Exercícios de Programação Lua

Exercícios de Programação Nível Intermediário a Pré-Avançado: Estruturas Compostas

Exercício 1 ----------

Crie uma tabela chamada *aluno* que armazena as seguintes informações de um estudante: nome, idade e curso. Imprima cada uma dessas informações no console.

Exercício 2 ----------

Crie um array que armazene as notas de um aluno em cinco disciplinas. Calcule e imprima a média das notas.

Exercício 3 ----------

Implemente uma função chamada *somaElementos* que recebe uma tabela de números e retorna a soma de todos os elementos.

Exercício 4 ----------

Crie uma função que aceita múltiplos argumentos e retorna o produto de todos eles.

Exercício 5 ----------

Escreva uma função que divide dois números. Utilize variáveis locais e controle o escopo da função para evitar acessos externos.

Exercício 6 ----------

Crie uma função que recebe uma tabela e uma função de filtro. A função filtra deve retornar uma nova tabela contendo apenas os elementos que satisfazem a condição definida pela função de filtro.

Ex: Use para encontrar apenas os números pares de uma tabela.

Exercício 7 ----------

Implemente uma fila de atendimento em que novos clientes são adicionados à fila e o próximo cliente é atendido em ordem de chegada. Crie funções *adicionaCliente* e *atendeCliente*.

Exercício 8 ----------

Usando tabelas e recursão, crie uma estrutura que represente uma árvore genealógica simples. A tabela pessoa deve conter o nome da pessoa e referências a seus pais, que também são tabelas. Implemente uma função *imprimeFamilia* que percorre a árvore genealógica e imprime o nome da pessoa e dos seus pais.

Exercício 9 ----------

Crie uma função *adivinhaNumero* que recebe um número secreto e múltiplos palpites como argumentos. A função deve retornar true se algum dos palpites for igual ao número secreto; caso contrário, false.

Exercício 10 ----------

Implemente um simulador de loja em que clientes realizam compras. Use uma corrotina para cada cliente, com cada corrotina adicionando um item ao carrinho de compras a cada segundo. A simulação termina quando o cliente adiciona três itens.

Exercício 11 ----------

Crie uma função *safeLog* que tenta abrir um arquivo e escrever uma mensagem de log. Se ocorrer um erro (como permissão negada), capture-o e retorne uma mensagem de erro amigável.