Tratamento de colisão

segunda-feira, 11 de novembro de 2024 11:51

↳ idealmente, a função deve ter 1 saída diferente p/ qualquer entrada (chave)

O que são colisões?

↳ mapeia os índices da tabela Hash

São quando uma função hash aponta p/ diferentes "inputs" os mesmos "outputs" (hashs).

Ex: Entrada -> "maçã"
Saída -> Endereço de memória 0

Entrada -> "banana"
Saída -> Endereço de memória 0

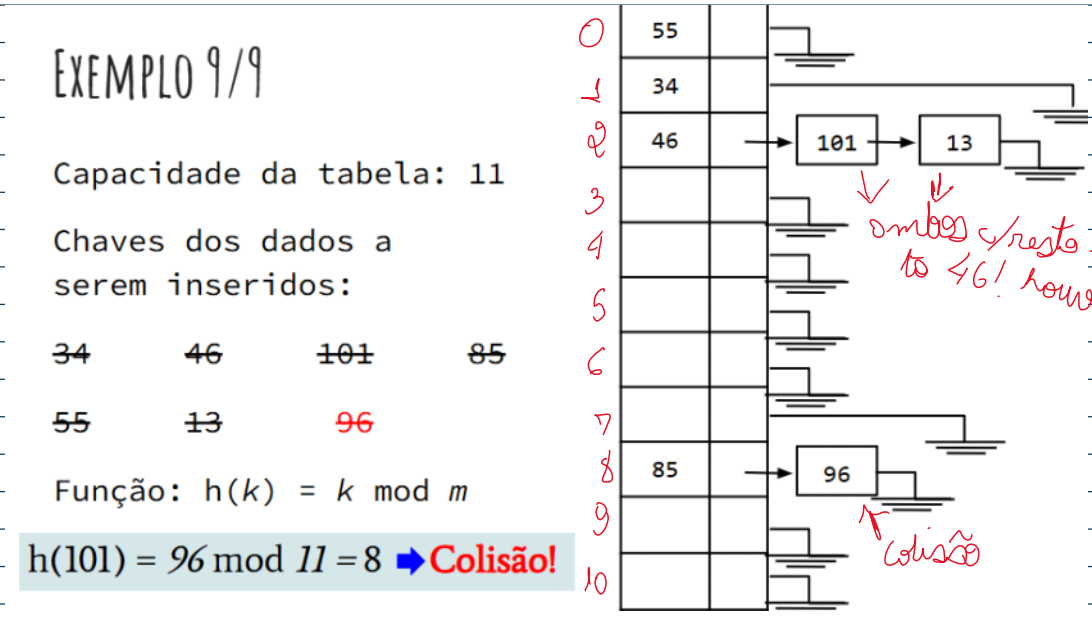
↳ Isso causa problemas! Como podem dois elementos ≠s serem armazenados no exato mesmo lugar?

Formas de se lidar c/ colisão:

1 - Encadeamento

Se ocorrer uma colisão, armazenaremos o novo elemento endereçado à posição junto a ele em uma lista encadeada.

Podemos, também, implementar a hash como um vetor de listas encadeadas.



Buscamos ter ∠ 70% de ocupação, p/ evitarmos degeneração da tabela hash.

2 - Endereçamento aberto

Se a função hash mapear p/ um índice já ocupado procuraremos após uma posição livre p/ armazenar o dado.

EXEMPLO 4/5

Capacidade da tabela: 7

Chaves dos dados a serem inseridos:

~~34~~ ~~46~~ ~~85~~ 55 13

Próxima posição vazia????

$h(55) = 55 \bmod 7 = 6$ **Colisão !**

	0
85	1
	2
	3
46	4
	5
34	6

← próx. pos livre

← pos desejada

No caso acima, temos uma colisão na posição 6 da tabela, que faz com que o elemento 5 tenha de ocupar a próx. pos disponível da tabela ao invés da inicialmente desejada. Nesse sentido, ao invés de ter seu valor armazenado no último espaço da tabela, será na pos. vaga em seu início.

EXEMPLO 5/5

Capacidade da tabela: 7

Chaves dos dados a serem inseridos:

~~34~~ ~~46~~ ~~85~~ ~~55~~ 13

Próxima posição vazia????

$h(13) = 13 \bmod 7 = 6$ **Colisão !**

55	0
85	1
	2
	3
46	4
	5
34	6

← próx pos livre

← pos desejada

O mesmo ocorrerá c/ o valor 13, que tem como dado a posição 6, mas terá que ocupar a pos 2, que é a próxima livre.

* Remoção de elementos – Endereçamento aberto.

↳ Podemos demarcar pos. removidas c/ uma "flag".

OBS: caso hajam muitos elementos removidos, a busca se tornará menos eficiente, devido à grande quantidade de flags.

Nesse sentido, p/ evitarmos isso, devemos reinserir os elementos NÃO removidos da tabela p/ tornar as posições não ocupadas livres novamente.

55	
-1	
13	
17	
46	
34	

→ "flag" de elemento removido!