

## Parte 3: Hashing

Eduardo Freire Nakamura  
eduardo.nakamura@fucapi.br

CESF – FUCAPI  
Ciência da Computação, Engenharia da  
Computação e Sistemas de Informação

# Tabela de endereçamento direto

- O que é
  - Uma tabela (arranjo) onde qualquer **chave** pode ser localizado através de acesso direto em tempo  $O(1)$
- Aplicações
  - Dicionário: consulta a elementos em tempo constante
  - Ex: Tabela de símbolos em compiladores

# Endereçamento direto

- Em resumo
  - Cada chave possível tem um endereço único
  - O acesso à chave é direto, pelo endereço
- Problema
  - Universo de possíveis chaves/itens é muito grande
  - O arranjo precisa ser muito grande

# Um exemplo

- Universo de chaves
  - Nomes com 10 caracteres
  - 26 possíveis caracteres (alfabeto)
- Tamanho do arranjo
  - $26^{10} = 141.167.095.653.376$  elementos!!!
  - 140 TB??

Quais os problemas que  
você identifica?



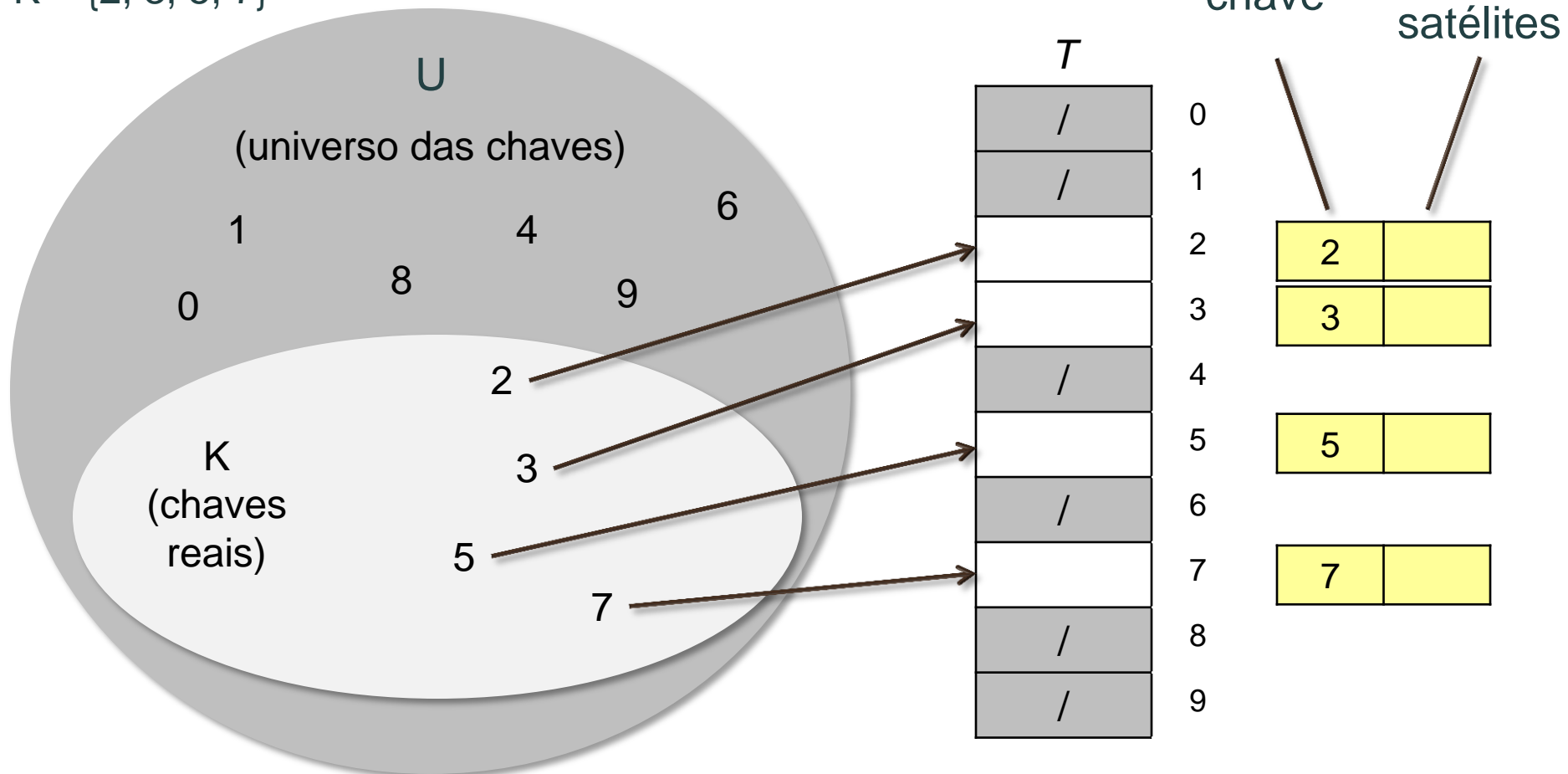
# Definindo o endereçamento direto

- Dados
  - Cada elemento possui uma chave
  - Não existem elementos com chaves repetidas
  - O conjunto universo de chaves (chaves possíveis) é definido por  $U = \{0, 1, 2, \dots, m-1\}$ , onde  $m$  não é muito grande
- A tabela de endereçamento direto
  - Arranjo dinâmico  $T[0 \dots m-1]$
  - Cada posição corresponde a uma chave no universo  $U$

# Visualizando o endereçamento direto

$U = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$K = \{2, 3, 5, 7\}$



# Funções do endereçamento direto

DIRECT-ADDRESS-SEARCH ( $T, k$ )

1: **return**  $T[k]$ ;

Qual o custo destas?

