### STL

vector

* push\_back
* pop\_back
* size
* clear
* insert vec.insert(it, x);(O(N))
* erase vec.erase(vec.begin() + 3);(O(N)) vec.erase(vec.begin(), vec.end());(O(N))
* reserve
* resize

string

* size
* length
* insert str.insert(pos, str1); str.insert(it, it*begin, it*end); str1.insert(str1.begin()+3, str2.begin()+1, str2.end())
* erase str.erase(it); str.erase(first, last); str.erase(pos, len);
* clear str.clear();(O(1))
* substr str.substr(pos, len);(O(len))
* string::npos 用以作为find函数失配时的返回值
* find str.find(str2);(O(nm)) str.find(str2, str.find(str2) + 1);(O(nm))
* replace str.replace(pos, len, str2);(O(str.length())) str.replace(it*begin, it*end, str2);

set  
unordered\_set

* insert st.insert(x);(O(logN))
* find st.find(x);(O(logN)) 没找到返回st.end()
* erase st.erase(it);(O(1)) st.erase(x);(O(logN)) st.erase(first, last);(O(last-first))
* size
* clear (O(N))

map  
unordered\_map

* find mp.find(key);(O(logN)) 没找到返回mp.end()
* erase mp.erase(it);(O(1)) mp.erase(key);(O(logN)) mp.erase(first, last);(O(last-first))
* size
* clear (O(N))

queue

* push
* front
* back
* pop
* empty
* size

stack

* push
* top
* pop
* empty
* size

pair

algorithm

* max
* min
* abs 只能对整数取绝对值
* swap
* reverse reverse(it, it+1)
* next*permutation next*permutation(arr, arr+n); next\_permutation(it, it+n);
* fill fill(arr, arr+n, 0);
* sort
* lower*bound [begin, end)范围内第一个小于等于val的元素的位置 lower*bound(arr, arr+n, val);返回指针 lower\_bound(it, it+n, val);返回迭代器 没有找到返回可以插入的位置(O(log(begin-end)))
* upper*bound [begin, end)范围内第一个大于val的元素的位置 lower*bound(arr, arr+n, val);返回指针 lower\_bound(it, it+n, val);返回迭代器 没有找到返回可以插入的位置(O(log(begin-end)))