


Mapas

Um mapa é uma estrutura associativa na qual os elementos armazenados estão organizados na forma de pares (chave, valor), de tal forma que é possível ter acesso a valor (que pode eventualmente ser um objeto com uma estrutura complexa) a partir da chave. Conceitualmente, um mapa pode ser visualizado como uma tabela (Figura 3.1). Assim, é possível obter o valor def a partir da especificação da chave β , ou o valor xyz a partir de ζ .

Figura 3.1 Visão conceitual de uma tabela.

chave	valor
α	abc
β	def
γ	ghi
	
ζ	xyz

Exemplo de aplicação

Uma tabela de símbolos, utilizada em compiladores para manter a informação sobre cada variável de um programa, é um exemplo de uma possível aplicação de uma estrutura do tipo mapa. Neste caso, a chave é usualmente o nome de uma variável e o valor é o conjunto de informações sobre a variável, tais como o seu tipo, endereço de memória e local de definição.

Já em uma tabela de arquivos, uma estrutura utilizada pelo sistema operacional para manter a informação sobre cada um dos arquivos associados a um processo, a chave pode ser um identificador inteiro (por exemplo, o terceiro arquivo aberto pelo processo) e o valor o conjunto de informações sobre o arquivo, tais como a posição corrente no arquivo.

Implementação

Em C++, a biblioteca STL implementa mapas com a classe `map`. A declaração de uma coleção deste tipo especifica dois parâmetros, o primeiro para o tipo da chave e o segundo para o tipo do valor:

```
#include <map>
```

```
#include <string>
```

```
...
```

```
map<string,int> compras;
```

Neste exemplo, a chave é uma string e o valor associado é um inteiro.

Para operar com os elementos de um mapa, o operador [] é sobrecarregado. Por exemplo, para associar o valor 6 à chave vinho, basta utilizar a expressão:

```
compras["vinho"] = 6;
```

Do mesmo modo, para obter o valor associado a uma chave, o mesmo operador pode ser utilizado:

```
int quantidade = compras["vinho"];
```

Além da sobrecarga desse operador, os métodos básicos presentes nos outros tipos de estrutura (como empty e size) também estão disponíveis para mapas.

Iteradores para coleções do tipo map percorrem elementos que são pares, com o primeiro elemento correspondente à chave e o segundo, ao valor:

```
map<string,int>::iterator itm;  
  
itm = compras.begin();  
  
while (itm != compras.end()) {  
    cout << (*itm).first << ": ";  
    cout << (*itm).second << endl;  
    itm++;  
}
```

A estrutura map não admite duplicação de chaves; STL oferece também uma implementação multimap, na qual as chaves podem ser repetidas:

```
#include <map>  
  
#include <string>  
  
...  
  
multimap<string,int> mm;
```

Como para multiset, count retorna a quantidade de elementos com a chave especificada e equal_range retorna um par de iteradores para o segmento da coleção que contém os elementos com a mesma chave, especificada como argumento.

Resumo:

- Um mapa armazena pares (chave, valor) chamados **itens**
 - Chaves e valores podem ser de qualquer tipo
 - Elemento e Valor são sinônimos
- A chave é utilizada para achar um elemento rapidamente
 - Estruturas especiais são usadas para que a pesquisa seja rápida
- Diz-se, portanto, que um mapa "**mapeia chaves para valores**"
- O Mapa pode ser mantido **ordenado** ou não (com respeito às chaves)
- Normalmente implementada como "Tabela Hash" ou "Árvore"
 - Esses dois tipos de estrutura de dados foram vistos na disciplina de ED II
- As operações mais importantes de uma coleção do tipo Conjunto são:
 - Adição de elementos
 - Adicionar um item no mapa (fornecendo chave e valor)
 - Remoção de elementos
 - Remover um item com chave dada
 - Acesso aos elementos
 - Iterar sobre os itens
 - Pesquisa de elementos
 - Descobrir se um elemento com chave dada está na coleção
 - Indagar sobre atributos
 - Obter o número de elementos
- Observe que o acesso à coleção sempre é feita **conhecendo a chave**

Exercícios:

1. Proponha uma estrutura de dados em C++ para armazenar cada entrada de um mapa, suponha que a chave seja uma string para guardar a placa do carro e o valor guarde os dados do carro (modelo, ano, cor).
2. Proponha uma estrutura de dados em C++ para representar um mapa utilizando uma lista ligada simples e a estrutura de entrada (nó) definida anteriormente.
3. Implemente o método de busca no mapa.
4. Implemente o método de inserção no mapa. O método deve inserir somente se o dado ainda não existir na lista.
5. Implemente o método de remoção no mapa.