|  |  |
| --- | --- |
| #include <algorithm>  find(Start\_location, End\_location, key)  auto id = find(arr,arr+n,key);  bool present = binary\_search(arr,arr+n,key);  //lower bound first occurrence of element >= the key  //upper bound first occurrence of element > the key | bool compare(type a, type b)  sort(arr,arr+n,compare); type can be string/integer/char/double  auto lb= lower\_bound(arr,arr+n,key);  auto ub= upper\_bound(arr,arr+n,key);  index=lb-arr; |
| s.empty() bool  s.append("str") || s+="str"  s.erase(st\_index,length to remove)  for(int i=0;i<s.length();i++)   cout<<s[i]<<":"; for(char c:s0) cout<<c;  #include<cstring>  char arr[s.length()+1];  strcpy(arr,s.c\_str());  for(auto it=strtok(arr," ");it!=NULL;it=strtok(NULL," "))   cout<<it; | #include<string> string s0;  string s1("Hello World!");  string s2="hello World";  string s3(s2);  string s4=s3;  char a[]={'a','b','c','d','\\n'}; string s5(a);  s.clear() || s.length()  s1.compare(s2) || s1<s2  s[0] || s.find("apple") |
| s.remove("apple"); //remove all instances of apple  auto it = s.begin(); it++; it++;  s.erase(it);// remove 3rd element  auto it = s.begin(); it++; it++;  s.insert(it,"the new string");   for(string ss:s)  cout<<ss<<endl; | #include <list>  list<string> s{"apple","mango","banana"};  push\_back push\_front  pop\_back pop\_front  insert erase remove reverse sort  list <int> :: iterator it;  auto it = g.begin;  for(it = g.begin(); it != g.end(); ++it)   cout << \*it;  auto j=v.end();// travers from back to rfont  j--;  for(;\*j>=i;j--) |
| stack<int> st;  st.push(10);  st.pop();   while (!st.empty())  st.top();  queue<int> st;  st.push(10);  st.pop();   while (!st.empty())  st.front();  priority\_queue<int> pqx; // max heap  priority\_queue<int,vector<int>,greater<int>> pq;  pq.pop(); pq.top() pq.push(no); | priority\_queue<Person,vector<Person>,comparatorPerson> Q;  class comparatorPerson {  public :  bool operator()(Person p1, Person p2)  { return p1.height < p2.height; } }; //ascending order  map<int, int> m; m[key]=value  // check if key exists // m.find(key)!=m.end();  map<int, int>::iterator itr;  for (itr = gquiz1.begin(); itr != gquiz1.end(); ++itr)  cout << itr->first<< itr->second << '\n';  num = gquiz2.erase(4); //remove 4 from the map  unordered\_map<string, int> umap;  // rest is same import map |