附录：STL 函数解析

一、vector

function	definition	example

operator =	assign content	vector <int> a;
		a.push_back(1);
		a.push_back(2);
		vector <int> b=a;
		//element in vector b:1 2

begin	return iterator to beginning	Show elements in a vector: vector <int> a; vector <int> ::iterator it; for(it=a.begin();it!=a.end();it++) cout<<(*it)<<endl; Note:*(a.end()) is wrong Note:data store:[a.begin(),a.end())
end	return iterator to end	
size	return size	Show elements in a vector: vector <int> a; for(int i=0;i<a.size();i++) cout<<a[i]<<endl;

reserve	request a change in capacity	vector <int> a;
		a.push_back(1);
		a.push_back(2);
		reverse(a.begin(),a.end());
		//element in vector a:2 1

front	access first element	a.front() similar to: *(it.begin()), a[0]
back	access last element	a.back() similar to: *(--it.end()), a[a.size()-1]

push_back	add element at the end	vector <int> a;
		a.push_back(1);
		//element in vector a:1

pop_back	delete last element	vector <int> a;
		a.push_back(1);
		a.push_back(2);
		a.pop_back();
		//element in vector a:1

clear	clear content	//a good coding style
		vector <int> a;
		cin>>t;
		while (t--)
		{
		a.clear();
		...
		}

二、set

function	definition	example
operator =	Copy container content	Similar to vector operator =
begin	return iterator to beginning	Show elements in a vector: set <int> a; set <int> ::iterator it; for(it=a.begin();it!=a.end();it++) cout<<(*it)<<endl; Note:elements show in ascending sequence, same elements will appear only once.(排序去重)
end	return iterator to end	

clear	clear content	//a good coding style
		set <int> a;
		cin>>t;
		while (t--)
		{
		a.clear();
		...
		}

insert	insert element	set <int> a;
		a.insert(2);
		a.insert(2);
		a.insert(1);
		elements in set a:1 2
		Note:elements shore in ascending sequence.

count	count elements with a specific value	usually used to check whether the elements always exist in the set if element e does not exist in set a, then a.count(e)==0, otherwise, a.count(e)==1
size	return container size	usually used to count the numbers of different elements in the set

三、multiset

Similar to set:

①element appear in ascending sequence

②using almost the same function prototype(函数原型)

Different to set:

①same element can appear more than once in multiset,based on the times the element is inserted

②in multiset return value of a.count(e) can be more than 1

四、string

function	definition	example
operator =	string assignment	①string s="hello world"; ②char t[]="hello world" string s=t;

operator +=	append to string	string s="hello ";
		string t="world";
		string u=s+t; //u="hello world"
		s+=t; //s="hello world"

operator <,>,<=,>=, !=,==	compare two string constants	//compare based on dictionary order "ab"<"b" "ab"<"abc" "a"=="a"

begin	return iterator to beginning	Reverse a string constant:
		string s,revs;
		revs=s;
		reverse(revs.begin(),revs.end());

end	return iterator to end	
length size	return length of string	Show elements in string: string s; for (int i=0;i<s.length();i++) cout<<s[i]<<endl;

c_str	get C string equivalent	usually used to used some functions in language C
		//change from string to long long
		string s;
		long long val=atoll(s.c_str());
		//function: atoll(char *s) are used to change from char * to long long

find	find content in string	//a.find(b):
		If string b is a sub-string of string a, return the first occurrence of the content in string.
		If string b is not a sub-string of string a,return string::npos.
		string a="hello world hello world";
		a.find("world"); //6
		a.find("hi"); //string::npos

rfind	find last occurrence of content in string	string a="hello world hello world";
		a.rfind("world"); //18
		a.rfind("hi"); //string::npos

substr	generate substring	//function prototype:
		a.substr(size_t pos, size_t len);
		string a="hello world";
		string b=a.substr(6,5);
		//b="world"

五、string 读写

getline	get line from stream into string	Note:does not read '\n'
operator >>	Extract string from stream	string s; cin>>s;
operator <<	Insert string into stream	string s: cout<<s;

六、stringstream

①int_to_string:change from int to string

```
#include <sstream>
#include <string>
string int_to_string(int n)
{
    stringstream ss;
    ss<<n;
    return ss.str();
}
```

②deal_string:cut off a long string to several small strings

```
#include <sstream>
#include <string>
#include <vector>
vector <string> words;
void deal_string(string s)
{
    words.clear();
    stringstream ss(s);
    while (ss>>t)
    {
        words.push_back(t);
    }
}

//s="this is a simple test"
//element in vector words:"this","is","a","simple","test"
```

附录：STL 示例

一、单词表：

描述：提取一行英文文本中的单词，重复出现的单词只取一个，把它们按照字典顺序排序，建立为一个单词表，并输出。

```
1.  int main()
2.  {
3.      string s,t;
4.      int q,cas=0;
5.      cin>>q;
6.      getchar();
7.      set <string> word;
8.      while (q--)
9.      {
10.         word.clear();
11.         getline(cin,s);
12.         for (int i=0;i<s.length();i++)
13.             if (!isalpha(s[i])) s[i]=' ';
14.         stringstream ss(s);
15.         while (ss>>t)
16.         {
17.             if (!word.count(t))
18.                 word.insert(t);
19.         }
20.         printf("case #%d:\n",cas++);
21.        for (set<string>::iterator iter=word.begin();
                iter!=word.end();iter++)
22.             cout<<(*iter)<<' ';
23.         cout<<endl;
24.     }
25.     return 0;
26. }
```

二、查找单词

描述：有一个单词 W，输出它在字符串 S 中从左到右第一次出现的位置 IDX(设 S 中的第 1 个字符的位置为 1)。W 只由英文字母组成，S 除英文字母和汉字之外在任何位置（包括头和尾）另有一个或多个连续的空格。查找单词时，不区分大小写，但要求完全匹配，即单词 W 必须与 S 中的某一独立单词在不区分大小写的情况下完全匹配。W 仅是 S 中某一单词的一部分就不算匹配。

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int t,cas=0;
    cin>>t;
    getchar();
    while (t--)
    {
        string key,str;
        getline(cin,key);
        getline(cin,str);
        for (int i=0;i<key.length();i++)
            key[i]=tolower(key[i]);
        for (int i=0;i<str.length();i++)
            str[i]=tolower(str[i]);
        key=" "+key+" ";
        str=" "+str+" ";
        //note: important!!!
        printf("case #%d:\n",cas++);
        if (str.find(key)!=string::npos)
            cout<<str.find(key)+1<<endl;
        else cout<<"None"<<endl;
    }
    return 0;
}
```

三、子串间距

描述：计算字符串 s1 和字符串 s2 在字符串 s 中的最大间距。间距指一个字符串的尾字符与另一个字符串首字符之间的字符个数，显然间距是一个非负整数。当 s1 或 s2 没有在 s 中出现时，间距为 0。例如 :s1 为 ab, s2 为 bd, s 为 bdabbdcdabdababcccbd, 则最大间距为 14。

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int t,cas=0;
    string s,s1,s2;
    cin>>t;
    getchar();
    while (t--)
    {
        getline(cin,s1);
        getline(cin,s2);
        getline(cin,s);
        printf("case #%d:\n",cas++);
        if (s.find(s1)==string::npos || s.find(s2)==string::npos)
        {
            cout<<0<<endl;
            continue;
        }
        int p=-s.find(s1)+s.rfind(s2)-s1.length();
        int q=-s.find(s2)+s.rfind(s1)-s2.length();
        cout<<max(max(p,q),0)<<endl;
    }
    return 0;
}
```