Hash

Definição, endereçamento aberto e encadeamento

Prof. Edson Alves - UnB/FGA

2019

Sumário

Definição de hash

Motivação para a criação da hash

- Estruturas lineares permitem N armazenar elementos sem que o valor de N seja conhecido *a priori*, em tempo de compilação
- Estas estruturas, poém, não são eficientes na busca do elementos (complexidade $O(N)$)
- As árvores de busca, como as estruturas lineares, também permitem o armazenamento de um número arbitrário de elementos (limitado somente pela memória disponível)
- Em árvores busca perfeitamente balanceadas a ordem de complexidade da busca é $O(\log N)$
- Porém a organização de memória das árvores não é contígua, levando à perda de eficiência em relação à *cache*
- A idéia da *hash* é deduzir o índice de um elemento em um vetor a partir apenas da informação armazenada pelo elemento, reduzindo a ordem de complexidade busca para $O(1)$

Hash

Definição

Uma função h é uma função de *hash* se ela transforma uma chave K no índice do elemento que contém K na tabela. Se h transforma chaves distintas em índices distintos, ela é uma função de *hash* perfeita.

- Para se criar uma função h de *hash* perfeita, a tabela deve conter, no mínimo, o número de elementos cujas chaves serão transformadas pela função h
- Uma colisão ocorre se duas chaves distintas K_1 e K_2 gerarem o mesmo índice, isto é, se $h(K_1) = h(K_2)$ com $K_1 \neq K_2$
- A idéia é encontrar uma função h que gere o mínimo de colisões mas que não seja sofisticada ao ponto de seu cálculo interferir na performance do programa

Exemplos de funções *hash*

Soma dos elementos

Chave	Uma string K
Algoritmo	Atribui-se um código numérico para cada um dos caracteres que aparecem na string K e somam-se estes valores
Nível de colisão	Alto

```cpp
int h(const string& K)
{
    int v = 0;

    for (const auto& c : K)
        v += c;

    return v;
}
```

Resto da divisão

Chave	Um inteiro K
Algoritmo	Obtêm-se o resto da divisão da chave K pelo tamanho T da tabela. De preferência, T deve ser um número primo
Nível de colisão	Inversamente proporcional a T

```c
unsigned long h(unsigned long K, size_t T)
{
        return K % T;
}
```

Enlaçamento deslocado

Chave	Uma string K
Algoritmo	Divide-se a string em N partes de, no máximo, m caracteres, e computa-se a chave aplicando-se a operação XOR em todas as partes
Nível de colisão	Médio

```
unsigned long h(const string& S) {
    unsigned long v = 0, p = 0, m = 4, i = 0;

    for (const auto& c : S) {
        p |= (c << 8*i++);

        if (i == m) v ^= p, i = p = 0;
    }

    return v ^ p;
}
```

Enlaçamento no limite

Chave	Uma string K
Algoritmo	Variante do enlaçamento deslocado. Divide-se a chave eme 3 partes, e se enlaça os extremos com a parte do meio invertida
Nível de colisão	Médio

```cpp
unsigned long h(const string& S) {
    string s(S);

    while (s.size() % 3) s.push_back(0);

    unsigned long v = 0, N = s.size(), M = N/3;

    for (size_t i = 0; i < M; ++i)
        v ^= (s[i] ^ s[2*M - 1 - i] ^ s[2*M + i]);

    return v;
}
```

Meio quadrado

Chave	Um inteiro K
Algoritmo	Eleva-se K ao quadrado e toma-se a parte central do resultado
Nível de colisão	Médio

```
unsigned long h(int K)
{
        auto s = K*K;

        return (s & 0x00FFFF00) >> 8;
}
```

Polinomial

Chave	Uma string K com N caracteres
Algoritmo	$g(x)$ é um polinômio de grau $N-1$ com coeficientes $a_i = K[i]$ e $h(K) = g(p) \pmod{T}$, onde T é o tamanho da tabela e $p \neq T$ é primo
Nível de colisão	Baixo

```
unsigned long h(const string& K, size_t p, size_t T)
{
    unsigned long h = 0;

    for (int i = K.size() - 1; i >= 0; --i)
    {
        h = (h * p) % T;
        h = (h + K[i]) % T;
    }

    return h;
}
```