DIRECT-ADDRESS-INSERT ( $T, x$ )

1:  $T[x.chave] \leftarrow x$ ;

DIRECT-ADDRESS-DELETE ( $T, x$ )

1:  $T[x.chave] \leftarrow \text{NIL}$ ;



# Para que serve o hashing?

- O que é
  - Adaptação do endereçamento direto
  - Reduzir espaço, impactando pouco no desempenho
- Objetivo
  - Mapear um espaço enorme de chaves em um espaço de inteiros relativamente pequeno
- Como é feito?
  - Através de uma função chamada *função hash*
  - O inteiro gerado pela *função hash* é chamado *valor hash* e é usado para encontrar a localização do item



# Tabela hash

- Tabela de endereçamento direto
  - Espaço necessário  $\Theta(|U|)$
  - Tempo de acesso é sempre  $O(1)$
  - Elemento de chave  $k$  se encontra na posição  $k$
- Tabela hashing
  - Espaço necessário  $\Theta(|K|)$
  - Tempo médio de acesso  $O(1)$
  - Elemento de chave  $k$  se encontra na posição  $h(k)$

$h(k)$  é uma **função hash** que mapeia o universo  $U$  de chaves nas posições da **tabela hash**  $T[0..m-1]$

# Um problema do hashing?

Qual problema do hashing  
você identifica?

