

**Porque os índices
de array
começam com
0 (zero)?**



A escolha de começar os índices de um array em 0 em muitas linguagens de programação, incluindo Java tem raízes históricas e práticas.

Existem algumas razões para isso:

Convenção e consistência:

Muitas linguagens de programação, incluindo C, C++, Java, GO, Javascript, PHP e Python, adotaram a convenção de começar os índices de arrays em 0. Isso cria consistência entre diferentes linguagens e facilita a interoperabilidade e o entendimento do código.



Matemática e algoritmos:

Muitos algoritmos e fórmulas matemáticas são naturalmente baseados em índices começando em 0. Por exemplo, em muitas situações, é mais conveniente calcular índices como $i + 1$ ou $i - 1$, e começar em 0 facilita esse tipo de manipulação.

Eficiência de memória e desempenho:

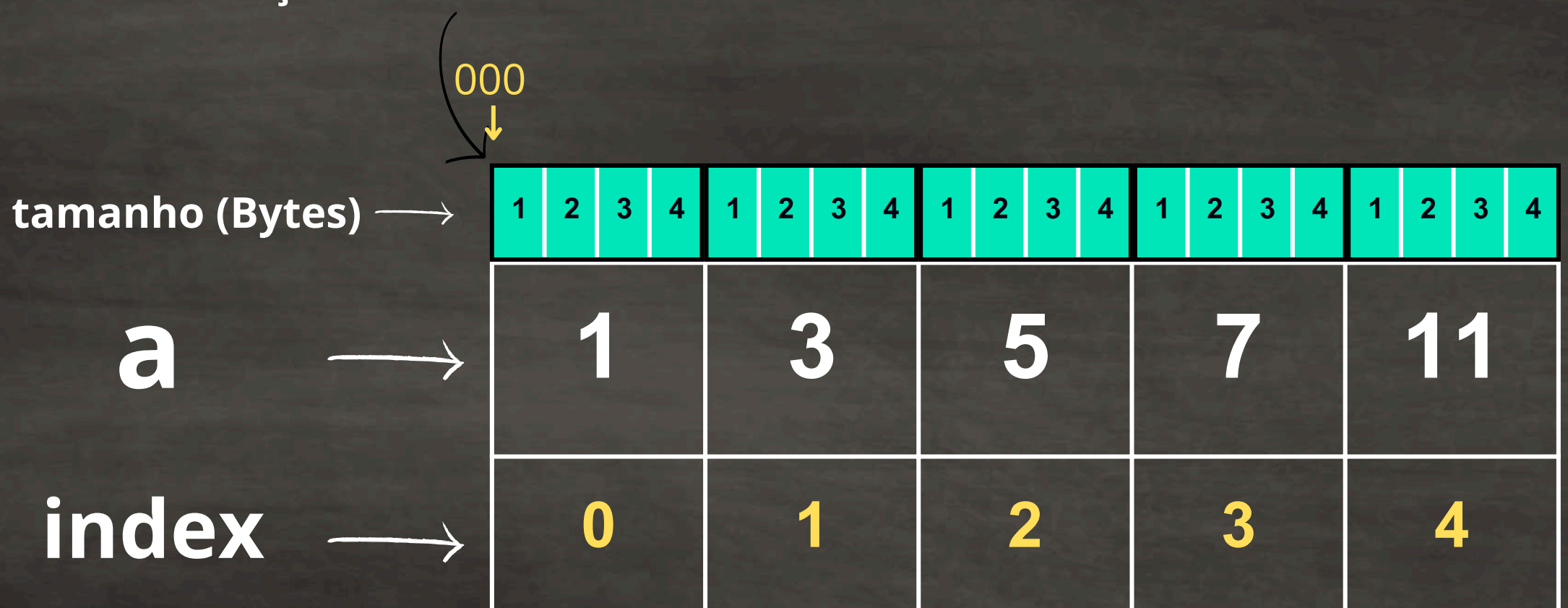
Começar os índices em 0 simplifica o cálculo do deslocamento de memória para acessar um elemento específico do array. Isso pode resultar em código mais eficiente e um melhor desempenho em operações de acesso a elementos do array.



Vamos iniciar um array de inteiros
(em java)

```
int[] a = new int[5];
```

Posição inicial da memória = índice 0



O índice do array começa em 0 e termina no 4
(uma vez que tem cinco elementos).

