Using Queue in C++ Standard Template Library (STL)

1. Introduction

std::queue is a container adapter in the C++ Standard Template Library (STL) that operates on the **FIFO (First In, First Out)** principle. This means the first element added to the queue is the first one to be removed. In other words, it follows the *"first come, first served"* rule.

The setup of a queue is similar to that of a stack; however, their behaviors are opposite. While a stack uses **LIFO (Last In, First Out)**, a queue inserts elements at the back and removes them from the front.

2. Setup and initialization

To use a queue in C++, the header must be included:

```
#include <queue>
```

Declaring a Queue

The syntax to declare a queue is:

```
std::queue<type> queue_name;
```

Example:

```
std::queue<int> q;
```

3. Core operations

A queue restricts access to specific ends of the container. Only the front and back elements can be accessed; elements in the middle cannot be accessed directly.

3.1/ Adding items (push)

The **push(item)** function inserts a new element at the back of the queue.

Time Complexity: O(1)

Example:

```

```

```
q = [25, 12, 50, 43]
q.push(6)
q = [25, 12, 50, 43, 6]
```

3.2/ Accessing items (front and back)

front() Returns a reference to the first element (the one waiting the longest).

Time Complexity: O(1)

Undefined behavior if the queue is empty

back() Returns a reference to the last element (the most recently added).

Time Complexity: O(1)

Undefined behavior if the queue is empty

Example:

```
q = [25, 12, 50, 43]
q.front() = 25
q.back() = 43
```

3.3/ Removing items (pop)

The **pop()** function removes the element from the front of the queue.

Time Complexity: O(1)

The function returns void, so front() should be called before **pop()** if the removed value is needed.

Example:

```
q = [25, 12, 50, 43]
q.**pop()**
q = [12, 50, 43]
```

4. Status operations

empty() Returns true if the queue contains no elements.

size() Returns the number of elements currently in the queue.

5. Applications of queue

std::queue is commonly used in:

- Buffering data (e.g., keyboard input, network packets)
- Breadth-First Search (BFS) in graph algorithms
- CPU and task scheduling
- Service systems and line management

The queue guarantees that data is processed in the exact order it arrives.

6. Conclusion

The **std::queue** container is an essential STL data structure when ordered processing is required. Its strict FIFO behavior makes it ideal for real-world systems involving scheduling, buffering, and sequential task handling.