Endereçamento aberto

Problemas com a colisão

- Como visto, as funções de *hash* podem gerar colisões, isto é, um mesmo índice para duas chaves distintas
- Naturalmente surge o seguinte questionamento: como inserir duas chaves que colidem em uma mesma tabela, e como resgatá-las em uma busca?
- Uma alternativa para o tratamento de colisões é o endereçamento aberto

Endereçamento aberto

Definição

Se a chave K for mapeada para uma posição já ocupada da tabela, o endereçamento aberto utiliza a sequência de sondagem

$$N(h(K)+p(1)), N(h(K)+p(2)), \ldots, N(h(K)+p(i)), \ldots$$

onde p é a função de sondagem, i é o índice de sondagem e N a função de normalização, até que

1. se encontre uma posição desocupada
2. $N(h(K)+p(j)) = N(h(K))$
3. se verifique que a tabela está cheia

Sondagem linear

- Na sondagem linear, temos a função de sondagem é a identidade, isto é, $p(i) = i, \forall i$
- A função de normalização faz com que o índice resultante esteja dentro dos limites da tabela, usando o resto da divisão:

$$N(K) = K (\text{mod } T),$$

onde T é o tamanho da tabela

- Se uma posição $N(h(K) + p(i))$ já estiver ocupada, tenta-se o próximo índice de sondagem $(i + 1)$ até que se encontre um espaço vago ou ocorra uma das outras condições
- Esta estratégia tende a formação de agrupamentos de chaves, com pontos de acumulação na tabela e intervalos contíguos não ocupados

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 51

0	1	2	3	4	5	6	7	8	9	10

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 51

$$h(51) = 51 \pmod{11} = 7$$

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 16

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 16

$$h(16) = 16 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 76

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 76

$$h(76) = 76 \pmod{11} = 10$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 35

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 35

$$h(35) = 35 \pmod{11} = 2$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: -6

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: -6

$$h(-6) = -6 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$
Elemento a ser inserido: -6

$$N(h(-6) + 1) = (5 + 1) \pmod{11} = 6$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 49

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 49

$$h(49) = 49 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 49

$$N(h(49) + 1) = (5 + 1) \pmod{11} = 6$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 49

$$N(h(49) + 2) = (5 + 2) \pmod{11} = 7$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem linear

Sondagem linear, $T = 11$

Elemento a ser inserido: 49

$$N(h(49) + 3) = (5 + 3) \pmod{11} = 8$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51	49		76

Implementação da sondagem linear

```cpp
#include <bits/stdc++.h>

using namespace std;

template<typename I, size_t T>
class HashSet {
private:
    size_t mod(const I& a, int b)
    {
        return ((a % b) + b) % b;
    }

    size_t h(const I& K) { return mod(K, T); }

    size_t N(const I& K, size_t i) { return mod(h(K) + i, T); }

    vector<I> xs;
    bitset<T> used;

public:
    HashSet() : xs(T) {}
```

Implementação da sondagem linear

```
    bool insert(const I& K)
    {
        if (used.count() == T)
            return false;

         for (size_t i = 0; i < T; ++i)
         {
            auto pos = N(K, i);

            if (not used[pos])
            {
                xs[pos] = K;
                used[pos] = true;
                break;
            }
         }

         return true;
    }
```

Sondagem quadrática

- Na sondagem quadrática, a função de sondagem é dada por

$$p(i) = (-1)^{i-1}\left\lfloor \frac{i+1}{2} \right\rfloor^2,$$

  para $i = 1, 2, \ldots, T-1$
- A função de normalização é dada por $N(K) = K \pmod T$, onde T é o tamanho da tabela
- A sondagem quadrática pode ser interpretada como a sequência

$$h(K) + i^2, h(K) - i^2, h(K) + (i+1)^2, h(K) - (i+1)^2, \ldots$$

  para $i = 1, 2, \ldots, (T-1)/2$
- Se T for um número primo da forma $4k+3$, a sequência acima passa por todas as posições da tabela (Radke, 1970)

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 51

0	1	2	3	4	5	6	7	8	9	10

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 51

$$h(51) = 51 \pmod{11} = 7$$

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 16

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 16

$$h(16) = 16 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 76

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 76

$$h(76) = 76 \pmod{11} = 10$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 35

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 35

$$h(35) = 35 \pmod{11} = 2$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: -6

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: -6

$$h(-6) = -6 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: -6

$$N(h(-6) + 1^2) = (5 + 1) \pmod{11} = 6$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 49

