



HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.



CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT



ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN

BẢNG BĂM

ONE LOVE. ONE FUTURE.

1. Địa chỉ trực tiếp, bảng băm và hàm băm
2. Xung đột và giải quyết xung đột

MỤC TIÊU

Sau bài học này, người học có thể:

1. Hiểu được ý nghĩa của **bảng băm**, **hàm băm** trong việc lưu trữ và tra cứu dữ liệu
2. Nắm được cách **giải quyết xung đột** khi sử dụng bảng băm

ĐẶT VẤN ĐỀ

- Cho bảng T và bản ghi x gồm khoá key và dữ liệu data đi kèm, ta cần hỗ trợ các thao tác sau:
 - $\text{Insert}(T, x)$: thêm bản ghi $x(\text{key}, \text{data})$ vào bảng T
 - $\text{Delete}(T, x)$: xóa bản ghi $x(\text{key}, \text{data})$ khỏi bảng T
 - $\text{Search}(T, x)$: tìm kiếm bản ghi $x(\text{key}, \text{data})$ có xuất hiện trong bảng T hay không
- Ta muốn thực hiện các thao tác này một cách nhanh chóng mà không phải thực hiện việc sắp xếp các bản ghi. Bảng băm (hash table) là cách tiếp cận giải quyết vấn đề đặt ra.
- Trong bài này, ta sẽ chỉ xét khoá là các số nguyên dương (có thể rất lớn).

1. Địa chỉ trực tiếp, bảng băm và hàm băm
2. Xung đột và giải quyết xung đột

1. ĐỊA CHỈ TRỰC TIẾP, BẢNG BĂM VÀ HÀM BĂM

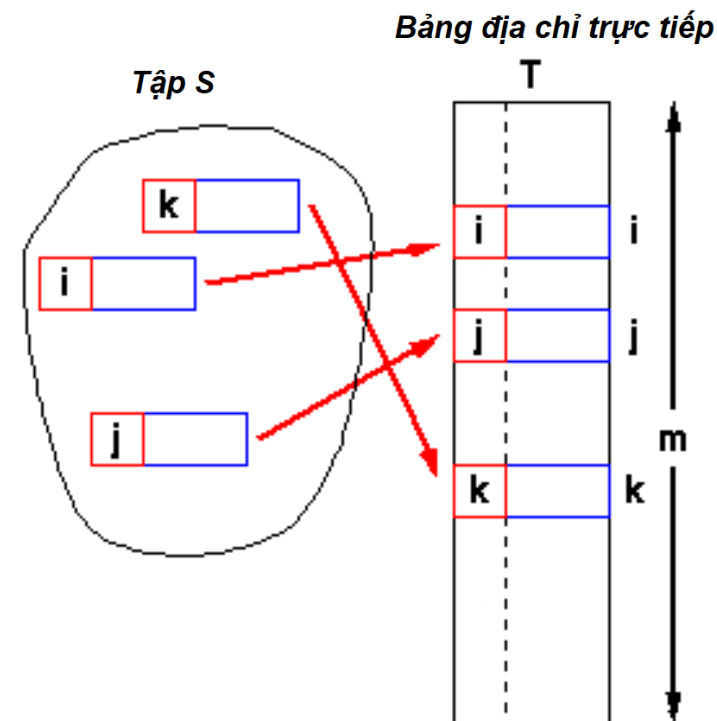
- Giả sử tập S gồm n phần tử, mỗi phần tử gồm khóa **key** và dữ liệu **data**. Trường khóa **key** của mỗi phần tử x :

- Là các số trong khoảng từ 0 đến $m-1$ ($m \geq n$)
- Các khoá là khác nhau từng đôi.

- Nếu sử dụng phương pháp địa chỉ trực tiếp để lưu trữ n phần tử này, ta dùng một mảng $T[0..m-1]$:

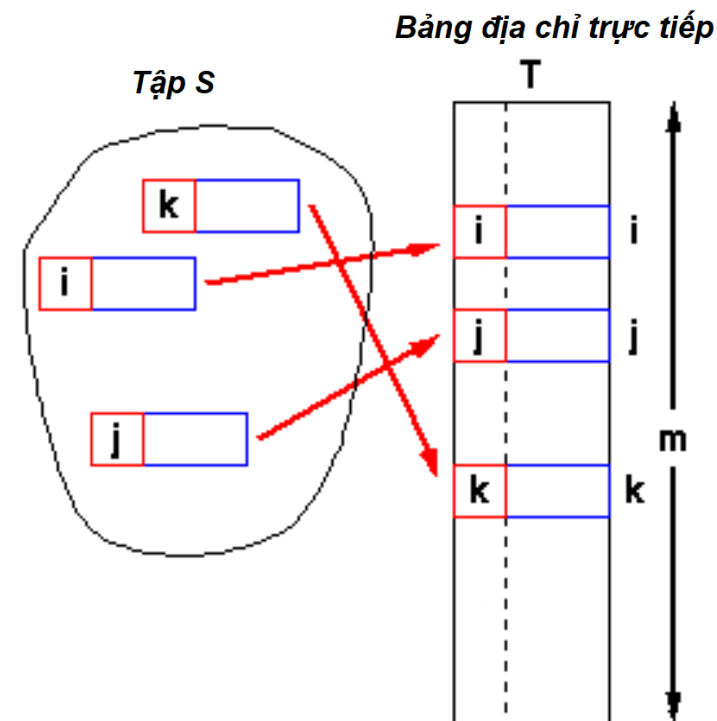
- $T[i] = x$ nếu $x \in T$ và $\text{key}[x] = i$
- $T[i] = \text{NULL}$ nếu không có phần tử nào trong tập S có trường $\text{key} = i$

→ T được gọi là bảng địa chỉ trực tiếp (*direct-address table*), các phần tử trong bảng T sẽ được gọi là các ô.



1. ĐỊA CHỈ TRỰC TIẾP, BẢNG BĂM VÀ HÀM BĂM

- Nếu sử dụng phương pháp địa chỉ trực tiếp để lưu trữ n phần tử này, ta dùng một mảng $T[0..m-1]$:
 - $T[i] = x$ nếu $x \in T$ và $\text{key}[x] = i$
 - $T[i] = \text{NULL}$ nếu không có phần tử nào trong tập S có trường $\text{key} = i$
- Các phép toán được cài đặt một cách trực tiếp:
 - **SEARCH(T, k)**
return $T[k]$
 - **INSERT(T, x)**
 $T[\text{key}[x]] = x$
 - **DELETE(T, x)**
 $T[\text{key}[x]] = \text{NULL}$



1. ĐỊA CHỈ TRỰC TIẾP, BẢNG BĂM VÀ HÀM BĂM

- Hạn chế của phương pháp địa chỉ trực tiếp:
 - Tốn bộ nhớ khi: số phần tử $n \ll m$ (Phương pháp địa chỉ trực tiếp làm việc tốt nếu như biên độ m của các khoá là tương đối nhỏ).
 - Nếu các khoá là các số nguyên 32-bit thì sao?
 - Vấn đề 1: bảng địa chỉ trực tiếp sẽ phải có 2^{32} (hơn 4 tỷ) phần tử
 - Vấn đề 2: ngay cả khi bộ nhớ không là vấn đề, thì thời gian khởi tạo các phần tử là NULL cũng là rất tốn kém
- Cách giải quyết: Ánh xạ khoá từ khoảng $0..m-1$ vào khoảng biến đổi nhỏ hơn.
→ Ánh xạ này được gọi là hàm băm (*hash function*) và bảng lưu trữ lúc này được gọi là bảng băm (hash table).

