

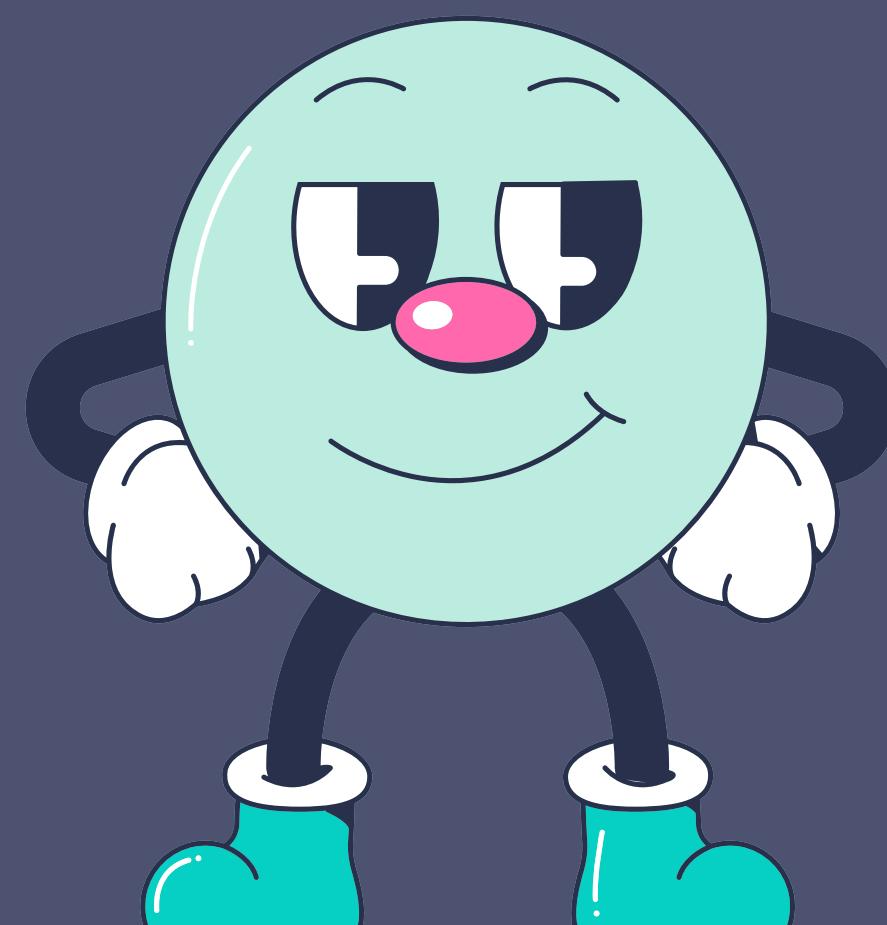
Hashing



MONITORIA DE ALGORITMOS

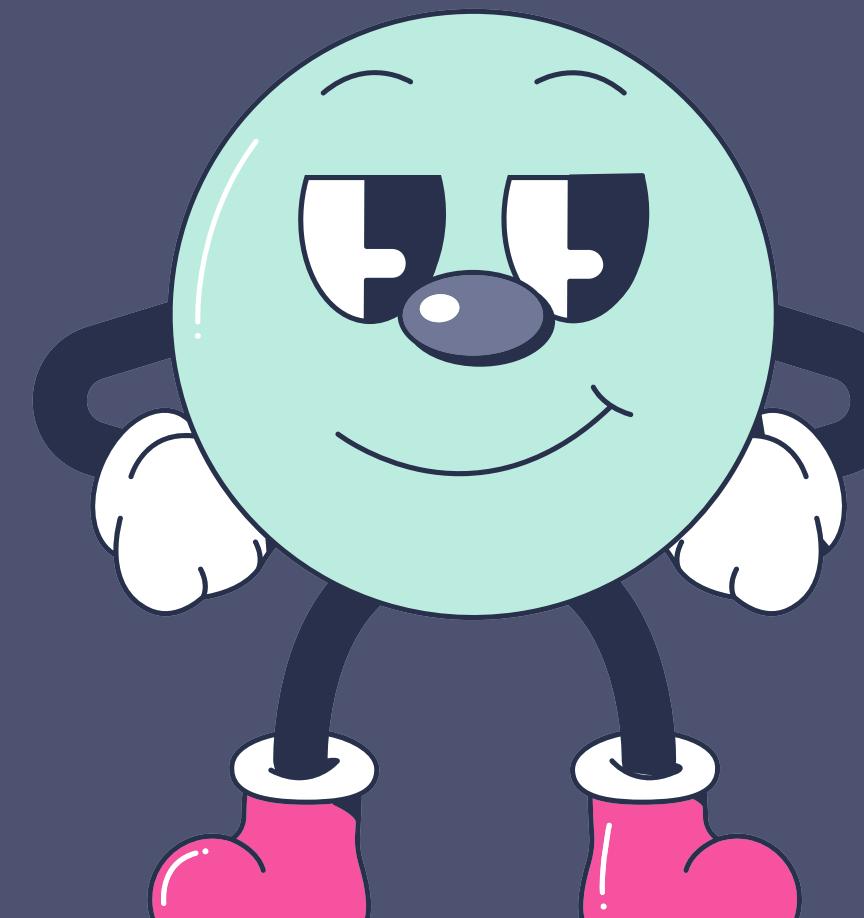
Hash

É uma função que associa a um valor dado uma chave única.



