

Aula 11: Bits: XOR, AND e deslocamento

Lógica de Programação – Superior · Bloco 1

José Dihego · IFBA

O que você precisa saber hoje

Objetivo: Manipular os bits de um inteiro com $\wedge$, $\&$ e $\gg$.

- $a \wedge b$ (XOR): bit 1 onde a e b diferem
- $x \& 1$ é o último bit; $x \gg k$ desloca k casas para a direita
- Bit k de x : $(x \gg k) \& 1$
- Inverter o bit k : $x \wedge (1 \ll k)$
- Equivalente sem bits: $\% 2$ e $/ 2$

```
int x = a ^ b;           // bits diferentes
int dif = 0;
for (int k = 0; k < 4; k++)
    dif += (x >> k) & 1; // conta os 1
```

Erros que custam pontos

Errado	Certo	Por quê
<code>if (x & 1 == 1)</code>	<code>if ((x & 1) == 1)</code>	<code>==</code> tem prioridade maior que <code>&</code>
<code>a ^ 2 /* "a ao quadrado" */</code>	<code>a * a</code>	<code>^</code> é XOR, não potência

```
int a = 9, b = 3, x = a ^ b, dif = 0;
for (int k = 0; k < 4; k++) {
    dif += (x >> k) & 1;
}
```

k	(x » k) & 1	dif
0	0	0
1	1	1
2	0	1
3	1	2

Avaliação 1 – 2026.1, Questão II

avaliacao_1_ano_2026_1.pdf

Números n -binariamente distintos

Estratégia:

1. XOR marca os bits diferentes entre a e b
2. Contar os bits 1 do XOR (4 bits)
3. Comparar a contagem com n

Enunciado completo (com figuras e exemplos) no PDF da prova.

Solução

```
1  #include <stdio.h>                                // printf e scanf
2
3  int main(void) {                                    // início do programa
4      int n, a, b;                                    // limiar n e os dois números (menores que 16)
5      printf("n, a e b: ");                          // pede os dados
6      scanf("%d %d %d", &n, &a, &b);                  // lê os três inteiros
7      int x = a ^ b;                                  // OU exclusivo: bit 1 onde a e b diferem
8      int diferentes = (x & 1) + ((x >> 1) & 1) + ((x >> 2) & 1) + ((x >> 3) & 1); // conta os 4
                                                bits
9      if (diferentes >= n) printf("%d e %d são %d-binariamente distintos\n", a, b, n); // bits
                                                diferentes suficientes
10     else printf("%d e %d não são %d-binariamente distintos\n", a, b, n); // menos bits
                                                diferentes que n
11     return 0;                                       // fim sem erro
12 }
```

Avaliação 1 – 2023.2, Questão II

avaliacao_1_ano_2023_2.pdf

Mutação por swap (inverter um bit)

Dicas:

- Inverter um bit: XOR com $1 \ll k$
- Cuidado com a numeração das posições (da direita, a partir de 0 ou 1?)
- Imprima em binário para conferir

Para conferir: entrada 28 3 $\rightarrow$ a saída deve conter 20

A solução comentada está na página do curso: <https://josedihego.net/logica/#aula-11>