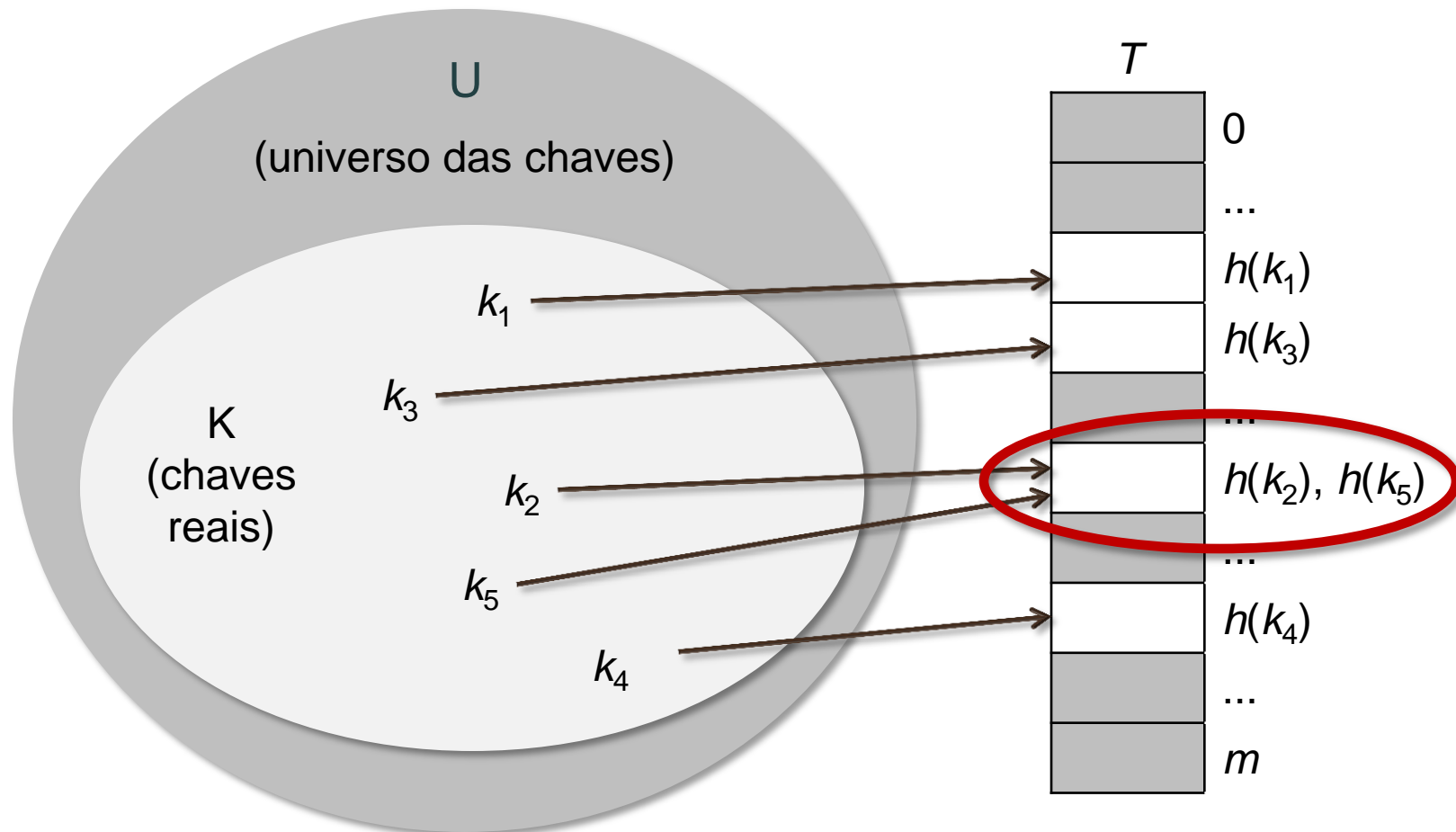


## COLISÃO

Eventualmente duas chaves  
podem obter o mesmo valor hash

$$k_1 \neq k_2 \rightarrow h(k_1) = h(k_2)$$

# Visualizando o hashing



# Função hash

- Interpretando as chaves
  - As funções hash normalmente assumem que o conjunto universo das chaves é o conjunto  $\mathbf{N} = \{0, 1, 2, 3, \dots\}$
  - Se as chaves não são números naturais, então precisamos encontrar uma forma de interpretá-las como números naturais
- ASCII
  - 128 caracteres (ASCII simples - sem acentuação)
  - 256 caracteres (ASCII estendido - com acentuação)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	<b>NUL</b> (null)	32	20	040	&#32;	<b>Space</b>	64	40	100	&#64;	<b>@</b>	96	60	140	&#96;	<b>`</b>
1	1	001	<b>SOH</b> (start of heading)	33	21	041	&#33;	<b>!</b>	65	41	101	&#65;	<b>A</b>	97	61	141	&#97;	<b>a</b>
2	2	002	<b>STX</b> (start of text)	34	22	042	&#34;	<b>"</b>	66	42	102	&#66;	<b>B</b>	98	62	142	&#98;	<b>b</b>
3	3	003	<b>ETX</b> (end of text)	35	23	043	&#35;	<b>#</b>	67	43	103	&#67;	<b>C</b>	99	63	143	&#99;	<b>c</b>
4	4	004	<b>EOT</b> (end of transmission)	36	24	044	&#36;	<b>\$</b>	68	44	104	&#68;	<b>D</b>	100	64	144	&#100;	<b>d</b>
5	5	005	<b>ENQ</b> (enquiry)	37	25	045	&#37;	<b>%</b>	69	45	105	&#69;	<b>E</b>	101	65	145	&#101;	<b>e</b>
6	6	006	<b>ACK</b> (acknowledge)	38	26	046	&#38;	<b>&amp;</b>	70	46	106	&#70;	<b>F</b>	102	66	146	&#102;	<b>f</b>
7	7	007	<b>BEL</b> (bell)	39	27	047	&#39;	<b>'</b>	71	47	107	&#71;	<b>G</b>	103	67	147	&#103;	<b>g</b>
8	8	010	<b>BS</b> (backspace)	40	28	050	&#40;	<b>(</b>	72	48	110	&#72;	<b>H</b>	104	68	150	&#104;	<b>h</b>
9	9	011	<b>TAB</b> (horizontal tab)	41	29	051	&#41;	<b>)</b>	73	49	111	&#73;	<b>I</b>	105	69	151	&#105;	<b>i</b>
10	A	012	<b>LF</b> (NL line feed, new line)	42	2A	052	&#42;	<b>*</b>	74	4A	112	&#74;	<b>J</b>	106	6A	152	&#106;	<b>j</b>
11	B	013	<b>VT</b> (vertical tab)	43	2B	053	&#43;	<b>+</b>	75	4B	113	&#75;	<b>K</b>	107	6B	153	&#107;	<b>k</b>
12	C	014	<b>FF</b> (NP form feed, new page)	44	2C	054	&#44;	<b>,</b>	76	4C	114	&#76;	<b>L</b>	108	6C	154	&#108;	<b>l</b>
13	D	015	<b>CR</b> (carriage return)	45	2D	055	&#45;	<b>-</b>	77	4D	115	&#77;	<b>M</b>	109	6D	155	&#109;	<b>m</b>
14	E	016	<b>SO</b> (shift out)	46	2E	056	&#46;	<b>.</b>	78	4E	116	&#78;	<b>N</b>	110	6E	156	&#110;	<b>n</b>
15	F	017	<b>SI</b> (shift in)	47	2F	057	&#47;	<b>/</b>	79	4F	117	&#79;	<b>O</b>	111	6F	157	&#111;	<b>o</b>
16	10	020	<b>DLE</b> (data link escape)	48	30	060	&#48;	<b>0</b>	80	50	120	&#80;	<b>P</b>	112	70	160	&#112;	<b>p</b>
17	11	021	<b>DC1</b> (device control 1)	49	31	061	&#49;	<b>1</b>	81	51	121	&#81;	<b>Q</b>	113	71	161	&#113;	<b>q</b>
18	12	022	<b>DC2</b> (device control 2)	50	32	062	&#50;	<b>2</b>	82	52	122	&#82;	<b>R</b>	114	72	162	&#114;	<b>r</b>
19	13	023	<b>DC3</b> (device control 3)	51	33	063	&#51;	<b>3</b>	83	53	123	&#83;	<b>S</b>	115	73	163	&#115;	<b>s</b>
20	14	024	<b>DC4</b> (device control 4)	52	34	064	&#52;	<b>4</b>	84	54	124	&#84;	<b>T</b>	116	74	164	&#116;	<b>t</b>
21	15	025	<b>NAK</b> (negative acknowledge)	53	35	065	&#53;	<b>5</b>	85	55	125	&#85;	<b>U</b>	117	75	165	&#117;	<b>u</b>
22	16	026	<b>SYN</b> (synchronous idle)	54	36	066	&#54;	<b>6</b>	86	56	126	&#86;	<b>V</b>	118	76	166	&#118;	<b>v</b>
23	17	027	<b>ETB</b> (end of trans. block)	55	37	067	&#55;	<b>7</b>	87	57	127	&#87;	<b>W</b>	119	77	167	&#119;	<b>w</b>
24	18	030	<b>CAN</b> (cancel)	56	38	070	&#56;	<b>8</b>	88	58	130	&#88;	<b>X</b>	120	78	170	&#120;	<b>x</b>
25	19	031	<b>EM</b> (end of medium)	57	39	071	&#57;	<b>9</b>	89	59	131	&#89;	<b>Y</b>	121	79	171	&#121;	<b>y</b>
26	1A	032	<b>SUB</b> (substitute)	58	3A	072	&#58;	<b>:</b>	90	5A	132	&#90;	<b>Z</b>	122	7A	172	&#122;	<b>z</b>
27	1B	033	<b>ESC</b> (escape)	59	3B	073	&#59;	<b>:</b>	91	5B	133	&#91;	<b>[</b>	123	7B	173	&#123;	<b>{</b>
28	1C	034	<b>FS</b> (file separator)	60	3C	074	&#60;	<b>&lt;</b>	92	5C	134	&#92;	<b>\</b>	124	7C	174	&#124;	<b> </b>
29	1D	035	<b>GS</b> (group separator)	61	3D	075	&#61;	<b>=</b>	93	5D	135	&#93;	<b>]</b>	125	7D	175	&#125;	<b>}</b>
30	1E	036	<b>RS</b> (record separator)	62	3E	076	&#62;	<b>&gt;</b>	94	5E	136	&#94;	<b>^</b>	126	7E	176	&#126;	<b>~</b>
31	1F	037	<b>US</b> (unit separator)	63	3F	077	&#63;	<b>?</b>	95	5F	137	&#95;	<b>_</b>	127	7F	177	&#127;	<b>DEL</b>

