Deque

`#include <deque>

std::deque<DataType> dequeName;`

Decoding Deque:

- Doubly Ended Queue: Insertion and deletion from both front and back.
- Dynamic & Distributed: No contiguous memory, uses fixed arrays.
- Deque's Dynamism: Adapts in size as needed.
- Random Access Reality: Yes, random access is possible.
- Deque's Toolbox: Methods include push_front, push_back, pop_front, pop_back, size, empty, and more.

1. push_back(value): Adds an element to the end of the deque.
2. push_front(value): Adds an element to the beginning of the deque.
3. pop_back(): Removes the last element from the deque.
4. pop_front(): Removes the first element from the deque.
5. emplace_back(): Constructs and adds an element to the end in-place.
6. emplace_front(): Constructs and adds an element to the beginning in-place.
7. insert(position, value): Inserts elements at the specified position.
8. erase(position): Removes the element at the specified position.
9. clear(): Removes all elements from the deque.

10. size(): Returns the number of elements in the deque.

11. max_size(): Returns the maximum possible number of elements the deque can hold.

12. resize(new_size[, value]): Changes the size of the deque. Optionally, a value can be provided to initialize new elements.

13. empty(): Checks if the deque is empty (i.e., if its size is zero).
14. at(index): Accesses the element at the specified index, performing bounds checking.

15. operator[] (index): Accesses the element at the specified index. No bounds checking is performed.

16. front(): Returns a reference to the first element in the deque.

17. back(): Returns a reference to the last element in the deque.

18. begin(): Returns an iterator to the beginning of the deque.

19. end(): Returns an iterator to the end of the deque.

20. rbegin(): Returns a reverse iterator to the reverse beginning of the deque.

21. rend(): Returns a reverse iterator to the reverse end of the deque.

22. swap(other_deque): Swaps the contents of the deque with another deque of the same type and size.

23. shrink_to_fit(): Attempts to reduce the deque's capacity to fit its size.

24. data(): Returns a pointer to the underlying array, allowing direct memory manipulation.