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 49

$$h(49) = 49 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	−6	51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 49

$$N(h(49) + 1^2) = (5 + 1) \pmod{11} = 6$$

0	1	2	3	4	5	6	7	8	9	10
		35			16	-6	51			76

Exemplo de inserção utilizando sondagem quadrática

Sondagem quadrática, $T = 11$

Elemento a ser inserido: 49

$$N(h(49) - 1^2) = (5 - 1) \pmod{11} = 4$$

0	1	2	3	4	5	6	7	8	9	10
		35		49	16	-6	51			76

Implementação da sondagem quadrática

```
#include <bits/stdc++.h>

using namespace std;

template<typename I, size_t T>
class HashSet {
private:
    size_t mod(const I& a, int b) { return ((a % b) + b) % b; }

    size_t h(const I& K) { return mod(K, T); }

    size_t N(const I& K, size_t i)
    {
        auto sign = i % 2 ? 1 : -1;
        auto j = (i + 1)/2;

        return mod(h(K) + sign * j * j, T);
    }

    vector<I> xs;
    bitset<T> used;
```

Implementação da sondagem quadrática

```
public:
    HashSet() : xs(T) {}

    bool insert(const I& K)
    {
       if (used.count() == T) return false;

        for (size_t i = 0; i < T; ++i)
        {
            auto pos = N(K, i);

            if (not used[pos])
            {
                xs[pos] = K;
                used[pos] = true;
                break;
            }
        }

        return true;
    }
```

hash duplo

- O *hash* duplo é uma das melhores estratégias de endereçamento aberto
- Isto porque a sequência de sondagem gerada tem muitas das características das sequências aleatórias
- No *hash* duplo a sequência de sondagem tem a forma

$$h(K) = (h_1(K) + ih_2(K)) \pmod{T}, \qquad i = 0, 1, 2, \ldots, T-1$$

  onde $h_1(K), h_2(K)$ são duas funções de *hash* auxiliares, i é o índice de sondagem e T é o tamanho da tabela
- Diferentemente das sondagens lineares e quadráticas, a função $h_2(K)$ depende do valor do chave K
- Deste modo, as sequências de sondagem para chaves $K_1 \neq K_2$, com $h_1(K_1) = h_1(K_2)$, tendem a serem diferentes

hash duplo

- A função $h_2(K)$ deve gerar valores co-primos com o tamanho T da tabela
- Se T é uma potência de dois (isto é, $T = 2^k$ para algum k positivo), a função $h_2(K)$ deve gerar apenas números ímpares
- Se T é um número primo, $h_2(K)$ tem que gerar números positivos menores do que T
- Uma maneira de se obter isso é fazer $h_1(K) = K \pmod{T}$ e $h_2(K) = 1 + (K \pmod{T-1})$
- O *hash* duplo tem melhor desempenho do que a sondagem linear e a sondagem quadrática, pois gera um número $O(T^2)$ de sondagens possíveis, enquanto que as outras duas geram $O(T)$ sequências de sondagem

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 51

0	1	2	3	4	5	6	7	8	9	10

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 51

$$h(51) = h_1(51) = 51 \pmod{11} = 7$$

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 16

0	1	2	3	4	5	6	7	8	9	10
							51			

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 16

$$h(16) = h_1(16) = 16 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 76

0	1	2	3	4	5	6	7	8	9	10
					16		51			

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 76

$$h(76) = h_1(76) = 76 \pmod{11} = 10$$

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 35

0	1	2	3	4	5	6	7	8	9	10
					16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 35

$$h(35) = h_1(35) = 35 \pmod{11} = 2$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Hash duplo, $T = 11$

Elemento a ser inserido: -6

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: -6

$$h(-6) = h_1(-6) = -6 \pmod{11} = 5$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: -6

$$h(-6) = (h_1(-6) + h_2(-6)) \pmod{11} = 5 + 5 = 10$$

0	1	2	3	4	5	6	7	8	9	10
		35			16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: -6

$$h(-6) = (h_1(-6) + 2h_2(-6)) \pmod{11} = 5 + 10 \pmod{11} = 4$$

0	1	2	3	4	5	6	7	8	9	10
		35		-6	16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 49

0	1	2	3	4	5	6	7	8	9	10
		35		-6	16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 49