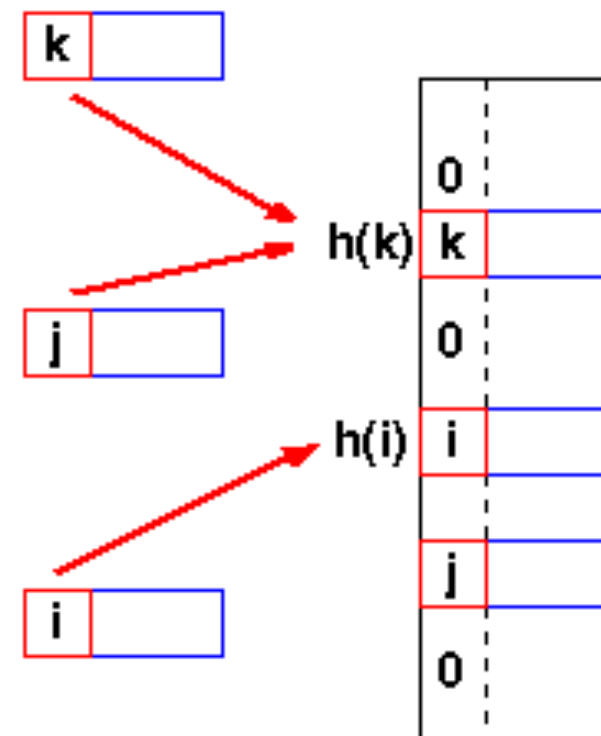
1. ĐỊA CHỈ TRỰC TIẾP, BẢNG BĂM VÀ HÀM BĂM

- Trong phương pháp địa chỉ trực tiếp, phần tử với khoá k được cất giữ ở ô k .
- Với bảng băm, phần tử với khoá k được cất giữ ở ô $h(k)$, trong đó ta sử dụng hàm băm h để xác định ô cất giữ phần tử này từ khoá của nó (k).
- Định nghĩa.** Hàm băm h là ánh xạ từ không gian khoá U vào các ô của bảng băm $T[0..p-1]$:

$$h : U \rightarrow \{0, 1, \dots, p-1\}$$

- Ta sẽ nói rằng phần tử với khoá k được *gắn vào* ô $h(k)$, và nói $h(k)$ là *giá trị băm* của khoá k .

→ Vấn đề nảy sinh lại là xung đột (*collision*), khi nhiều khoá được đặt tương ứng với cùng một ô trong bảng địa chỉ T .



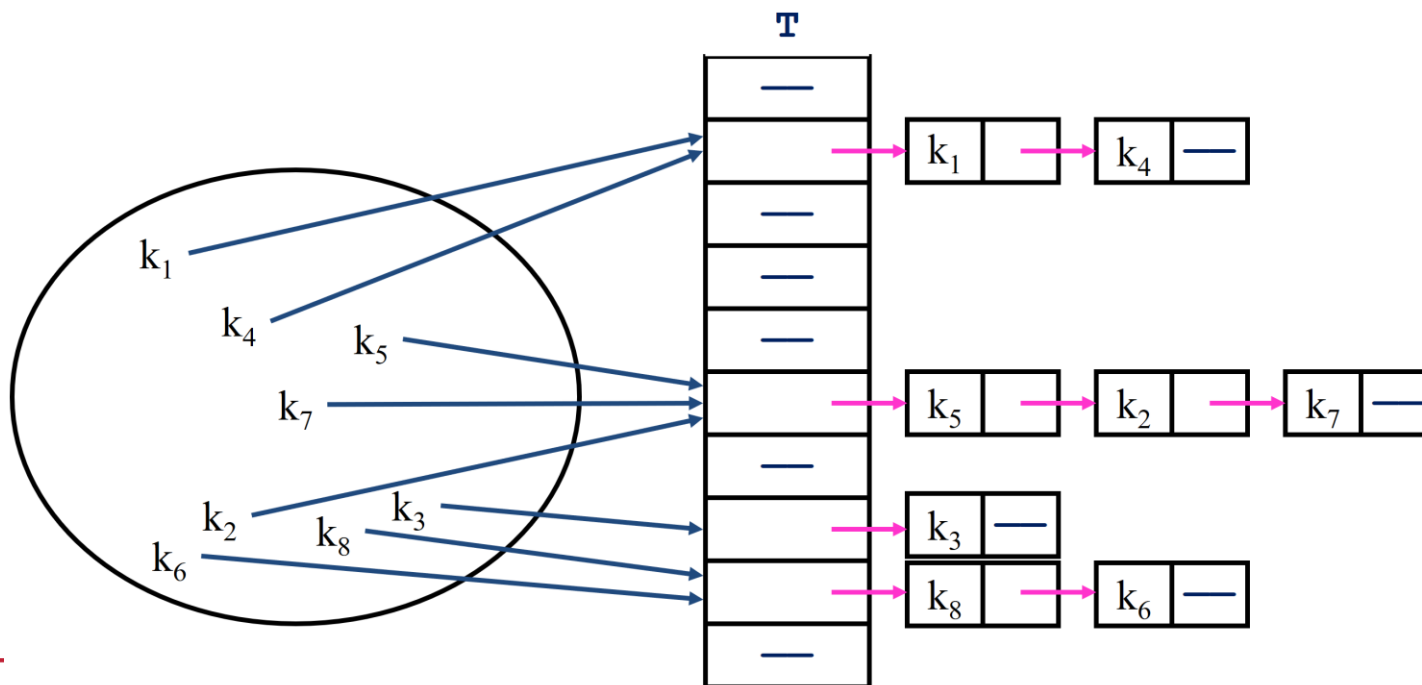
1. Địa chỉ trực tiếp, bảng băm và hàm băm

2. Xung đột và giải quyết xung đột

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Cách giải quyết xung đột bằng **phương pháp tạo chuỗi**:
 - Tạo danh sách liên kết để chứa các phần tử được gán với cùng một vị trí trong bảng.

Ví dụ: $h(k) = k \bmod p$. Nếu ô $T[h(k)]$ trong bảng băm đã bận, ta thêm phần tử mới này vào đầu danh sách móc nối tại ô $T[h(k)]$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

▪ Ví dụ: Giải quyết xung đột theo phương pháp tạo chuỗi

- Một bảng băm được cấp phát p vị trí đánh chỉ số $0, 1, \dots, p-1$ để lưu trữ cặp khóa - giá trị. Bảng băm áp dụng cơ chế tạo chuỗi với hàm dò $h(k) = k \bmod p$
- Ban đầu, bảng trống rỗng. Hãy vẽ trạng thái bảng (mỗi vị trí chỉ cần ghi khóa nếu đang dùng và ghi “/” nếu chưa dùng) khi chèn lần lượt các khóa **22, 1, 13, 11, 24, 33, 18, 42, 31** vào bảng với trường hợp $p = 11$.