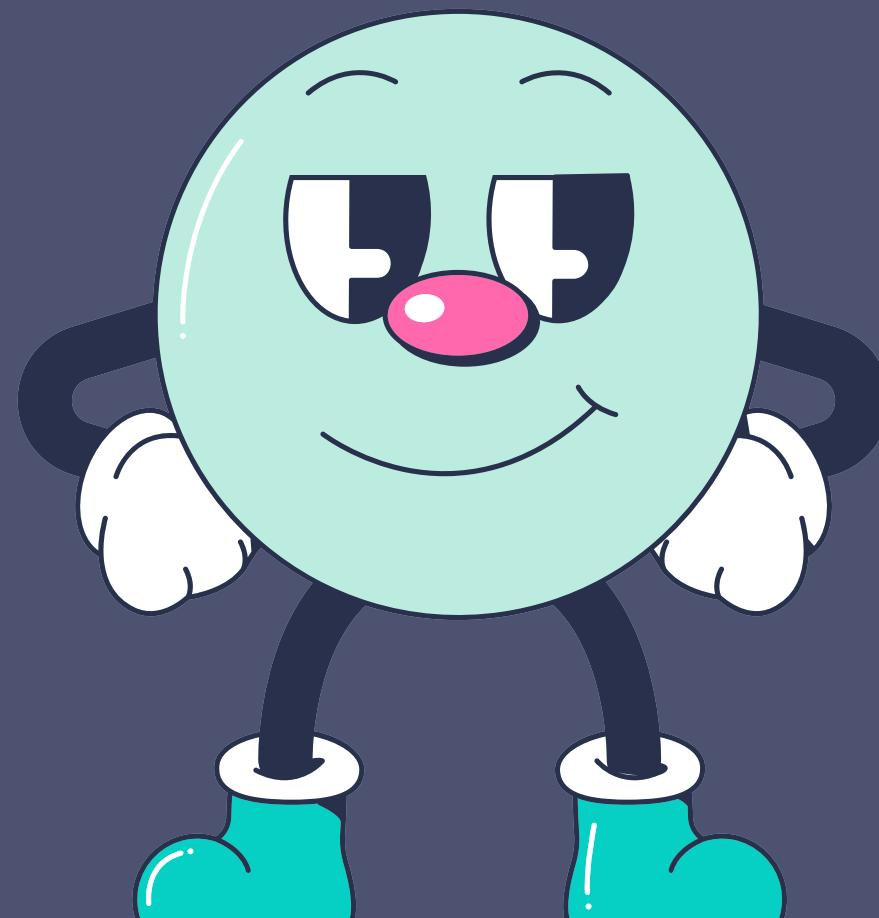
```
int h(string key){  
    int sum=0;  
    for(int i = 0;i<key.size();i++){  
        sum += 42 * key[i] * (i+1);  
    }  
    return sum % 200;  
}
```