$h(49) = h_1(49) \pmod{11} = 5$

0	1	2	3	4	5	6	7	8	9	10
		35		-6	16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 49

$$h(49) = (h_1(49) + h_2(49)) \pmod{11} = 5 + 10 \pmod{11} = 4$$

0	1	2	3	4	5	6	7	8	9	10
		35		-6	16		51			76

Exemplo de inserção utilizando hash duplo

Hash duplo, $T = 11$

Elemento a ser inserido: 49

$$h(49) = (h_1(49) + 2h_2(49)) \pmod{11} = 5 + 20 \pmod{11} = 3$$

0	1	2	3	4	5	6	7	8	9	10
		35	49	-6	16		51			76

Implementação do hash duplo

```
#include <bits/stdc++.h>

using namespace std;

template<typename I, size_t T>
class HashSet {
private:
    size_t mod(const I& a, int b) { return ((a % b) + b) % b; }

    size_t h1(const I& K) { return mod(K, T); }

    size_t h2(const I& K) { return 1 + mod(K, T - 1); }

    size_t N(const I& K, size_t i)
    {
        return mod(h1(K) + i*h2(K), T);
    }

    vector<I> xs;
    bitset<T> used;

```

Implementação do hash duplo

```
    HashSet() : xs(T) {}

    bool insert(const I& K)
    {
       if (used.count() == T) return false;

        for (size_t i = 0; i < T; ++i)
        {
            auto pos = N(K, i);

            if (not used[pos])
            {
                xs[pos] = K;
                used[pos] = true;
                break;
            }
        }

        return true;
    }

```

Encadeamento

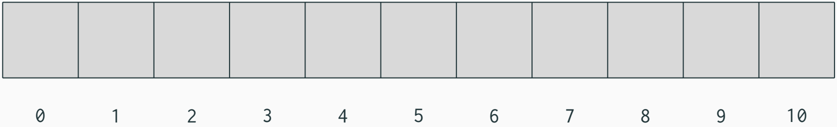
Resolução de colisão por encadeamento

- Uma outra forma de se tratar as colisões é o uso do encadeamento
- A ideia é que cada célula da tabela corresponda a uma lista duplamente encadeada
- Cada chave K tal que $h(K) = j$, onde $j = 0, 1, \ldots, T - 1$, é adicionada à lista que ocupa a posição j
- O uso de listas duplamente encadeadas permite uma remoção mais eficiente
- Se a opção de remoção não for implementada, uma lista simplesmente encadeada ou um vector também podem ser utilizados
- Deve-se tomar cuidado, porém, porque no pior caso todas as chaves colidem em uma mesma posição, de modo que a busca e a inserção passam a ter complexidade $O(N)$, porém ocupando mais espaço em memória que uma única lista

Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

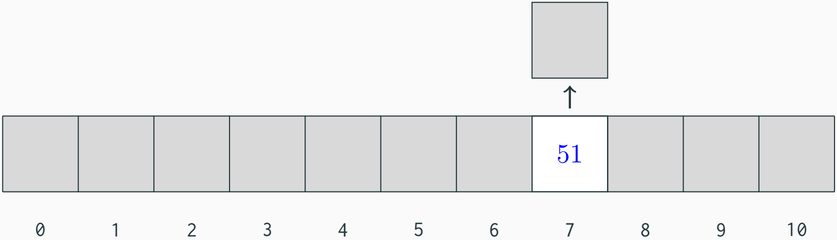
Elemento a ser inserido: 51



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

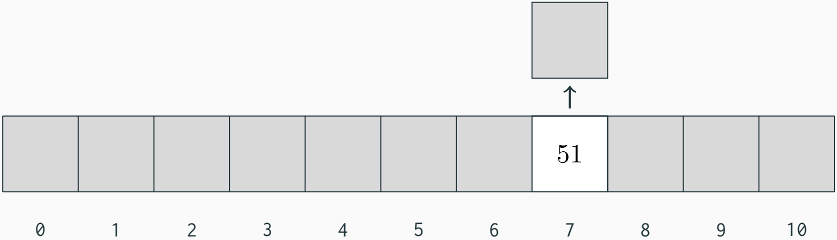
Elemento a ser inserido: 51



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

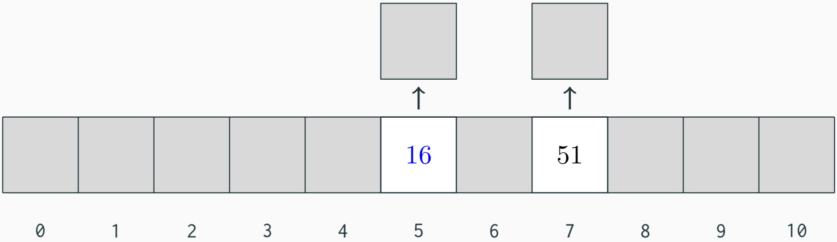
Elemento a ser inserido: 16



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

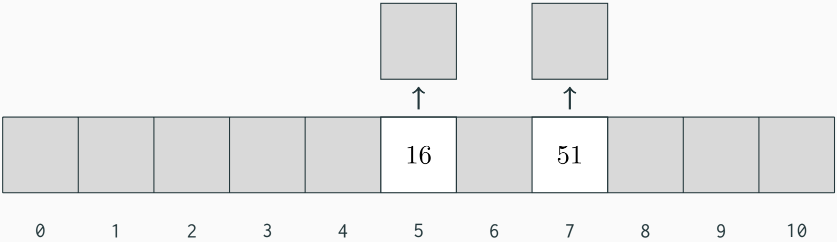
Elemento a ser inserido: 16



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

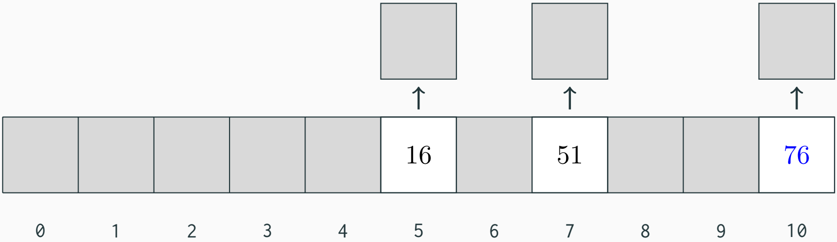
Elemento a ser inserido: 76



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

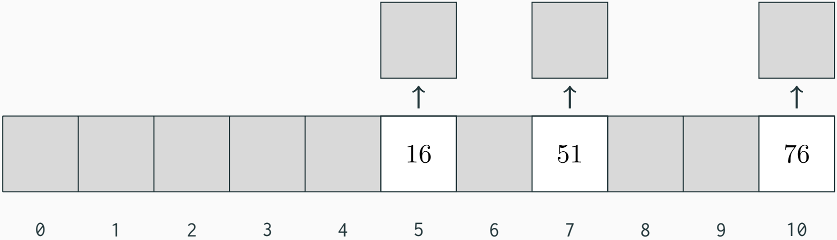
Elemento a ser inserido: 76



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

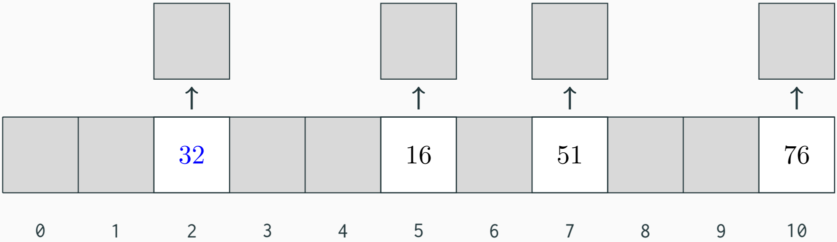
Elemento a ser inserido: 32



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

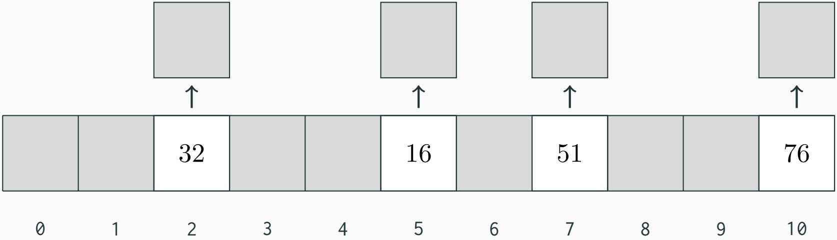
Elemento a ser inserido: 32



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

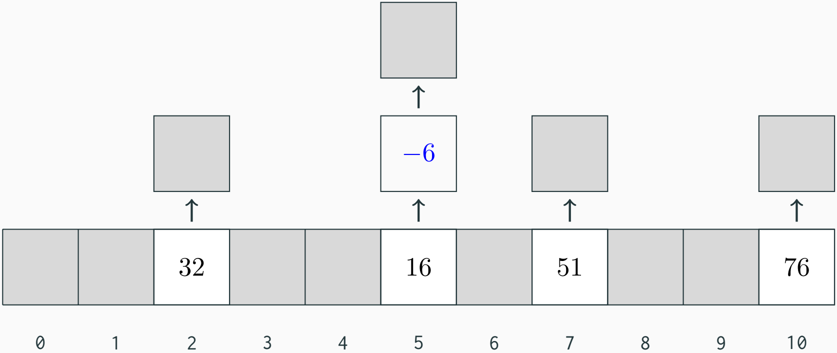
Elemento a ser inserido: -6



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

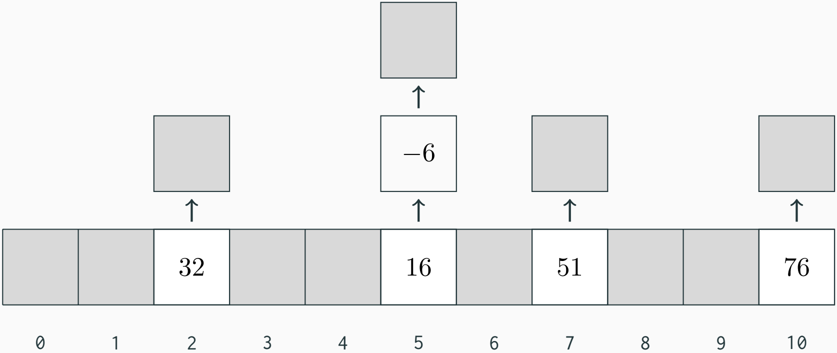