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

0	/
1	/
2	/
3	/
4	/
5	/
6	/
7	/
8	/
9	/
10	/

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

0	/
1	/
2	/
3	/
4	/
5	/
6	/
7	/
8	/
9	/
10	/

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(22)=0$$

0	/
1	/
2	/
3	/
4	/
5	/
6	/
7	/
8	/
9	/
10	/

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(22)=0$$

0		→ 22 /
1	/	
2	/	
3	/	
4	/	
5	/	
6	/	
7	/	
8	/	
9	/	
10	/	

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(1)=1$$

0		→	22	/
1	/			
2	/			
3	/			
4	/			
5	/			
6	/			
7	/			
8	/			
9	/			
10	/			

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(1)=1$$

0		→	22	/
1		→	1	/
2	/			
3	/			
4	/			
5	/			
6	/			
7	/			
8	/			
9	/			
10	/			

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(13)=2$$

0		→	22	/
1		→	1	/
2	/			
3	/			
4	/			
5	/			
6	/			
7	/			
8	/			
9	/			
10	/			

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(13)=2$$

0		→	22	/
1		→	1	/
2		→	13	/
3	/			
4	/			
5	/			
6	/			
7	/			
8	/			
9	/			
10	/			

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, **11**, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(11)=0$$

0		→	22	/
1		→	1	/
2		→	13	/
3	/			
4	/			
5	/			
6	/			
7	/			
8	/			
9	/			
10	/			

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

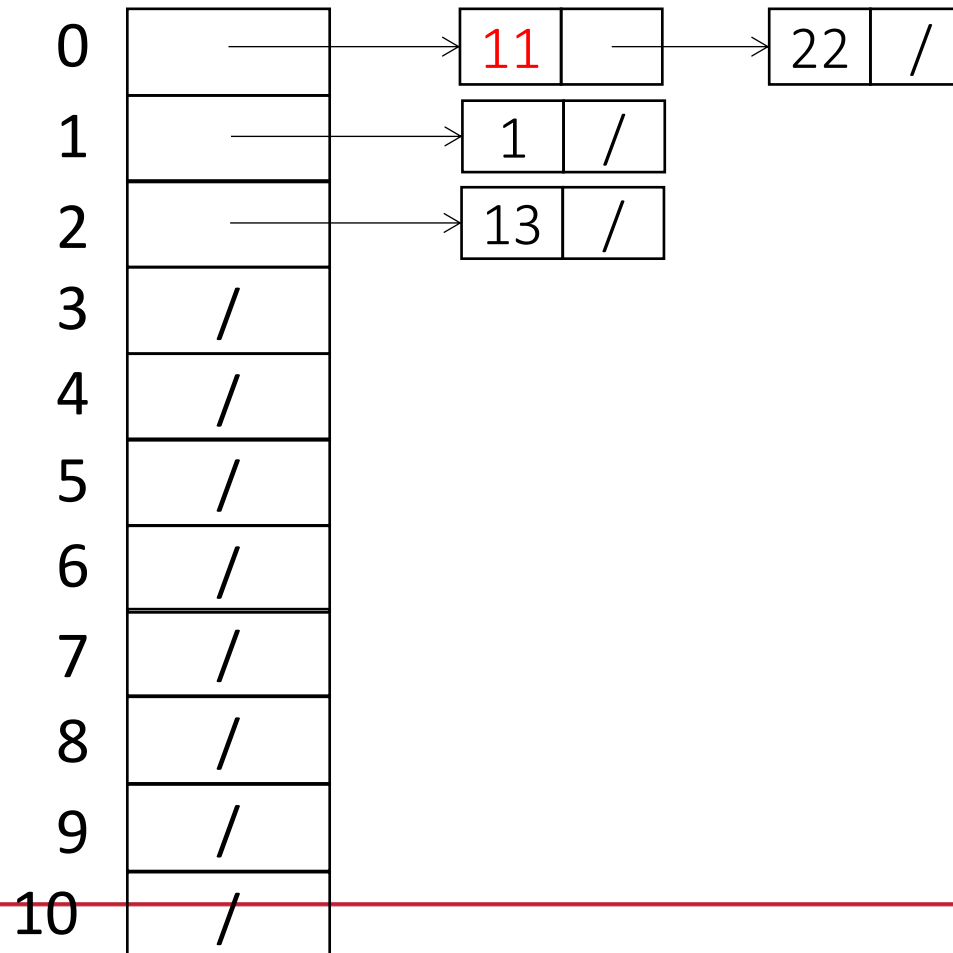
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, **11**, 24, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(11)=0$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

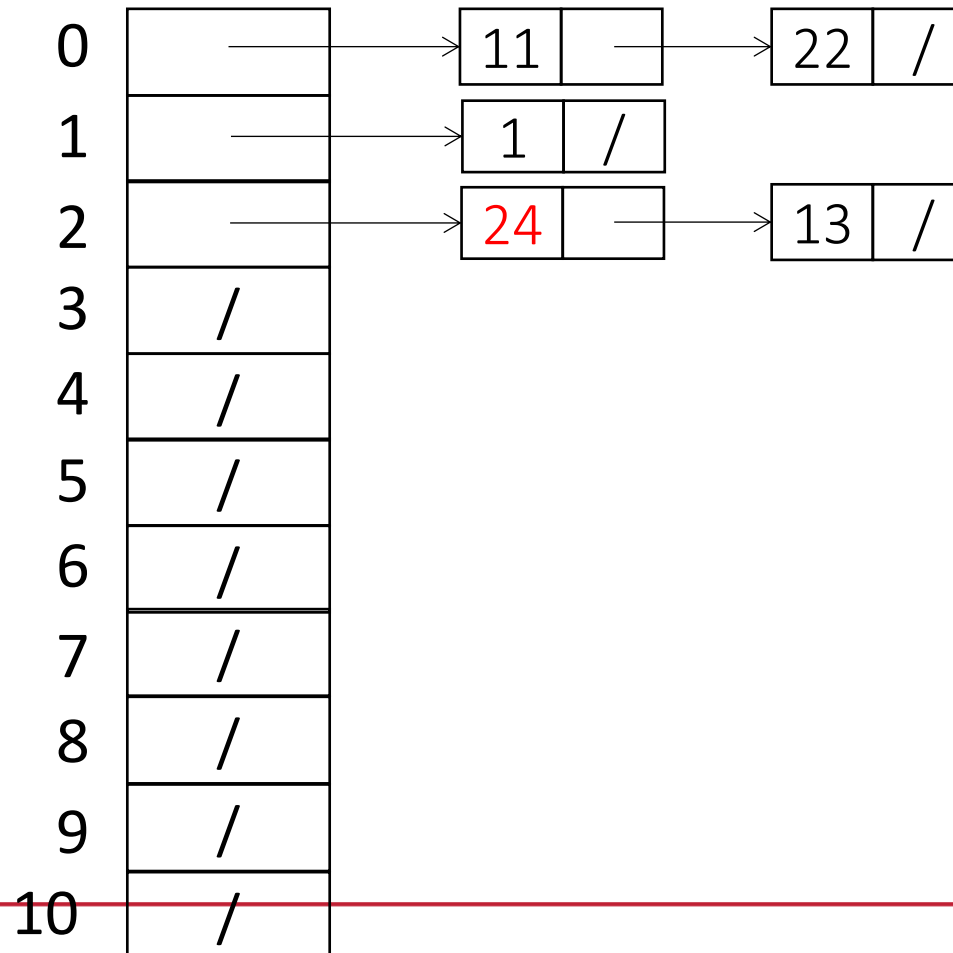
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, **24**, 33, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(24)=2$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