Hash table



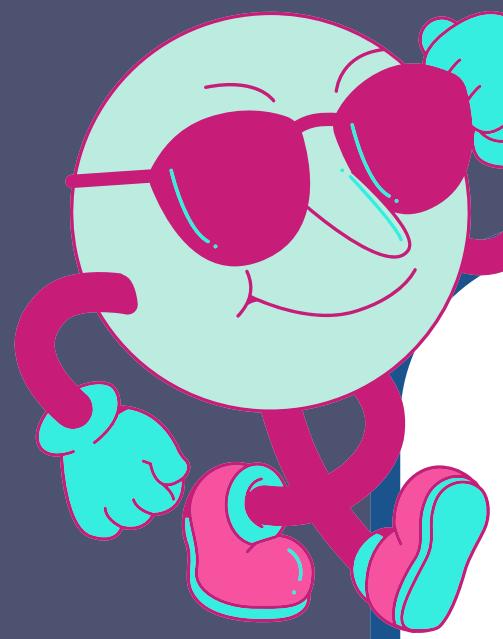
- 1 Open Hashing
- 2 Closed Hashing

Open Hashing

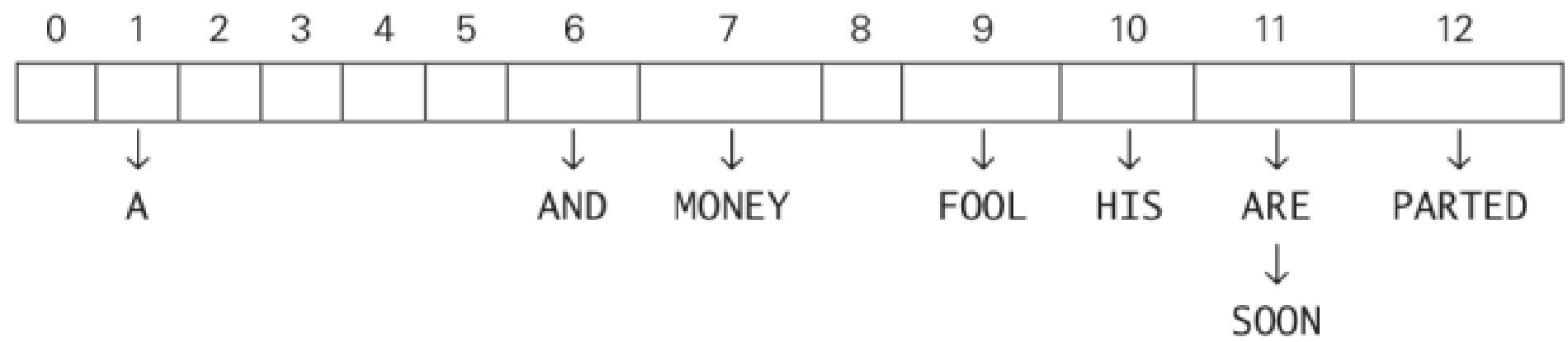


Cada endereço da Hash Table
está associado a uma lista ligada.

