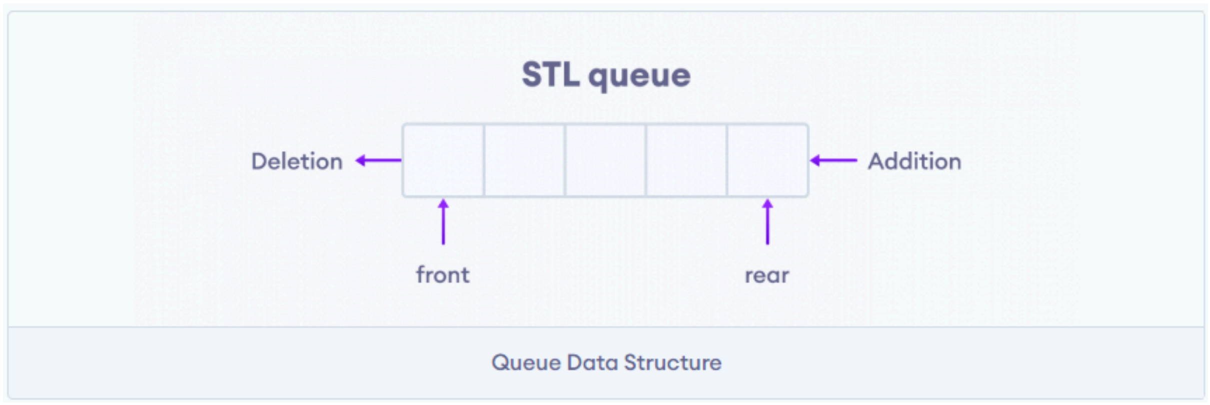
Queue

Introduction:

A queue is a linear data structure that follows the First-In-First-Out (FIFO) principle, where the first element added to the queue will be the first one to be removed. In C++, the Standard Template Library (STL) provides a queue class to implement queue operations efficiently.



Create C++ Queue:

In order to create a queue in C++, we first need to include the queue header file.

```
#include <queue>
```

Once we import this file, we can create a queue using the following syntax:

```
queue<type> q;
```

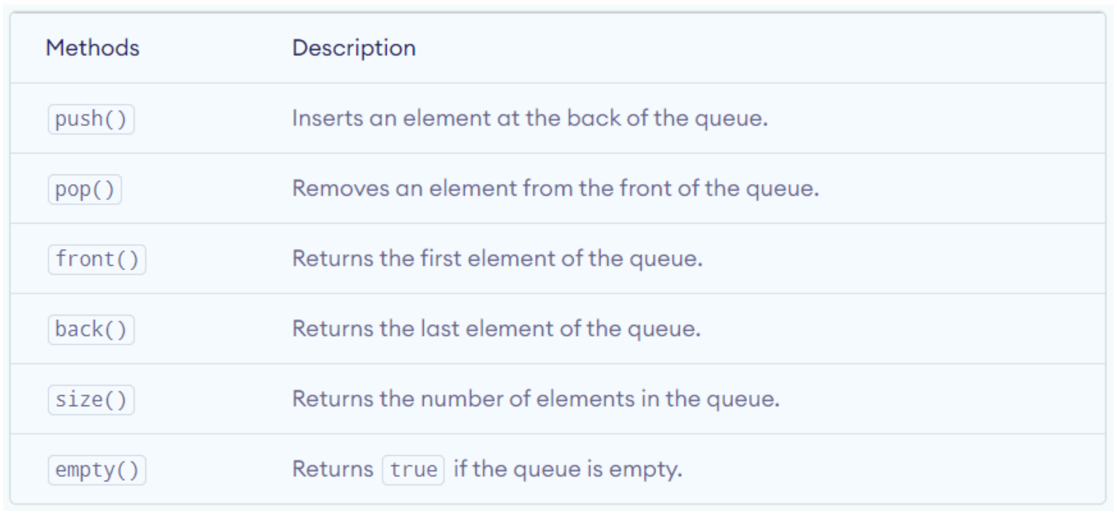
Here, type indicates the data type we want to store in the queue. For example,

```
// create a queue of integer data type
queue<int> integer_queue;

// create a queue of string data type
queue<string> string_queue;
```

C++ Queue Methods

In C++, queue is a class that provides various methods to perform different operations on a queue.



Methods	Description
`push()`	Inserts an element at the back of the queue.
`pop()`	Removes an element from the front of the queue.
`front()`	Returns the first element of the queue.
`back()`	Returns the last element of the queue.
`size()`	Returns the number of elements in the queue.
`empty()`	Returns `true` if the queue is empty.

Key Functions of queue:

1. Constructor:
   - queue(): Constructs an empty queue.
2. Element Access:
   - front(): Returns a reference to the first element in the queue.
   - back(): Returns a reference to the last element in the queue.
3. Capacity:
   - empty(): Returns true if the queue is empty, otherwise returns false.
   - size(): Returns the number of elements in the queue.
4. Modifiers:
   - push(const value_type& val): Adds an element to the end of the queue.
   - emplace(Args&&... args): Constructs and adds an element to the end of the queue in place.
   - pop(): Removes the first element in the queue.
   - swap(queue& other): Exchanges the contents of the queue with those of another queue.

Examples:

Here's a simple example demonstrating the usage of a queue in C++:

```
#include <iostream>
#include <queue>

int main() {
     queue<int> q;
```

```
    // Add elements to the queue
    q.push(10);
    q.push(20);
    q.push(30);

    // Display the front and back elements
     cout << "Front element: " << q.front() <<  endl;
     cout << "Back element: " << q.back() <<  endl;

    // Remove elements from the queue
    q.pop();
      cout << "Front element after pop: " << q.front() <<
endl;

    // Check if the queue is empty
    if (q.empty()) {
        cout << "The queue is empty." <<  endl;
    } else {
        cout << "The queue is not empty." <<  endl;
    }

   // Display the size of the queue
     cout << "Queue size: " << q.size() <<  endl;

    return 0;
}
```

Adding Elements:

- q.push(10): Adds the element 10 to the end of the queue.
- q.push(20): Adds the element 20 to the end of the queue.
- q.push(30): Adds the element 30 to the end of the queue.

Accessing Elements:

- q.front(): Returns the first element, which is 10.
- q.back(): Returns the last element, which is 30.

Removing Elements:

- q.pop(): Removes the first element (10) from the queue.

Checking if Empty:

- q.empty(): Returns false if the queue is not empty.

Getting Size:

- q.size(): Returns the number of elements in the queue, which is 2 after the pop() operation.