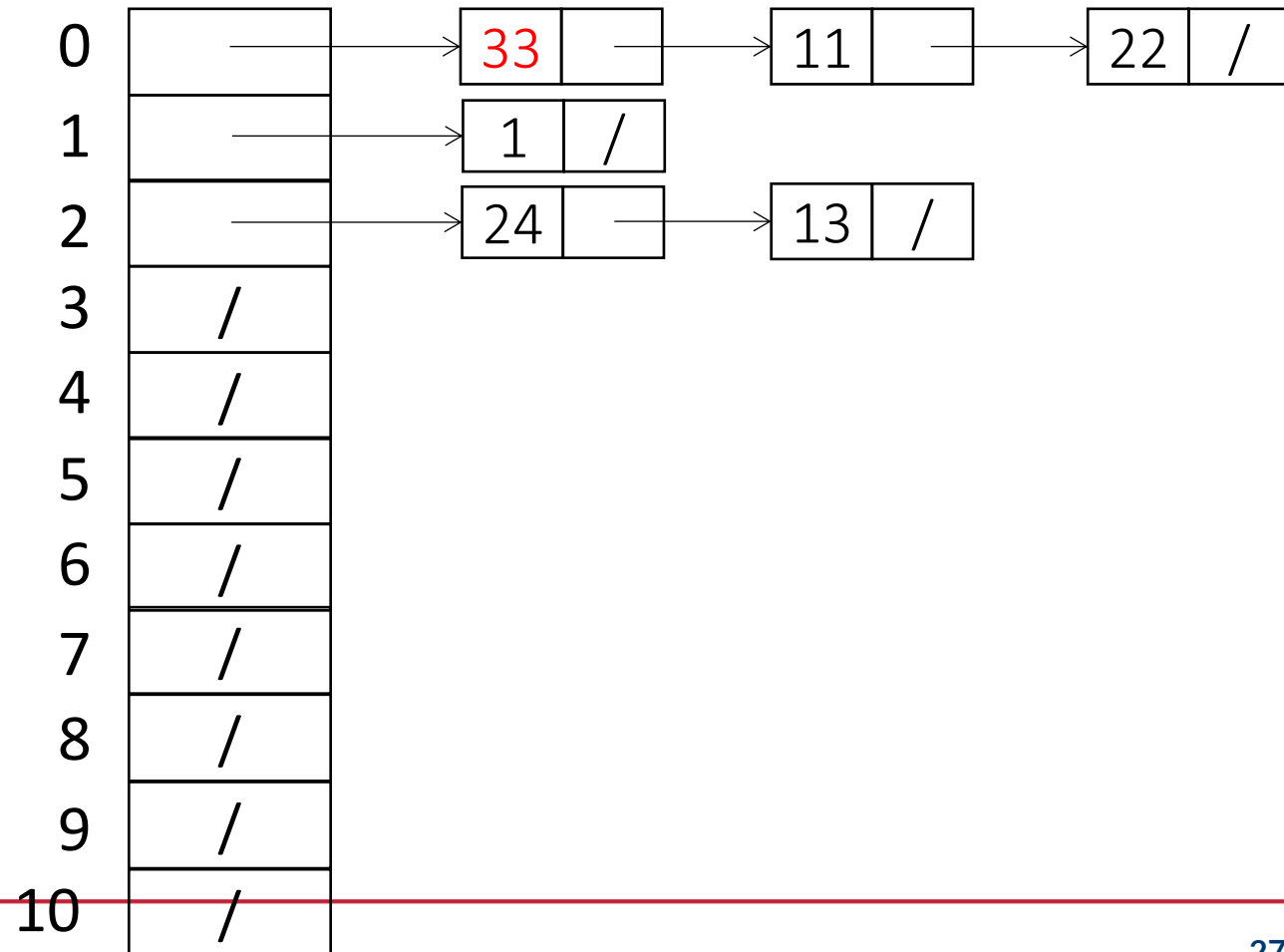
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, **33**, 18, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(33)=0$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

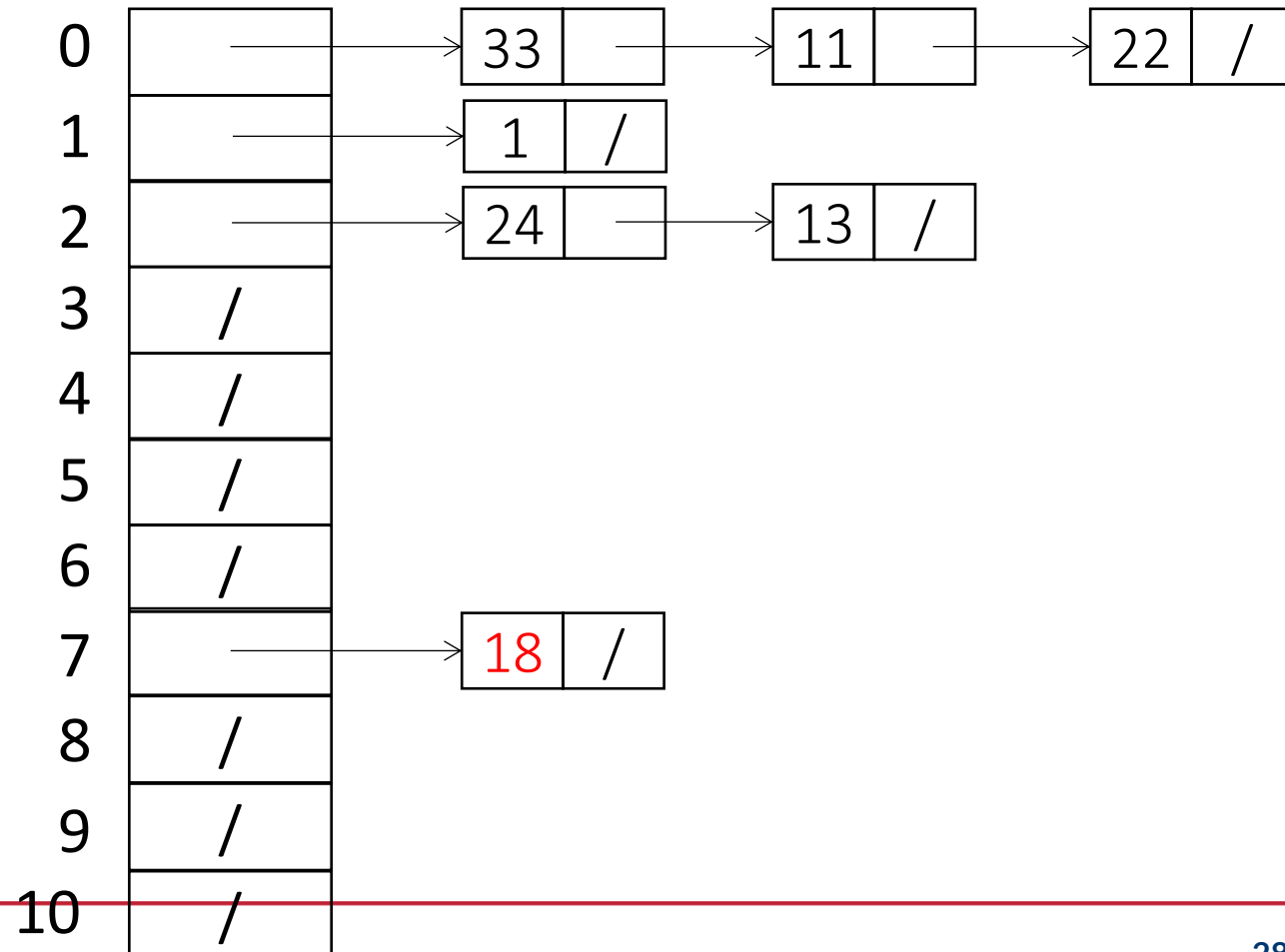
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, **18**, 42, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(18)=7$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