# Função hash

- ASCII
  - 128 caracteres (ASCII simples - sem acentuação)
  - 256 caracteres (ASCII estendido - com acentuação)
- Como mapear a chave *cara* usando o ASCII simples (128 caracteres)?

Caractere	ASCII
c	99
a	97
r	114
a	97

$$\begin{array}{rcl} & & 99 \times 128^3 \\ & + & 97 \times 128^2 \\ & + & 114 \times 128^1 \\ & + & 97 \times 128^0 \\ \hline \text{cara} & = & 209.221.985 \end{array}$$

# Método da divisão

- Mapeia uma chave  $k$  para uma das  $m$  posições da tabela hash, tomando o resto de  $k$  dividido por  $m$

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

- A escolha de  $m$ 
  - Se  $m$  é uma potência de 2, ou seja,  $m = 2^p$ , então a função hash irá usar somente as posições representadas pelos  $p$  bits de mais baixa ordem
  - Escolha para  $m$  um número primo não muito próximo de uma potência de 2

# Um exemplo

- Suponha que
  - O universo de chaves: números inteiros de quatro dígitos
  - Deseja-se mapeá-los na tabela hash  $T[0...10]$
- Função hash ( $m = 11$ )

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




# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

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

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

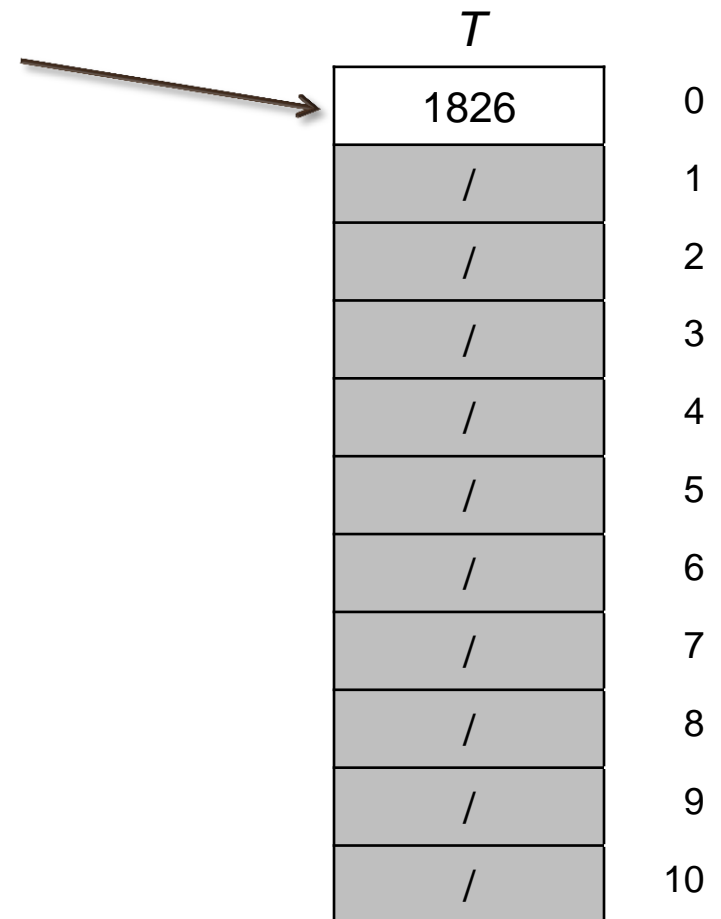


$T$	
1826	0
/	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

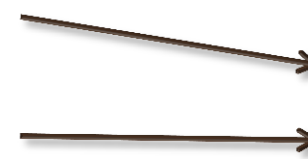


$T$	
1826	0
/	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$



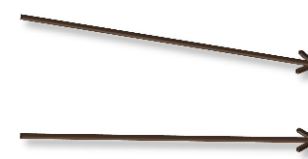
$T$	
1826	0
1981	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$



A diagram showing two arrows pointing from the equations to the table. The first arrow points from the result '0' of the first equation to the first row of the table. The second arrow points from the result '1' of the second equation to the second row of the table.


$T$	
1826	0
1981	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$



$T$	
1826	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1945 \rightarrow h(1945) = 1945 \bmod 11 = 9$$

$T$	
1826	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1945 \rightarrow h(1945) = 1945 \bmod 11 = 9$$

<i>T</i>	
1826	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
1945	9
/	10



# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1945 \rightarrow h(1945) = 1945 \bmod 11 = 9$$

$$k = 2011 \rightarrow h(2011) = 2011 \bmod 11 = 9$$

$T$	
1826	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
1945	9
/	10

# Um exemplo

$$k = 1826 \rightarrow h(1826) = 1826 \bmod 11 = 0$$


$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1945 \rightarrow h(1945) = 1945 \bmod 11 = 9$$

$$k = 2011 \rightarrow h(2011) = 2011 \bmod 11 = 9$$

$T$	
1826	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
1945, 2011	9
/	10