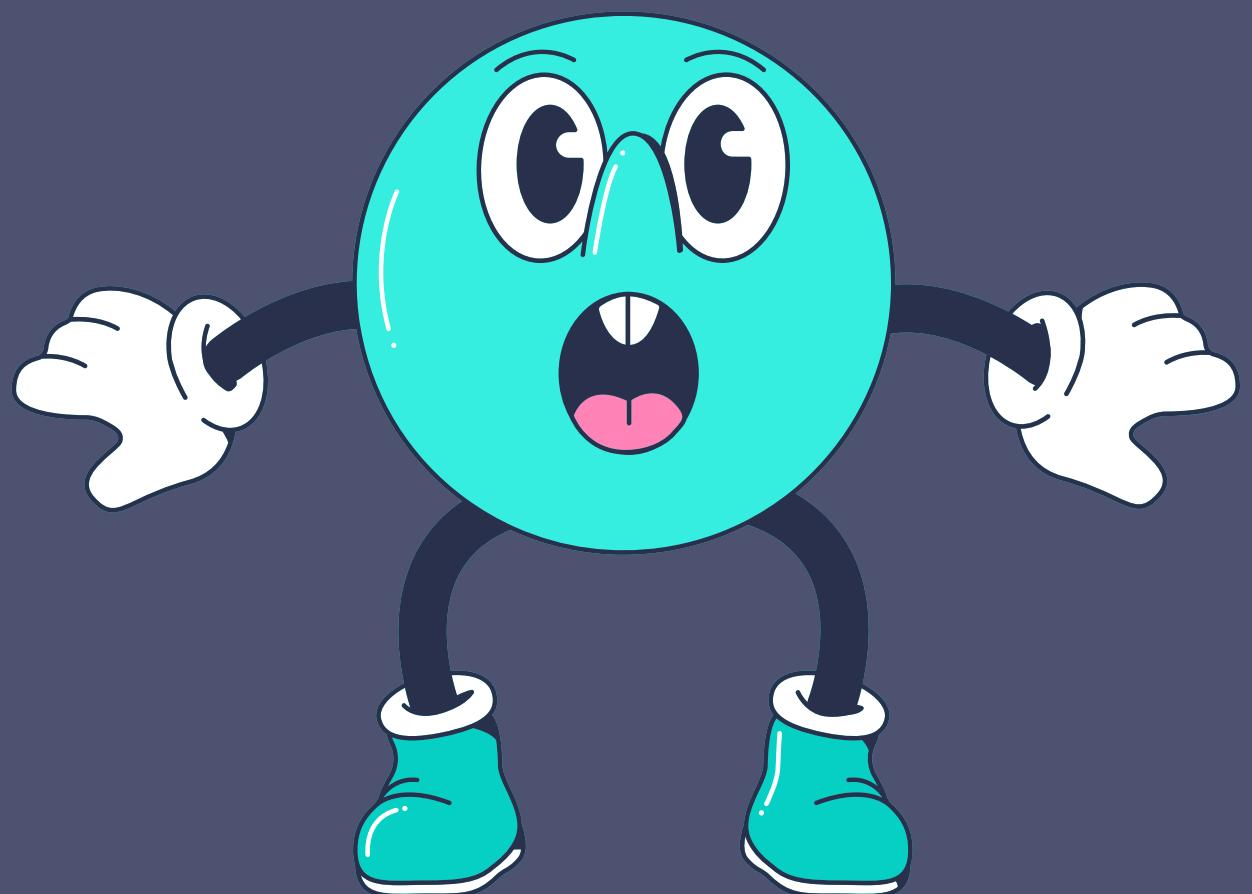
- Não lida com colisões
- Busca se torna linear



keys	A	FOOL	AND	HIS	MONEY	ARE	SOON	PARTED
hash addresses	1	9	6	10	7	11	11	12



Closed Hashing



Cada endereço da HashTable só pode conter um elemento.

Com isso agora teremos que lidar com possíveis colisões entre os elementos.