Cada elemento tem tamanho de 4 Bytes (que é o espaço de memória alocado dentro do Java para elementos do tipo int)



Como a memória se localiza ?
para exemplificar e simplificar supomos que a
posição inicial da memória é 700.



$$a[\text{<índice>}] = \text{<posição inicial do array>} + \text{<tamanho do elemento>} * \text{<índice>}$$

$$a[0] = 700 + 4 * 0 = 700$$

$$a[1] = 700 + 4 * 1 = 704$$

$$a[2] = 700 + 4 * 2 = 708$$

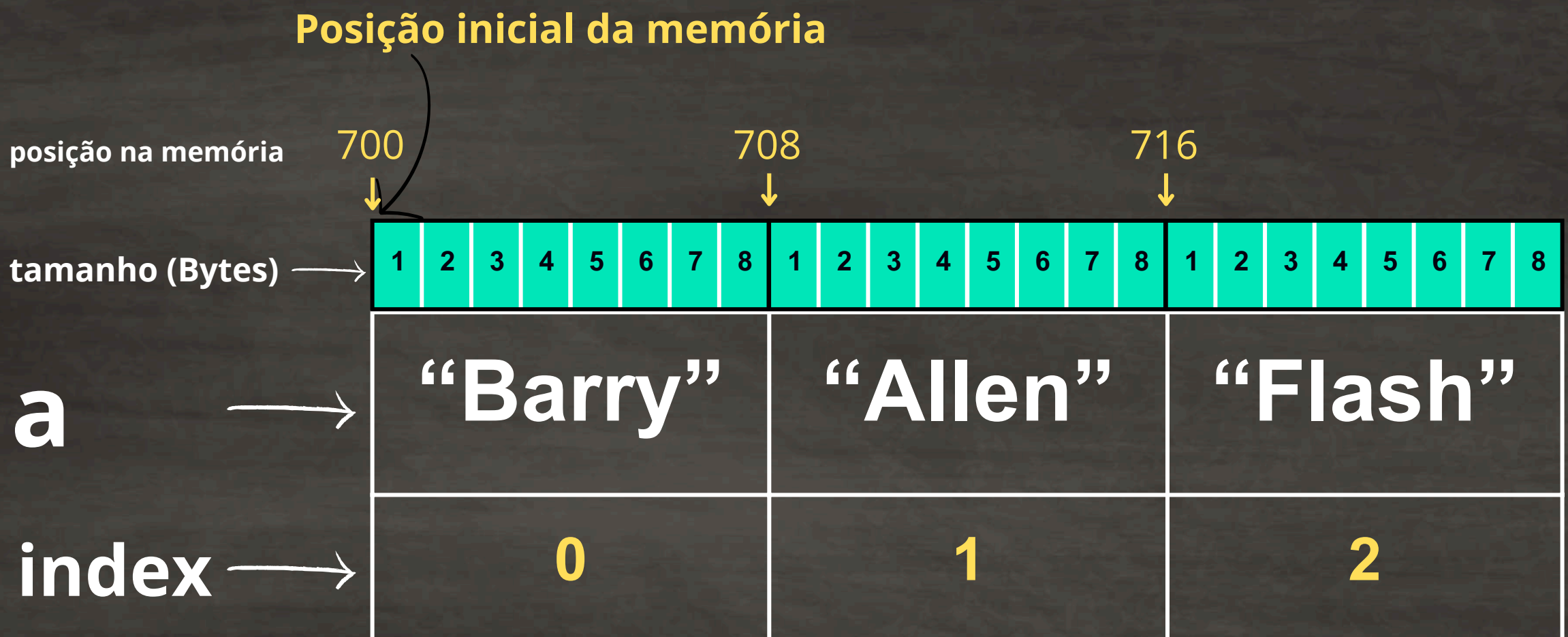
$$a[3] = 700 + 4 * 3 = 712$$

$$a[4] = 700 + 4 * 4 = 716$$



String[] a = new String[3];

(cada elemento tem tamanho de 8 Bytes)



$a[\text{<índice>}] = \text{<posição inicial do array>} + \text{<tamanho do elemento>} * \text{<índice>}$

$$a[0] = 700 + 8 * 0 = 700$$

$$a[1] = 700 + 8 * 1 = 708$$

$$a[2] = 700 + 8 * 2 = 716$$



valores de memória são frequentemente representados em hexadecimal, especialmente em contextos de programação e computação de baixo nível. Isso ocorre porque a representação hexadecimal é mais compacta que a representação decimal e facilita a visualização e manipulação de endereços de memória, valores de bytes e muitos outros aspectos relacionados à computação.

O formato de armazenamento interno da memória ainda é binário. O hexadecimal é apenas uma maneira de representar esses valores de forma legível para humanos.



Embora a escolha de começar os índices de array em 0 possa parecer estranha para algumas pessoas inicialmente, ela se torna natural com o tempo e oferece várias vantagens incluindo simplicidade, consistência, eficiência de memória, compatibilidade matemática, padronização e convenção.