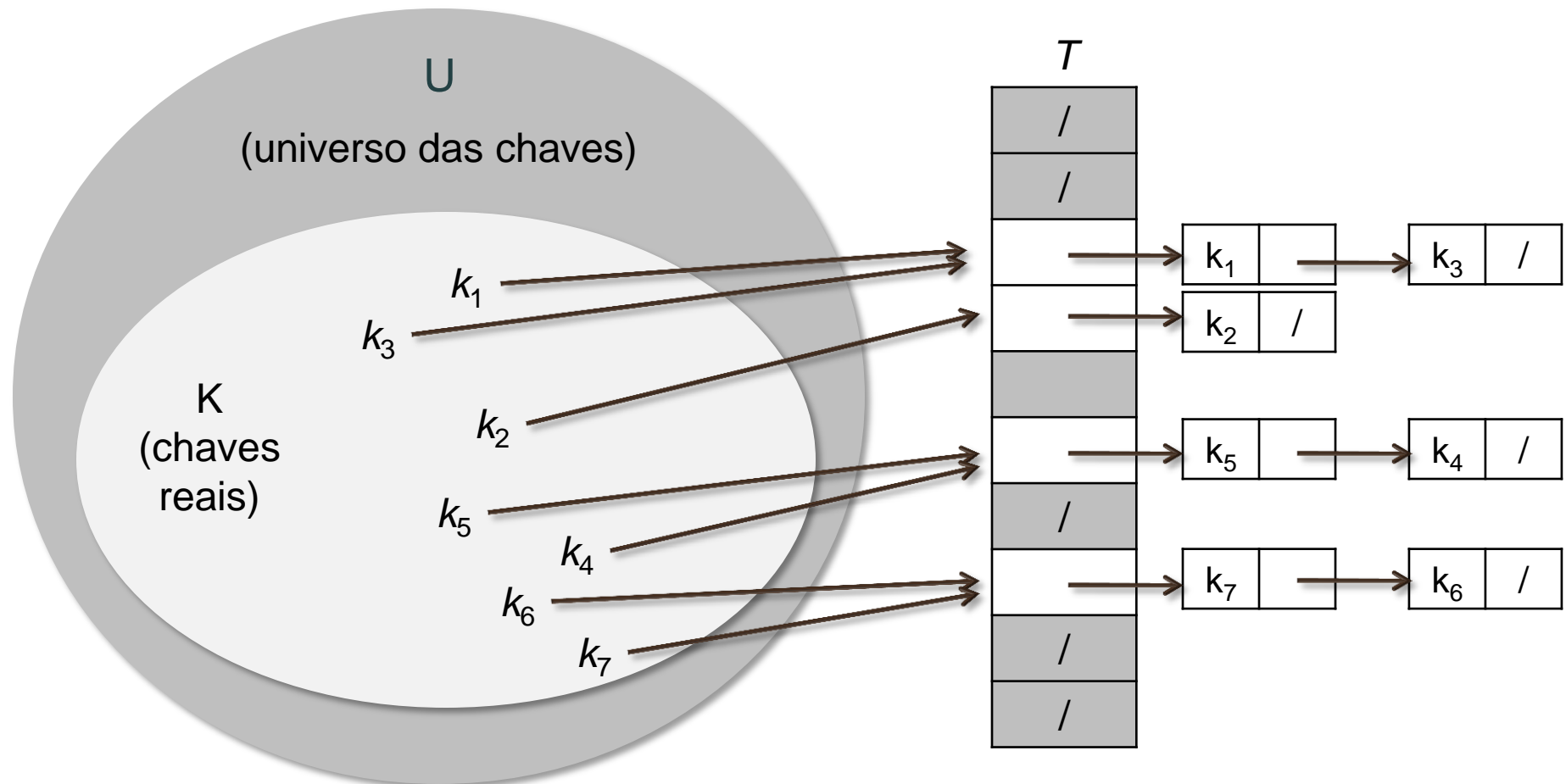
# Resolvendo colisões

- O desejável seria que a função fosse injetora
  - Evitaria colisões
  - Na prática isso é muito difícil
- Esquemas para trabalhar colisões
  - Endereçamento fechado
  - Endereçamento aberto

# Endereçamento fechado

- Forma simples de tratamento de colisões
  - Listas encadeadas
  - Cada entrada  $T[i]$  da **tabela hash** é uma lista encadeada, cujos elementos possuem **valor hash  $i$**
- Inserção do elemento  $k$ 
  - Calcule o valor hash  $i$  da chave do elemento  $k$
  - Insira o elemento  $k$  na lista encadeada  $T[i]$

# Endereçamento fechado



# Problemas

- Função hash bem escolhida promove um bom balanceamento
  - Não há como garantir que as listas terão tamanhos próximos
  - Listas podem ficar grandes
  - Perda de eficiência nas listas grandes
- Como melhorar?
  - Substituir a lista ligada por árvores balanceadas?
  - Na prática não se faz

# Endereçamento aberto

- Guarda todas as chaves na tabela
  - Mesmo quando ocorre colisão
  - $T[i]$  contém uma chave, ao invés da lista encadeada
  - Não usa espaço extra
  - Em caso de colisão, um novo endereço é computado
  - Esse processo é chamado rehashing

# Sondagem linear

- Forma mais simples de rehashing
  - Se  $h(k) = z$  e a posição  $T[z]$  estiver ocupada
  - Então a próxima posição disponível na tabela  $T$  será ocupada pela chave  $k$

$$h'(k, i) = (h(k) + i) \bmod m$$

- para  $i = 0, 1, \dots, m-1$



# Sondagem linear

- Funcionamento
  - Na primeira tentativa, aplicamos  $h'(k,0)$  , cujo resultado será o próprio  $h(k)$  original,
  - Se  $T[h'(k,0)]$  estiver ocupada, procuramos  $T[h'(k,1)]$
  - Se  $T[h'(k,1)]$  estiver ocupada, procuramos  $T[h'(k,2)]$
  - Se  $T[h'(k,2)]$  estiver ocupada, procuramos  $T[h'(k,3)]$
  - ...
- Note que
  - $T[h'(k,1)] = T[h'(k,0) + 1]$
  - $T[h'(k,2)] = T[h'(k,1) + 1]$
  - $T[h'(k,3)] = T[h'(k,2) + 1]$
  - ...