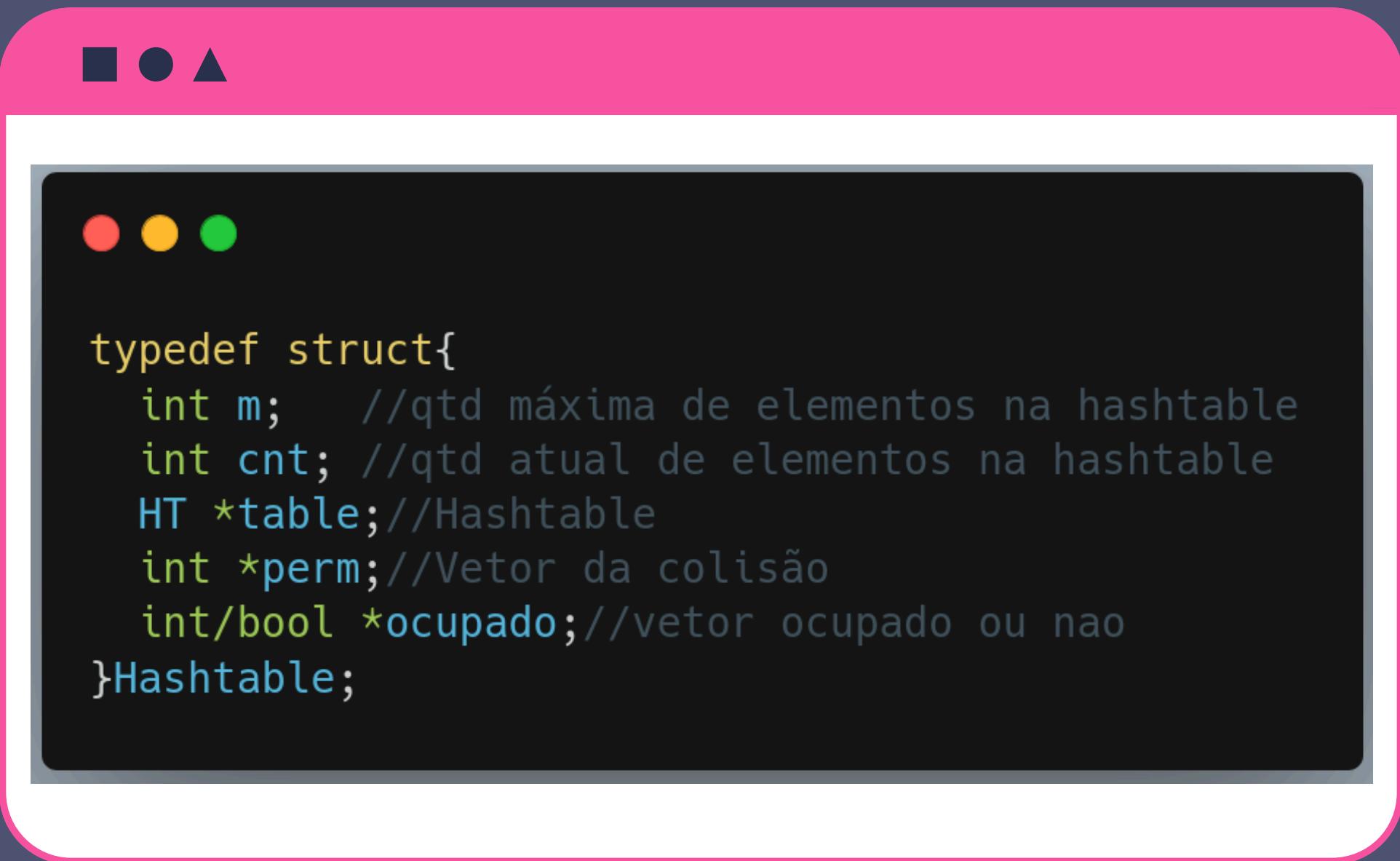
- Busca eficiente

Formas de lidar com colisões



- Linear Probing
- Pseudo-random Probing
- Quadratic Probing

implementação



implementação

Para você criar uma hashtable são necessárias quatro funções básicas: init, find, insert e delete.