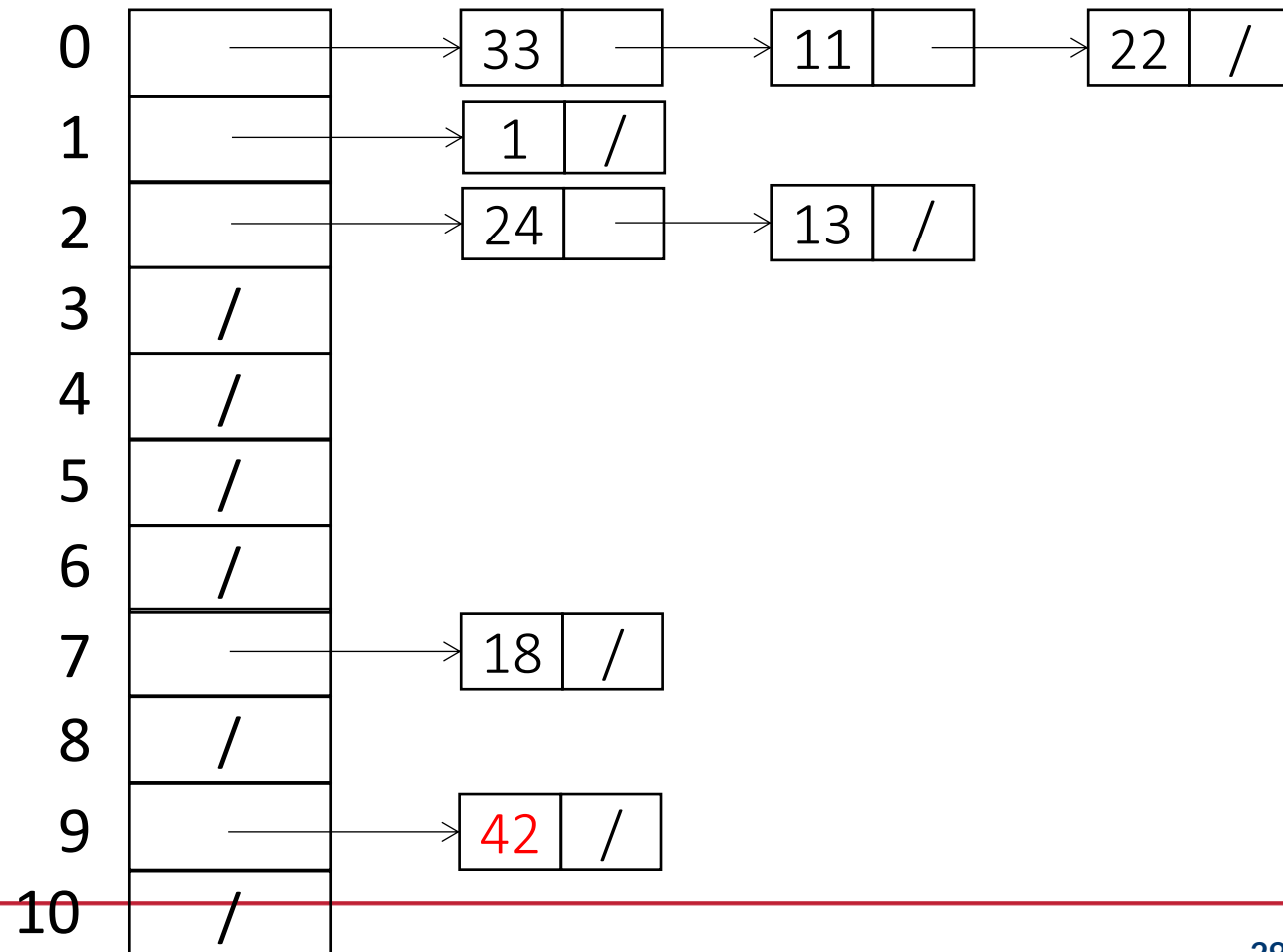
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, **42**, 31

Chaining

$$h(k) = k \bmod 11$$

$$h(42)=9$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

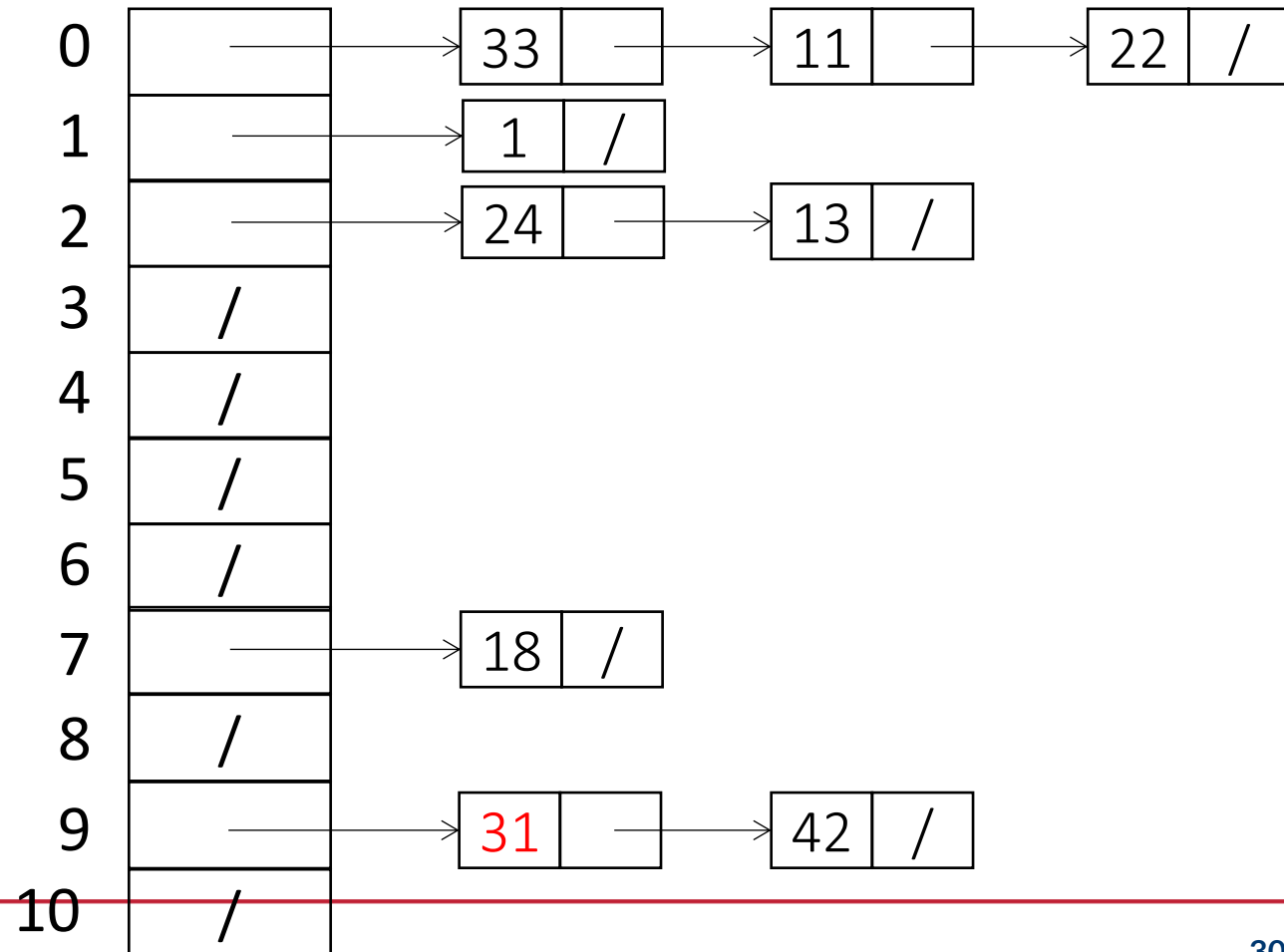
- Ví dụ: giải quyết xung đột theo phương pháp tạo chuỗi