Elemento a ser inserido: -6



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

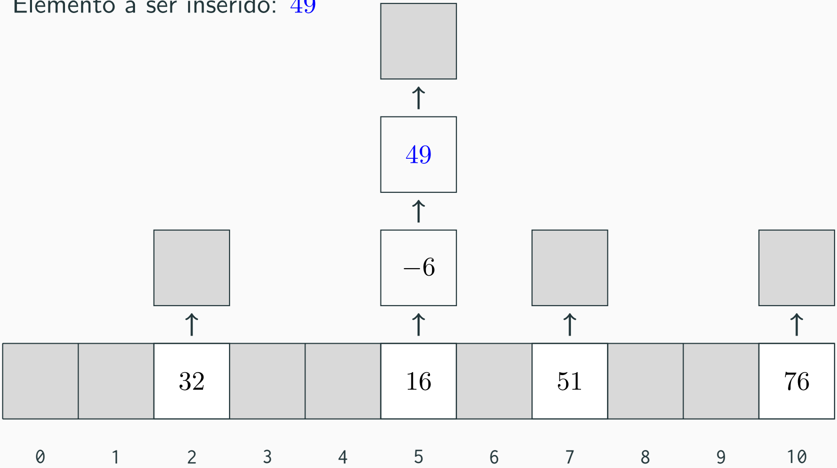
Elemento a ser inserido: 49



Exemplo de inserção utilizando encadeamento

Encadeamento, $T = 11$

Elemento a ser inserido: 49



Exemplo de implementação do encadeamento

```cpp
#include <bits/stdc++.h>

using namespace std;

template<typename I, size_t T>
class HashSet {
private:
    size_t mod(const I& a, int b) { return ((a % b) + b) % b; }

    size_t h(const I& K) { return mod(K, T); }

    vector<list<I>> xs;

public:
    HashSet() : xs(T) {}

    void insert(const I& K)
    {
        xs[h(K)].push_back(K);
    }

```

Referências

1. **CORMEN**, Thomas H.; **LEISERSON**, Charles E.; **RIVEST**, Ronald L.; **STEIN**, Clifford. *Introduction to Algorithms*, The MIT Press, 3rd edition, 2009.
2. **DROZDEK**, Adam. *Algoritmos e Estruturas de Dados em C++*, 2002.
3. **RADKE**, Charles E. *The Use of Quadratic Residue Research*, Communications of the ACM, volume 13, issue 2, pg 103–105, 1970[1].
4. **STROUSTROUP**, Bjarne. *The C++ Programming Language*, 2013.
5. C++ Reference[2].

[1] https://dl.acm.org/citation.cfm?id=362036
[2] https://en.cppreference.com/w/