C++ Lists Cheat Sheet

1. Declaring a List

```
#include <list>
std::list<int> l1;              // Empty list of integers
std::list<int> l2(5);           // List of size 5 with default values (0 for int)
std::list<int> l3(5, 10);       // List of size 5, all elements initialized to 10
std::list<int> l4 = {1, 2, 3, 4}; // List initialized with elements
```

2. Basic Functions

```
l.size();       // Returns the number of elements
l.empty();      // Returns true if the list is empty
l.clear();      // Removes all elements from the list
l.push_back(x);  // Adds an element 'x' at the end
l.push_front(x); // Adds an element 'x' at the beginning
l.pop_back();    // Removes the last element
l.pop_front();   // Removes the first element
```

3. Accessing Elements

```
l.front();      // Returns the first element
l.back();       // Returns the last element
```

4. Iterators

```
l.begin();      // Iterator to the beginning
l.end();        // Iterator to the end (one past the last element)
```

```
l.rbegin();      // Reverse iterator to the beginning (last element)
l.rend();        // Reverse iterator to the end (before first element)
```

5. Modifying Elements

```
l.insert(it, x);              // Inserts 'x' at iterator position 'it'
l.insert(it, n, x);           // Inserts 'n' copies of 'x' at position 'it'
l.erase(it);                  // Removes the element at iterator position 'it'
l.erase(startIt, endIt);      // Removes elements in the range [startIt, endIt)
l.resize(n);                  // Resizes the list to contain 'n' elements
l.resize(n, x);               // Resizes and fills new elements with 'x'
l.swap(l2);                   // Swaps elements with another list l2
std::swap(l1, l2);            // Alternative to swap two lists
```

6. Capacity Functions

```
l.max_size();    // Returns the maximum number of elements
```

7. Sorting and Searching

```
l.sort();                     // Sorts the list in ascending order
l.reverse();                  // Reverses the list
auto it = std::find(l.begin(), l.end(), x); // Find an element (returns iterator)
bool exists = (std::find(l.begin(), l.end(), x) != l.end()); // Check existence
```

8. Common Operations

```
std::list<int> l = {1, 2, 3, 4, 5};
int sum = std::accumulate(l.begin(), l.end(), 0);   // Sum of all elements
```

```
int min = *std::min_element(l.begin(), l.end());    // Minimum element
int max = *std::max_element(l.begin(), l.end());    // Maximum element
int count = std::count(l.begin(), l.end(), x);      // Count occurrences
```

9. Looping through a List

```
// Using a range-based for loop
for (int x : l) { std::cout << x << " "; }
// Using iterators
for (auto it = l.begin(); it != l.end(); ++it) { std::cout << *it << " "; }
```