STL

25 September 2024 19:59

STL stands for Standard Template Library. It is a powerful set of C++ template classes that provide general-purpose, reusable implementations of common data structures and algorithms.

Main Components of STL are:

Containers

These are data structures that store objects and data.

Sequence Containers:

- `vector` : Dynamic array that can grow in size.
- `deque` : Double-ended queue, allows insertion/deletion at both ends.
- `list` : Doubly linked list, allows fast insertion/deletion at any position.
- `array` : Static array with a fixed size (introduced in C++11).
- `forward_list` : Singly linked list, allows forward traversal only.

Associative Containers:

- `set` : Stores unique elements in sorted order.
- `multiset` : Similar to `set` but allows duplicate elements.
- `map` : Stores key-value pairs with unique keys in sorted order.
- `multimap` : Similar to `map` but allows duplicate keys.

Unordered Associative Containers (introduced in C++11):

- `unordered_set` : Stores unique elements, but the order is not guaranteed.
- `unordered_multiset` : Allows duplicates, but the order is not guaranteed.
- `unordered_map` : Stores key-value pairs, with no guaranteed order.
- `unordered_multimap` : Similar to `unordered_map` , but allows duplicate keys.

Iterators

Iterators are used to point to the elements of a container and traverse through

them. They work like pointers.

- **Input Iterator**: Can only read elements in a single pass.
- **Output Iterator**: Can only write elements in a single pass.
- **Forward Iterator**: Can read/write and traverse forward.
- **Bidirectional Iterator**: Can traverse both forward and backward.
- **Random Access Iterator**: Provides random access to elements (e.g., `vector`).

Algorithms

- `sort()`: Sorts the elements in a range.
- `find()`: Searches for an element in a range.
- `reverse()`: Reverses the order of elements in a range.
- `copy()`: Copies elements from one range to another.
- `count()`: Counts occurrences of a value in a range.

Utility Components

`pair`: A utility to store two values together.

`tuple`: A generalized version of `pair` that can store more than two values.

`function`: A wrapper for callable objects (introduced in C++11).