# Um exemplo

- Suponha que
  - O universo de chaves: números inteiros de quatro dígitos
  - Deseja-se mapeá-los na tabela hash  $T[0...10]$
- Função hash ( $m = 11$ )

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

# Inserção

HASH-INSERT ( $T, k$ )

```
1:  $i \leftarrow 0$ ;  
2: do  $j \leftarrow h'(k, i)$   
3:   if  $T[j] = \text{NIL}$  then  
4:      $T[j] \leftarrow k$ ;  
5:     return  $j$ ;  
6:   else  
7:      $i \leftarrow i+1$ ;  
8:   end if  
9: until  $i = m$ ;  
10: error "overflow";
```

Qual a complexidade?



# Busca

HASH-SEARCH ( $T, k$ )

```
1:  $i \leftarrow 0$ ;  
2: do  $j \leftarrow h'(k, i)$   
3:   if  $T[j] = k$  then  
4:     return  $j$ ;  
5:   end if  
6:    $i \leftarrow i+1$ ;  
8:   end if  
9: until  $T[j] = \text{NIL}$  or  $i = m$ ;  
10: return NIL;
```

Qual a complexidade?



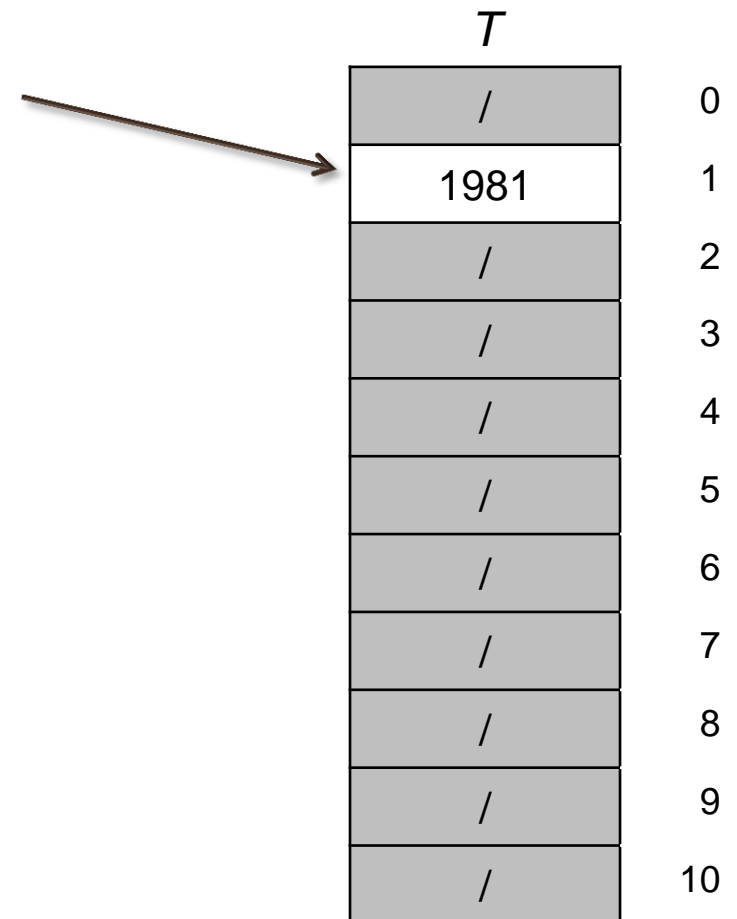
# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

<i>T</i>	
/	0
/	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$



$T$	
/	0
1981	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

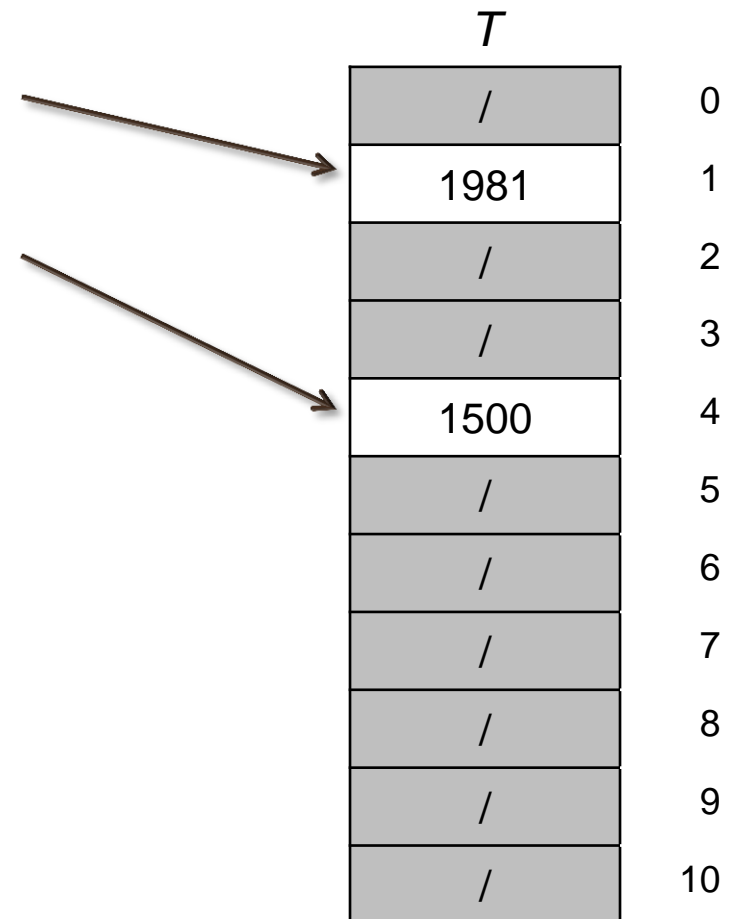
$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$T$	
/	0
1981	1
/	2
/	3
/	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$