Na função init você deve alokar o espaço necessário para os vetores que estão na struct Hashtable como também especificar o M e zerar o contador.

No Find você utilizará a chave recebida para procurar o índice do elemento. E retorna o índice dela caso ache, se não retorna -1.

implementação

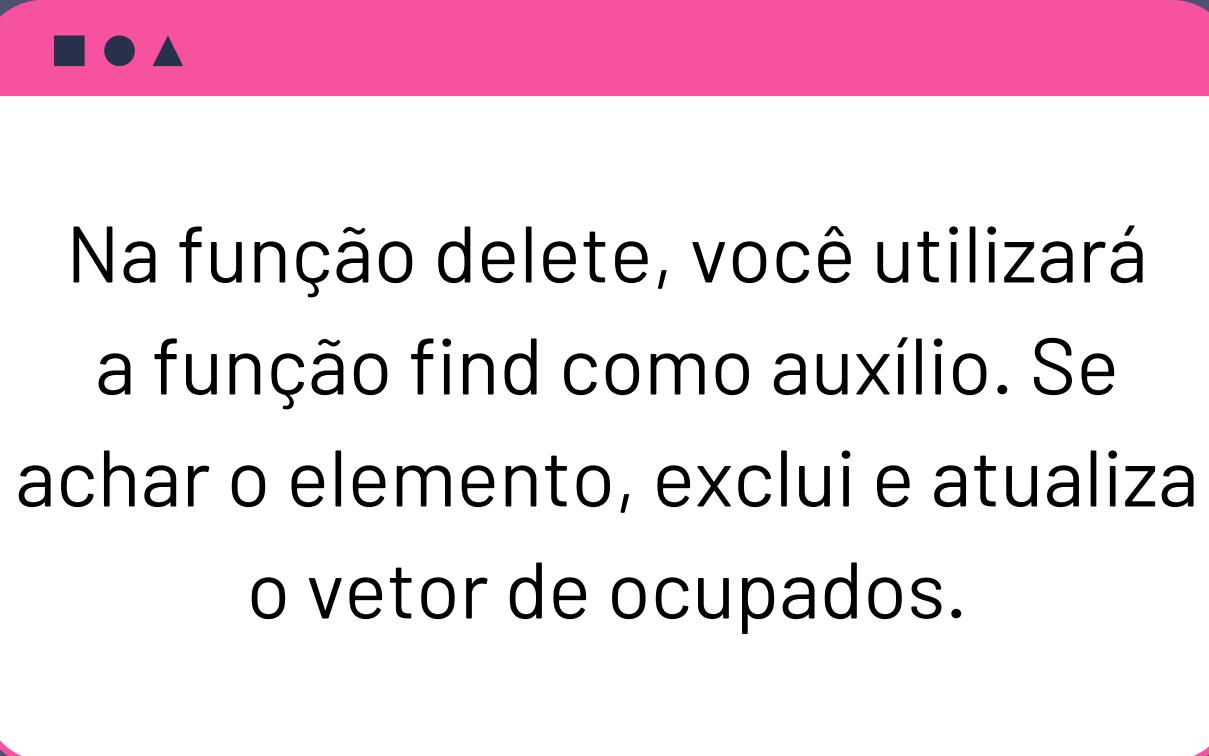
Para você criar uma hashtable são necessárias quatro funções básicas: init, find, insert e delete.

Na função insert, você utilizará a função find para verificar primeiro se o elemento já está na tabela. Caso esteja você não insere.