22, 1, 13, 11, 24, 33, 18, 42, **31**

Chaining

$$h(k) = k \bmod 11$$

$$h(31)=9$$



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

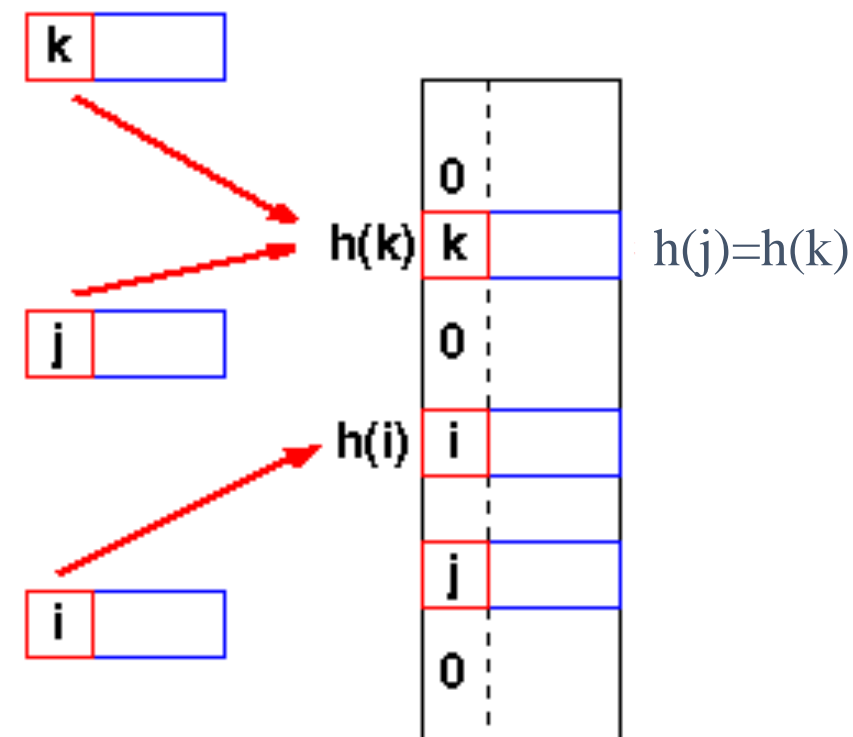
- Xung đột (*collision*): xảy ra khi nhiều khoá được đặt tương ứng với cùng một ô trong bảng băm T.
- Ta cần giải quyết xung đột như thế nào?

- Cách giải quyết 1:

Tạo chuỗi (chaining)

- Cách giải quyết 2:

Phương pháp địa chỉ mở (open addressing)



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Cách giải quyết xung đột bằng **phương pháp địa chỉ mở**:
 - Khi bổ sung (Insert): nếu ô là đã bận, thì ta tìm kiếm ô khác,, cho đến khi tìm được ô rỗng (phương pháp dò thử - *probing*).
 - Để tìm kiếm (search), ta sẽ tìm dọc theo dãy các phép dò thử giống như dãy dò thử khi thực hiện chèn phần tử vào bảng.
 - Nếu tìm được phần tử với khoá đã cho thì trả lại nó,
 - Nếu tìm được con trỏ NULL, thì phần tử cần tìm không có trong bảng
 - Một số phương pháp dò:
 - Dò tuyến tính (Linear Probing)
 - Dò bậc hai (Quadratic Probing)
 - Hàm băm kép (Double Hashing)

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính (Linear Probing):
 - Mỗi lần dò, tăng chỉ số i lên một đơn vị: $h(k, i) = (h'(k) + i) \bmod p$ với $0 \leq i \leq p-1$

$$h'(k) = k \bmod p$$

→ Đầu tiên thử $h(k, 0) = h'(k)$, nếu vị trí này bận, thử tiếp $h(k, 1)$... cho đến khi tìm được ô trống.

Lần bấm 0: $h'(k) \bmod p$

Lần bấm 1: $(h'(k) + 1) \bmod p$

Lần bấm 2: $(h'(k) + 2) \bmod p$

Lần bấm 3: $(h'(k) + 3) \bmod p$

...

Lần bấm i : $(h'(k) + i) \bmod p$

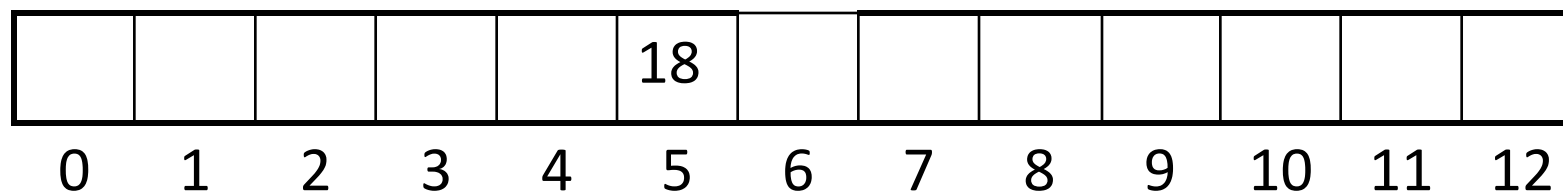
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5,



					18							
0	1	2	3	4	5	6	7	8	9	10	11	12

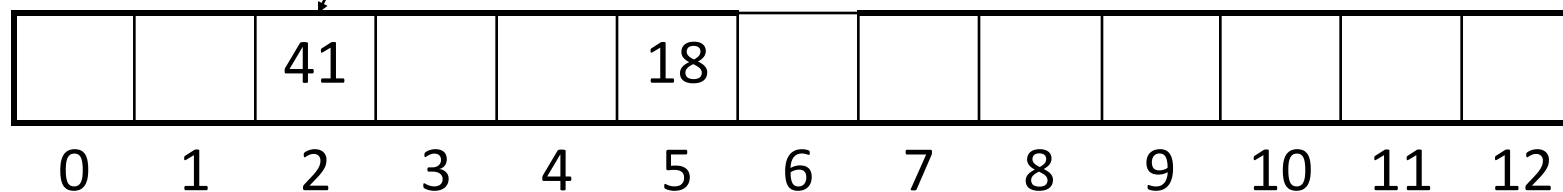
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2,



		41			18							
0	1	2	3	4	5	6	7	8	9	10	11	12

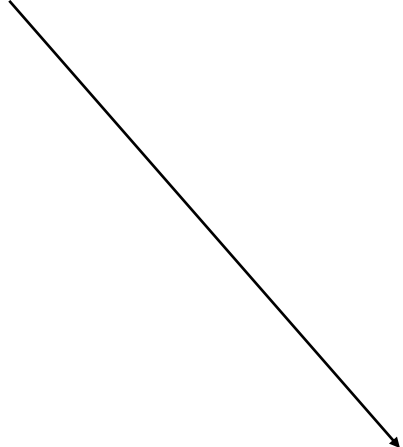
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9,



		41			18				22			
0	1	2	3	4	5	6	7	8	9	10	11	12


2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7,



		41			18		59		22			
0	1	2	3	4	5	6	7	8	9	10	11	12

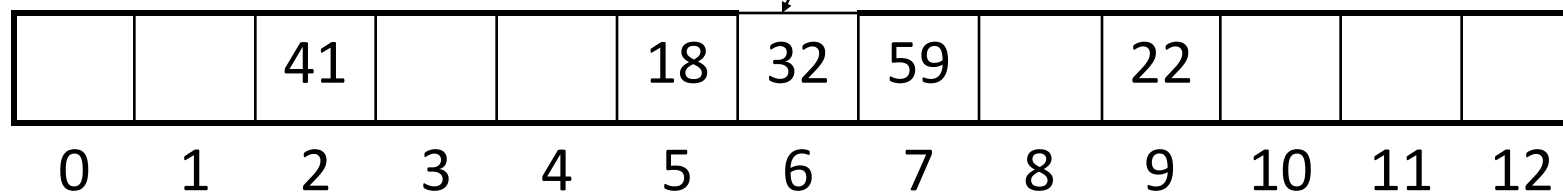
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6,



		41			18	32	59		22			
0	1	2	3	4	5	6	7	8	9	10	11	12

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

▪ Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở

• Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5,

		41			18	32	59		22			
0	1	2	3	4	5	6	7	8	9	10	11	12

Lần băm 0: $h'(k) \bmod m$

Lần băm 1: $(h'(k) + 1) \bmod m$

Lần băm 2: $(h'(k) + 2) \bmod m$

Lần băm 3: $(h'(k) + 3) \bmod m$

...