$T$	
/	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
/	9
/	10

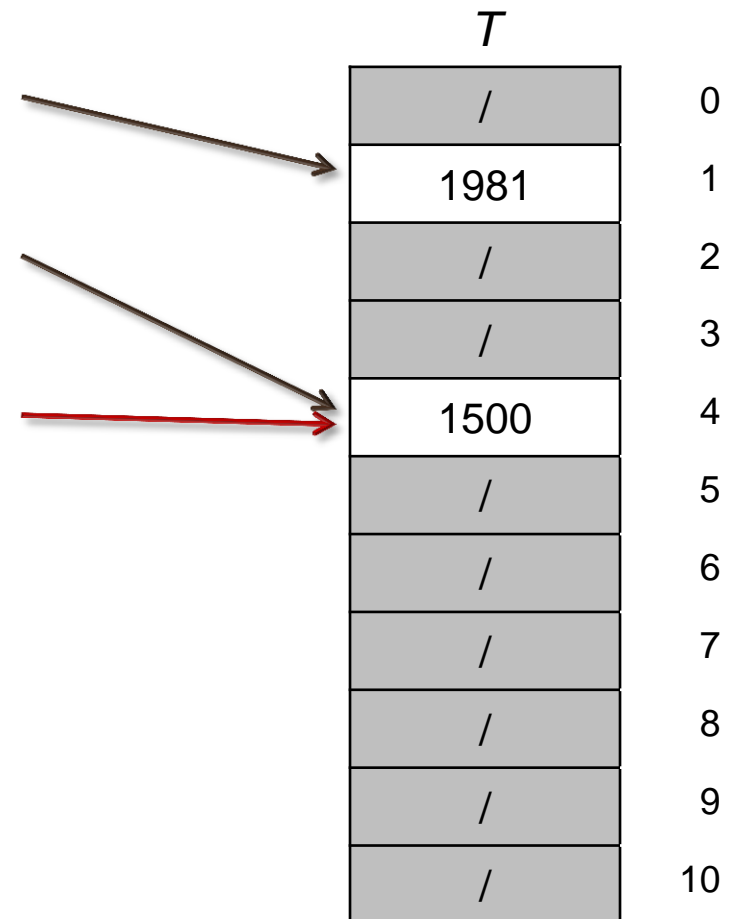


# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$



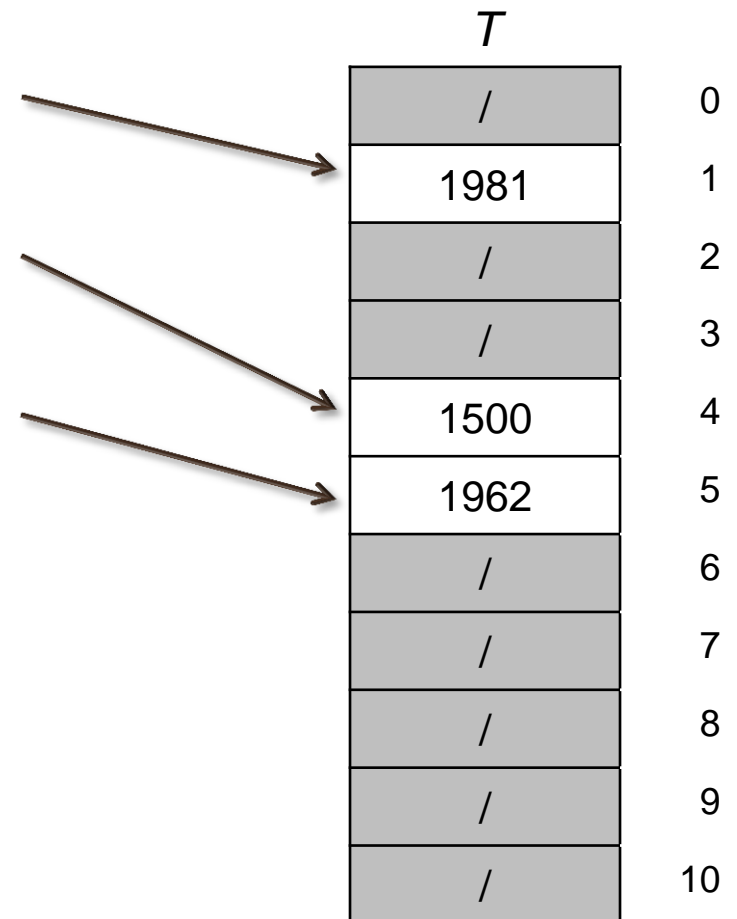
/	0
1981	1
/	2
/	3
1500	4
/	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$



