba seu conteúdo no console.
3. Modifique o programa anterior para adicionar a mensagem "Isso é uma adição!" ao final do arquivo "meuarquivo.txt" sem apagar o conteúdo existente.
4. Escreva um programa Java que conte quantas linhas existem no arquivo "meuarquivo.txt" e exiba esse número no console.
5. Crie um programa Java que leia o conteúdo do arquivo "meuarquivo.txt", substitua todas as ocorrências da palavra "Java" pela palavra "Python" e salve o resultado em um novo arquivo chamado "meuarquivo\_python.txt".
6. Implemente um programa Java que leia um arquivo CSV chamado "dados.csv" (com valores separados por vírgula) e exiba cada linha no console, dividida em campos.
7. Escreva um programa Java que leia o arquivo "meuarquivo.txt", ordene as linhas em ordem alfabética e salve o resultado em um novo arquivo chamado "meuarquivo\_ordenado.txt".
8. Implemente um programa Java que leia o arquivo "meuarquivo.txt" e remova todas as linhas que contêm a palavra "excluir". Salve o resultado em um novo arquivo chamado "meuarquivo\_sem\_excluir.txt".
9. Crie um programa Java que leia o arquivo "meuarquivo.txt" e conte quantas vezes a palavra "Java" aparece no texto. Exiba o resultado no console.
10. Escreva um programa Java que crie dois arquivos de texto ("arquivo1.txt" e "arquivo2.txt") com algum conteúdo. Em seguida, crie um terceiro arquivo chamado "arquivo\_concatenado.txt" que contenha o conteúdo dos dois primeiros arquivos concatenados.

# Arquivos Binários

1. Escreva um programa Java que serializa um objeto da classe Pessoa em um arquivo binário chamado "pessoa.dat".
2. Modifique o programa anterior para deserializar o objeto do arquivo "pessoa.dat" e exibir os detalhes da pessoa no console.
3. Crie um programa Java que serialize uma lista de objetos da classe Produto em um arquivo binário chamado "produtos.dat" e, em seguida, deserialize a lista e exiba todos os produtos no console.
4. Implemente um programa que carregue um objeto serializado da classe ContaBancaria do arquivo "conta.dat", atualize o saldo e, em seguida, serialize o objeto atualizado de volta ao arquivo.
5. Desenvolva um programa que copie um arquivo binário grande chamado "arquivo\_grande.bin" para outro arquivo chamado "copia\_arquivo\_grande.bin" usando buffers para melhorar o desempenho.
6. Escreva um programa que leia os primeiros 100 bytes de um arquivo binário chamado "arquivo.bin" e os exiba no console.
7. Implemente um programa que crie um arquivo binário personalizado contendo informações estruturadas, como registros de funcionários, e permita adicionar, atualizar e excluir registros no arquivo.

# Arquivos CSV

1. Crie um programa Java que use a biblioteca OpenCSV para ler um arquivo CSV chamado "alunos.csv" e exibir o conteúdo no console. O arquivo CSV contém informações sobre alunos, como nome, idade e nota.

Dados do arquivo alunos.csv:

Nome,Idade,Nota

Alice,22,85

Bob,21,78

Carol,23,92

David,20,67

Eva,22,95

1. Desenvolva um programa Java que use a biblioteca OpenCSV para criar um arquivo CSV chamado "funcionarios.csv" e escrever informações sobre funcionários nele. Peça ao usuário que insira os detalhes dos funcionários, como nome, cargo e salário, e adicione esses dados ao arquivo CSV.
2. Crie uma classe Java chamada `Produto` que represente um produto com campos como nome, preço e quantidade em estoque. Em seguida, escreva um programa que use a biblioteca OpenCSV para ler um arquivo CSV chamado "produtos.csv" que contém informações sobre produtos e armazenar esses dados em objetos da classe `Produto`. Exiba os detalhes dos produtos no console.

Dados do arquivo produtos.csv:

Nome,Preço,Quantidade

Notebook,1500,10

Smartphone,800,20

Tablet,500,15

Mouse,30,50

Teclado,40,40

1. Modifique o programa da Questão 20 para permitir que o usuário insira detalhes de produtos, como nome, preço e quantidade, e adicione esses produtos ao arquivo CSV "produtos.csv" usando a biblioteca OpenCSV.
2. Aprimore o programa da Questão 20 para permitir que o usuário atualize informações de produtos existentes no arquivo "produtos.csv". Permita que o usuário pesquise um produto pelo nome e, em seguida, atualize os campos, como preço ou quantidade em estoque, desse produto no arquivo CSV.
3. Adicione a funcionalidade de exclusão de produtos no programa da Questão 20. Permita que o usuário pesquise um produto pelo nome e, em seguida, exclua-o do arquivo CSV "produtos.csv". Certifique-se de atualizar o arquivo CSV após a exclusão.
4. Crie um programa Java que use a biblioteca OpenCSV para ler um arquivo CSV chamado "funcionarios.csv" contendo informações sobre funcionários, como nome, cargo e salário. Permita que o usuário filtre os funcionários com base em critérios, como cargo ou salário mínimo, e exiba os resultados no console.

Dados para o arquivo funcionarios.csv:

Nome,Cargo,Salário

João,Servidor,3000

Maria,Gerente,5000

Carlos,Desenvolvedor,4500

Lucia,Analista,4200

Marcos,Designer,3800

1. Desenvolva um programa que crie uma lista de objetos da classe `Venda`, onde cada objeto representa uma venda com informações como data, valor e produto vendido. Use a biblioteca OpenCSV para exportar essas vendas para um arquivo CSV chamado "vendas.csv".

# Banco de Dados

1. Escreva um programa Java que utilize o JDBC para se conectar a um banco de dados MariaDB local. Certifique-se de configurar corretamente a URL de conexão, nome de usuário e senha. Após a conexão bem-sucedida, exiba uma mensagem de confirmação no console.
2. Crie um programa Java que insira dados em uma tabela chamada "clientes" no banco de dados MariaDB. Solicite ao usuário que insira informações sobre um novo cliente, como nome, idade e endereço, e insira esses dados na tabela.
3. Desenvolva um programa Java que realize uma consulta SQL para recuperar todos os registros da tabela "produtos" no banco de dados MariaDB. Exiba os resultados no console.
4. Modifique o programa da Questão 22 para permitir que o usuário atualize informações de um cliente existente na tabela "clientes". Peça ao usuário que insira o ID do cliente a ser atualizado e, em seguida, atualize os campos, como endereço ou número de telefone.
5. Crie um programa Java que permita ao usuário excluir um cliente da tabela "clientes" no banco de dados MariaDB. Solicite ao usuário que forneça o ID do cliente a ser excluído e remova-o da tabela.
6. Escreva um programa Java que realize uma consulta SQL na tabela "pedidos" para recuperar todos os pedidos feitos por um cliente específico. Solicite ao usuário que insira o ID do cliente como parâmetro e exiba os resultados no console.
7. Garanta que, em todos os programas anteriores, você encerre a conexão com o banco de dados MariaDB adequadamente após a conclusão das operações. Use a estrutura try-with-resources ou um bloco finally para garantir o fechamento da conexão.