Lần băm i : $(h'(k) + i) \bmod m$

Đầu tiên thử

$h(k, 0) = h'(k)$, nếu vị trí này bận, thử tiếp $h(k, 1)...$ cho đến khi tìm được ô trống.

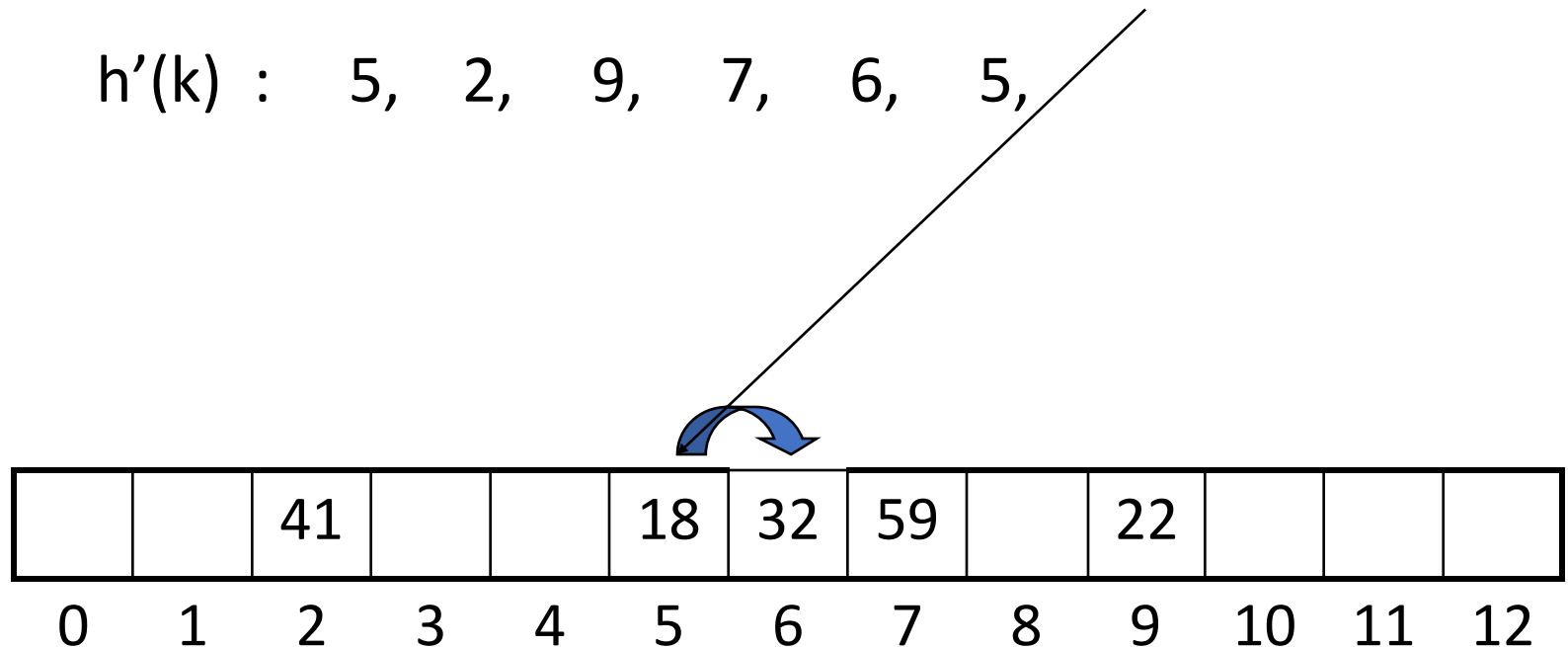
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5,



		41			18	32	59		22			
0	1	2	3	4	5	6	7	8	9	10	11	12

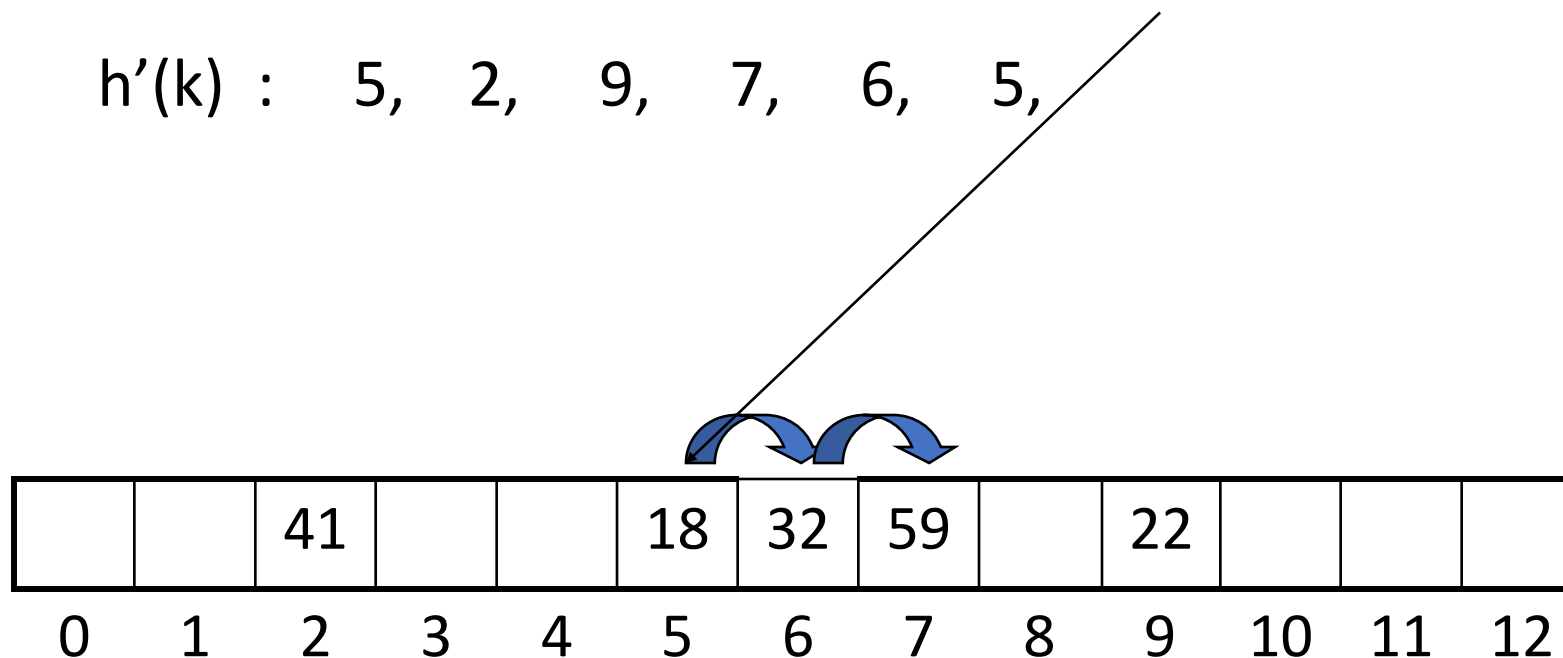
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5,