$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
/	6
/	7
/	8
/	9
/	10

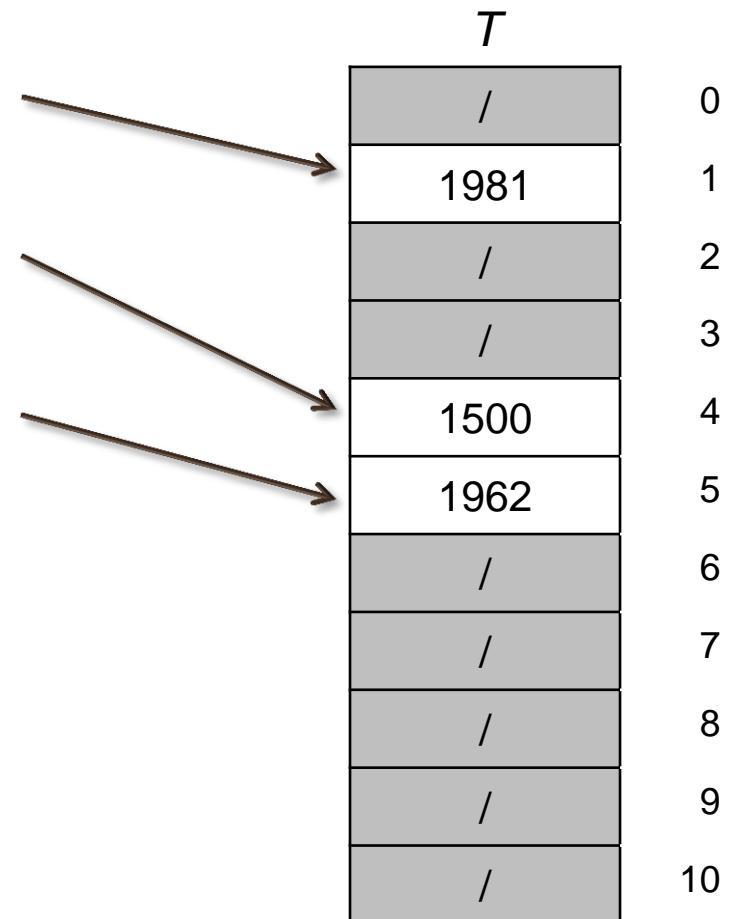
# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$



$T$		
/		0
1981		1
/		2
/		3
1500		4
1962		5
/		6
/		7
/		8
/		9
/		10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

<i>T</i>	
/	0
1981	1
/	2
/	3
1500	4
1962	5
/	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

<i>T</i>	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
/	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
1941	7
/	8
/	9
/	10



# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$$k = 2007 \rightarrow h(2007) = 2007 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
1941	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$$k = 2007 \rightarrow h(2007) = 2007 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
1941	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$$k = 2007 \rightarrow h(2007) = 2007 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
1941	7
/	8
/	9
/	10

# Um exemplo

$$k = 1981 \rightarrow h(1981) = 1981 \bmod 11 = 1$$

$$k = 1500 \rightarrow h(1500) = 1500 \bmod 11 = 4$$

$$k = 1962 \rightarrow h(1962) = 1962 \bmod 11 = 4$$

$$k = 1809 \rightarrow h(1809) = 1809 \bmod 11 = 5$$

$$k = 1941 \rightarrow h(1941) = 1941 \bmod 11 = 5$$

$$k = 2007 \rightarrow h(2007) = 2007 \bmod 11 = 5$$

$T$	
/	0
1981	1
/	2
/	3
1500	4
1962	5
1809	6
1941	7
2007	8
/	9
/	10

# Problemas

- Problemas de colisão
  - Quando a tabela está com alta taxa de ocupação, as colisões são mais freqüentes
  - Colisões afetam a busca
- Estratégia para desempenho
  - Manter uma taxa de ocupação abaixo de 50%

# Sondagem quadrática

- Sondagem quadrática

$$h'(k,i) = (h(k) + c_1 i + c_2 i^2) \bmod m$$

- para  $i = 0, 1, \dots, m-1$ ;  $c_1 \neq 0$  e  $c_2 \neq 0$

# Hash duplo

- Hash duplo (duas funções hash)

$$h'(k,i) = (h_1(k) + h_2(k) i) \bmod m$$

- para  $i = 0, 1, \dots, m-1$ ;
- Uma possibilidade
  - $h_1(k) = k \bmod m$
  - $h_2(k) = 1 + (k \bmod (m-1))$

# Inserção e busca iguais

HASH-INSERT ( $T, k$ )

```
1:  $i \leftarrow 0$ ;  
2: do  $j \leftarrow h'(k, i)$   
3:   if  $T[j] = \text{NIL}$  then  
4:      $T[j] \leftarrow k$ ;  
5:     return  $j$ ;  
6:   else  
7:      $i \leftarrow i+1$ ;  
8:   end if  
9: until  $i = m$ ;  
10: error "overflow";
```

HASH-SEARCH ( $T, k$ )

```
1:  $i \leftarrow 0$ ;  
2: do  $j \leftarrow h'(k, i)$   
3:   if  $T[j] = k$  then  
4:     return  $j$ ;  
5:   end if  
6:    $i \leftarrow i+1$ ;  
7:   end if  
8: until  $T[j] = \text{NIL}$  or  $i = m$ ;  
9: return  $\text{NIL}$ ;
```



# Exercício

- Considere um conjunto dinâmico  $S$ , representado por uma tabela de endereço direto  $T$  de comprimento  $m$ . Descreva um algoritmo para encontrar o maior elemento de  $S$ . Qual o desempenho no pior caso?
- Demonstre a inserção das chaves 5, 28, 19, 15, 20, 33, 12, 17, 10 em uma tabela hash com colisões resolvidas por encadeamento. Seja a tabela com 9 posições, e seja a função hash  $h(k) = k \bmod 9$ .
- Considere a inserção das chaves 10, 22, 31, 4, 15, 28, 17, 88, 59 em uma tabela hash de comprimento  $m = 11$  usando o endereçamento aberto com a função hash primário  $h(k) = k \bmod m$ . Ilustre o resultado da inserção dessas chaves com o uso da sondagem linear, sondagem quadrática (usando  $c_1 = 1$  e  $c_2 = 3$ ) e hash duplo com as funções sugeridas na aula.