Chapter 27 Hashing

Why Hashing?

The preceding chapters introduced search trees. An element can be found in $O(\log n)$ time in a well-balanced search tree. Is there a more efficient way to search for an element in a container? This chapter introduces a technique called hashing. You can use hashing to implement a map or a set to search, insert, and delete an element in $O(1)$ time.

Objectives

✦ To know what hashing is for (§27.3).
✦ To obtain the hash code for an object and design the hash function to map a key to an index (§27.4).
✦ To handle collisions using open addressing (§27.5).
✦ To know the differences among linear probing, quadratic probing, and double hashing (§27.5).
✦ To handle collisions using separate chaining (§27.6).
✦ To understand the load factor and the need for rehashing (§27.7).
✦ To implement MyHashMap using hashing (§27.8).

Map

A map is a data structure that stores entries. Each entry contains two parts: key and value. The key is also called a search key, which is used to search for the corresponding value. For example, a dictionary can be stored in a map, where the words are the keys and the definitions of the words are the values.

A map is also called a dictionary, a hash table, or an associative array. The new trend is to use the term map.
What is Hashing?

If you know the index of an element in the array, you can retrieve the element using the index in O(1) time. So, can we store the values in an array and use the key as the index to find the value? The answer is yes if you can map a key to an index.

The array that stores the values is called a hash table. The function that maps a key to an index in the hash table is called a hash function.

Hashing is a technique that retrieves the value using the index obtained from key without performing a search.

Hash Function and Hash Codes

A typical hash function first converts a search key to an integer value called a hash code, and then compresses the hash code into an index to the hash table.
Quadratic Probing

Quadratic probing can avoid the clustering problem in linear probing. Linear probing looks at the consecutive cells beginning at index \( k \). Quadratic probing increases the index by \( j^2 \) for \( j = 1, 2, 3, \ldots \). The actual index searched are \( k, k + 1, k + 4, \ldots \)

Double Hashing

Double hashing uses a secondary hash function on the keys to determine the increments to avoid the clustering problem.

\[ h'(k) = 7 - k \mod 7; \]

Handling Collisions Using Separate Chaining

The separate chaining scheme places all entries with the same hash index into the same location, rather than finding new locations. Each location in the separate chaining scheme is called a bucket. A bucket is a container that holds multiple entries.

Separate Chaining Animation
Implementing Map Using Hashing

MyMap

Run

Implementing Set Using Hashing

MySet

Run