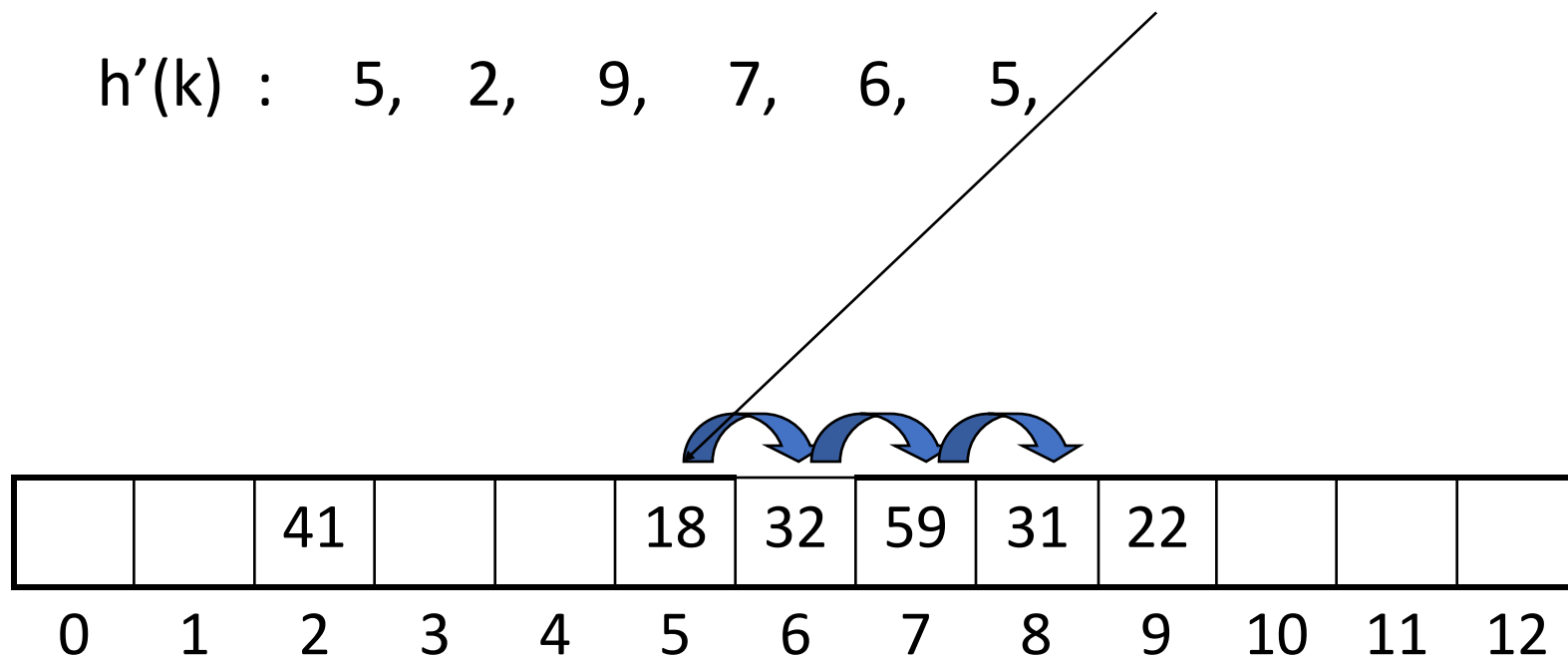
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5,



2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5, 8

Lần băm 0: $h'(k) \bmod m$

Lần băm 1: $(h'(k) + 1) \bmod m$

Lần băm 2: $(h'(k) + 2) \bmod m$

Lần băm 3: $(h'(k) + 3) \bmod m$

...

Lần băm i : $(h'(k) + i) \bmod m$

Đầu tiên thử
 $h(k, 0) = h'(k)$, nếu vị
trí này bận, thử tiếp
 $h(k, 1)...$ cho đến khi
tìm được
ô trống.

		41			18	32	59	31	22			
0	1	2	3	4	5	6	7	8	9	10	11	12

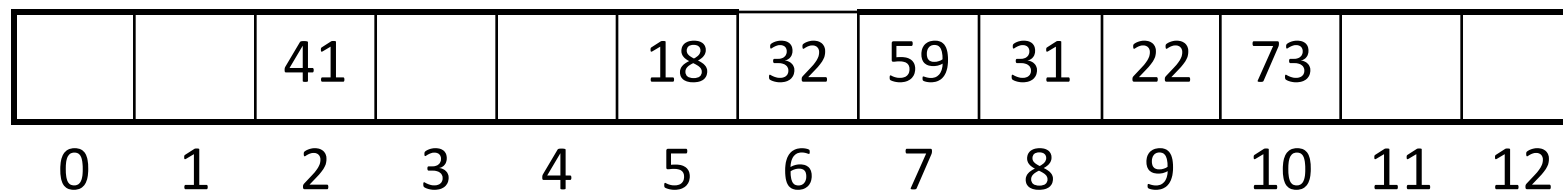
2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Ví dụ: giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò tuyến tính

Sử dụng hàm băm: $h'(k) = k \% 13$

Chèn: 18, 41, 22, 59, 32, 31, 73

$h'(k)$: 5, 2, 9, 7, 6, 5, 8



Lần băm 0: $h'(k) \bmod m$

Lần băm 1: $(h'(k) + 1) \bmod m$

Lần băm 2: $(h'(k) + 2) \bmod m$

Lần băm 3: $(h'(k) + 3) \bmod m$

...

Lần băm i : $(h'(k) + i) \bmod m$

Đầu tiên thử $h(k, 0) = h'(k)$, nếu vị trí này bận, thử tiếp $h(k, 1)...$ cho đến khi tìm được ô trống.

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Giải quyết xung đột bằng phương pháp địa chỉ mở

- Phương pháp dò bậc hai (Quadratic Probing):

- Mỗi lần dò, tăng chỉ số i lên bình phương:

$$h(k, i) = (h'(k) + i^2) \bmod p \text{ với } 0 \leq i \leq p-1; h'(k) = k \bmod p$$

→ Đầu tiên thử $h(k, 0) = h'(k)$, nếu vị trí này bận, thử tiếp $h(k, 1)$... cho đến khi tìm được ô trống.

Lần bấm 0: $h'(k) \bmod p$

Lần bấm 1: $(h'(k) + 1) \bmod p$

Lần bấm 2: $(h'(k) + 4) \bmod p$

Lần bấm 3: $(h'(k) + 9) \bmod p$

...

Lần bấm i : $(h'(k) + i^2) \bmod p$

2. XUNG ĐỘT VÀ GIẢI QUYẾT XUNG ĐỘT

- Giải quyết xung đột bằng phương pháp địa chỉ mở
 - Phương pháp dò theo hàm băm kép (Double Hashing Probing):
 - Mỗi lần dò, tăng chỉ số i lên một đơn vị:

$$h(k,i) = (h_1(k) + i h_2(k)) \bmod p \text{ với } 0 \leq i \leq p-1$$

với $h_1(k)$ và $h_2(k)$ là hai hàm băm

hàm băm $h_2(k)$ được coi là step function (số ô nhảy qua sau mỗi lần băm)

TỔNG KẾT VÀ GỢI MỞ

1. Bài học đã trình bày các kiến thức cơ bản liên quan đến **bảng băm, hàm băm, xung đột và giải quyết xung đột**
2. Tiếp sau bài này, người học sẽ làm **bài tập lập trình: tính mã băm của một chuỗi ký tự**

A large graphic on the left side of the slide. It features a dark blue background with a circular pattern of red dots of varying sizes, creating a sense of depth and movement. The word "HUST" is centered within this graphic in a white, bold, sans-serif font.

HUST

THANK YOU !