Curso de Linguagem C

Lista 2 - Operadores e Estruturas de Decisão

PET EEL

1º semestre de 2024

Exercícios

1. **Calculadora Simples** Implemente uma calculadora que recebe dois números inteiros e um operador (`+`, `-`, `*`, `/`, `%`), e realiza a operação indicada entre os números. Use estruturas de decisão `if` e `else if` para definir cada operação e verifique a divisão por zero.

2. **Verificador de Tensão Segura** Crie um programa que recebe um valor de tensão (em volts) e verifica se ele está dentro da faixa segura para um componente eletrônico específico (faixa segura: 3V a 5V). Use operadores relacionais e estruturas de decisão `if` para mostrar uma mensagem se a tensão está segura ou fora do limite.

3. **Verificação de Paridade de Bits** Escreva um programa que recebe um número inteiro e utiliza operadores *bit a bit* para verificar se ele tem um número par ou ímpar de bits definidos para 1. Use operadores como `&` e `^` para fazer a contagem e exiba uma mensagem com o resultado.

4. **Controlador de Motor com Níveis de Potência** Crie um programa que usa `switch-case` para configurar o nível de potência de um motor baseado em um valor de entrada entre 1 a 5 (onde 1 = baixa potência e 5 = potência máxima). Exiba uma mensagem para cada nível e use `default` para valores fora do intervalo. Utilize operadores de atribuição para simular o aumento de potência em cada nível.

5. **Comparador de Resistores** Receba dois valores de resistores (em ohms) e compare-os usando operadores relacionais. Exiba uma mensagem informando qual resistor tem maior resistência ou se são iguais.

6. **Ajuste de Brilho de uma Lâmpada** Crie um programa que simula o ajuste de brilho de uma lâmpada usando operadores *bit a bit* (`<<` e `>>`) para dobrar ou reduzir a intensidade do brilho (valor de uma variável `brilho`). Use operadores de atribuição para multiplicar e dividir o valor conforme o ajuste e exiba o valor final do brilho.

7. **Detector de Sobrecarga em Circuito** Escreva um programa que recebe a corrente e a tensão de um circuito, calcula a potência usando o operador aritmético `*` e verifica se a potência ultrapassa o limite de segurança (por exemplo, 100W). Exiba uma mensagem se o circuito está sobrecarregado ou operando normalmente, usando `if-else`.

8. **Conversor de Código Binário para Decimal** Receba um número binário de 4 bits (entre `0000` e `1111`) e converta-o para decimal usando operadores *bit a bit* e operadores aritméticos. Exiba o número decimal correspondente.

9. **Cálculo do Fator de Potência** Receba dois valores de potência: potência real (W) e potência aparente (VA), e calcule o fator de potência (FP) usando a fórmula `FP = W / VA`. Use operadores relacionais para verificar se o FP está na faixa aceitável (0.8 a 1) e exiba uma mensagem se o fator está adequado.

10. **Decodificação de Sinal usando Condicional** Crie um programa que recebe o valor de um sinal (`-1`, `0`, ou `1`). Use o operador ternário `?:` para exibir mensagens com base no valor do sinal: `-1` para "Sinal negativo", `0` para "Sinal nulo" e `1` para "Sinal positivo".

Dicas

- **Calculadora Simples** Use o `switch-case` ou uma série de `if-else` para comparar o operador recebido. Lembre-se de tratar a divisão (`/`) para evitar dividir por zero, usando um `if` para verificar quando o divisor é zero antes de calcular.
- **Verificador de Tensão Segura** Utilize operadores relacionais (`>=`, `<=`) para verificar se a tensão está na faixa segura. Combine duas condições usando `&&` para garantir que o valor esteja entre os limites inferiores e superiores.
- **Verificação de Paridade de Bits** Use operadores *bit a bit* (`&`, `>>`) para iterar sobre cada bit e verificar se ele está definido como 1. Lembre-se de que `num & 1` verifica o bit menos significativo; depois, use o deslocamento (`>>`) para avançar bit a bit.
- **Controlador de Motor com Níveis de Potência** Use `switch-case` para cada nível de potência e `break` para sair após encontrar o valor correto. Use `default` para casos fora da faixa (1-5), exibindo uma mensagem de "nível inválido".
- **Comparador de Resistores** Para comparar resistores, use os operadores relacionais (`>`, `<`, `==`). Assegure-se de que as mensagens exibidas deixam claro qual resistor é o maior, ou se são iguais.
- **Ajuste de Brilho de uma Lâmpada** Use `<<` para dobrar e `>>` para reduzir pela metade o valor de `brilho`. Esses operadores são ideais para manipular o brilho quando ele é uma potência de dois.
- **Detector de Sobrecarga em Circuito** Calcule a potência como `potencia = corrente * tensao`. Em seguida, use uma estrutura `if-else` para verificar se a potência ultrapassa o limite e exibir a mensagem apropriada.
- **Conversor de Código Binário para Decimal** Use operadores de deslocamento *bit a bit* para processar cada bit do código binário. Por exemplo, o valor mais à esquerda em um número de 4 bits (1000) representa 8 em decimal, o seguinte (0100) representa 4, e assim por diante.
- **Cálculo do Fator de Potência** Calcule o fator de potência dividindo `potencia_real` por `potencia_aparente`. Use `if-else` para verificar se o resultado está dentro da faixa aceitável (0.8 a 1).
- **Decodificação de Sinal usando Condicional** Utilize o operador condicional ternário (`?:`) para definir a mensagem com base no valor do sinal. Uma estrutura como `(sinal == -1) ? "Sinal negativo": (sinal == 0 ? "Sinal nulo": "Sinal positivo")` pode ajudar a simplificar o código.