Após essa verificação você calculará a hash da chave que vai ser adicionada e tentará adicionar. Caso a posição esteja ocupada você utiliza do sistema de colisão.

implementação

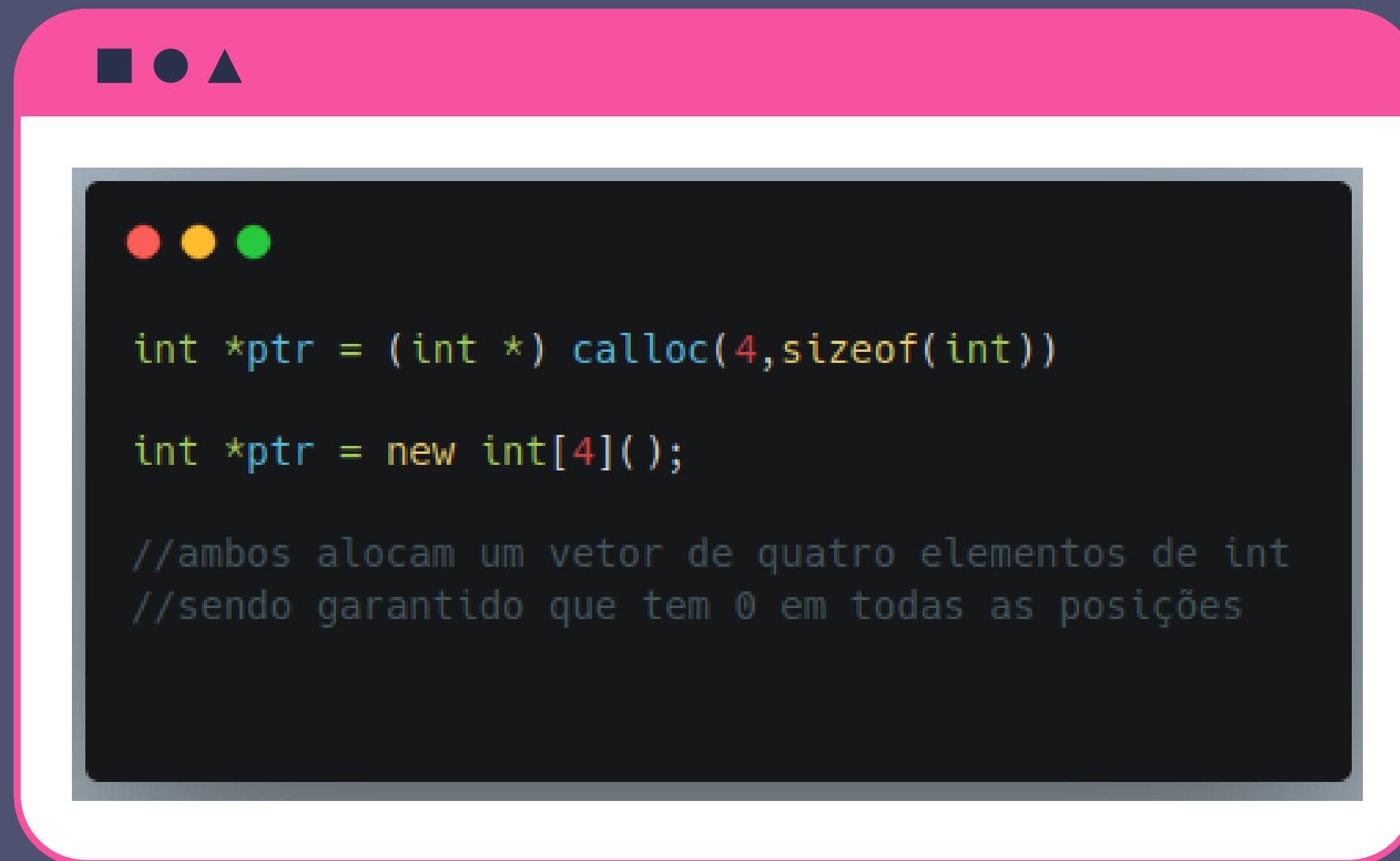
Para você criar uma hashtable são necessárias quatro funções básicas: init, find, insert e delete.



Na função delete, você utilizará a função find como auxílio. Se achar o elemento, exclui e atualiza o vetor de ocupados.

implementação

Para alocar um vetor com todas as posições preenchidas com zero.



Obrigado!

