0_cpp_stl.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;

// Pairs
    pair<int , int> p = {1, 2};
    cout<<p.first<<" "<<p.second;

    pair<int, pair<int, int>> p = {1, {2, 3}};
    cout<<p.first<<" "<<p.second.first<<" "<<p.second.second;

    pair<int int> p[] = {{1, 2}, {3, 4}, {5, 6}};
    cout<<p[1].second;

// Vectors
    vector<int> v;

    v.push_back(1);
    v.emplace_back(2);

    vector<pair<int, int>> v;
    v.push_back({1, 2});
    v.emplace_back(1, 2);

    v.push_back({1, 2});
    v.pop_back();

    vector<int> v(5);
    vector<int> v(5, 0);
    vector<int> v(5, -1);

    vector<int> v(v1);


    vector<int>::iterator it = v.begin();

    it++
    cout<<*(it);

    it = it + 2;
    cout<<*(it);

    vector<int>::iterator it = v.begin();
    vector<int>::iterator it = v.end();
    vector<int>::iterator it = v.rend();
    vector<int>::iterator it = v.rbegin();

    cout<<v[0];
    cout<<v.at(0);
    cout<<v.front();
    cout<<v.back();
    cout<<v.size();

    for(vector<int>::iterator it = v.begin(); it != v.end(); it++){
        cout<<*(it)<<" ";
    }

    for (auto it=v.begin(); it!=v.end(); it++) {
        cout<<*(it)<<" ";
    }

    for(auto it : v){
        cout<<it<<" ";
    }

    v.erase(v.begin() + 1);
    v.erase(v.begin() + 1, v.begin() + 3); // [start, end)

    vector<int> v(2, 100);
    v.insert(v.begin(), 300)
    v.insert(v.begin() + 1, 2, 200)
    v.insert(v.begin(), v2.begin(), v2.end());

    v1.swap(v2);
    v.clear();
    cout<<v.empty();


// List (doubly linked list)
    list<int> ls;

    ls.push_back(1);
    ls.emplace_back(2);

    ls.push_front(1);
    ls.emplace_front(2);

    ls.pop_front(1);
    ls.pop_back(2);

// Deque
    deque<int> dq;
    dq.push_back(1);
    dq.emplace_back(2);
    dq.push_front(1);
    dq.emplace_front(2);

    dq.pop_back();
    dq.pop_front();

    dq.front();
    dq.back();

// Stack
    stack<int> st;
    st.push(1);
    st.emplace(2);
    st.pop();

    st.top();

    st.size();
    st.empty();

// Queue
    queue<int> q;
    q.push(1);
    q.emplace(2);
    q.pop();

    q.front();
    q.back();

    q.front() += 4;

    q.size();
    q.empty();


// Priority Queue
    // Max Heap
    priority_queue<int> pq;
    pq.push(1);
    pq.emplace(2);

    pq.top();

    pq.pop();

    // Min Heap
    priority_queue<int, vector<int>, greater<int>> pq;

// Set
    set<int> s;
    s.insert(1);
    s.emplace(2);

    auto it = s.find(1);
    s. erase(it);
    s.erase(2);

    auto it = s.lower_bound(1);
    auto it = s.upper_bound(1);


// Multiset
    multiset<int> ms;
    ms.insert(1);
    ms.insert(1);
    ms.insert(1);

    ms.erase(1)

    int cnt = ms.count(1);

    ms.erase(ms.find(1))

    ms.erase(ms.find(1), ms.find(1)+2);

// unordered set
    unordered_set<int> us;
    us.insert(1);
    us.insert(1);
    us.insert(2);

// Map
    map <int, int> m;
    map <int, pair<int, int>> m;
    map <pair<int, int>, int> m;

    m[1] = 3;
    map.emplace({1 , 3})
    map.insert({1, 3});

    for(auto it: m){
        cout<<it.first<<" "<<it.second<<endl;
    }

    cout<<m[1];

    auto it = m.find(1);
    cout<<*(it).second;

// Multimap
// Sort
    sort(a, a+n);
    sort(a.begin(), a.end());

    sort(a+2, a+4);
    sort(a, a+n, greater<int>);

    pair<int, int> a[] = {{1, 2}, {3, 4}, {5, 6}};
    sort(a, a+n, comp);

    bool comp(pair<int, int> &a, pair<int, int> &b){
        if(a.first < b.first){
            return true;
        }else if(a.first == b.first){
            return a.second > b.second;
        }
        return false;
    }

// Basics
```

```
    int num  = 7;
    int cnt  = __builtin_popcount(num); // no of set bits

    long long num = 1234567843232;
    long long cnt = __builtin_popcountll(num);

    // Permutations
    string s = "abc";
    string s = "123";
    string s = "321";
    sort(s.begin(), s.end());

    do {
        cout<<s<<endl;
    } while(next_permutation(s.begin(), s.end()));

    int maxele = *max_element(a, a+n);
    int minele = *min_element(a, a+n);


```