

1. max () 、min () 和abs ()

- max (x, y) 和min (x, y) 分别返回x和y中的最大值和最小值，且参数必须为两个，如果要三个数比较，要用max (x, max (y, z))
- abs (x) 返回x的绝对值。而且x必须为整数。

2.reverse () 反转函数

reverse(it1,it2)可以将数组指针在[it1,it2)之间的元素或容器的迭代器在该范围内的元素进行反转。

例：int a[10] = {1,2,3,4,5,6,7,8,9,10};

reverse(a,a+4); //将a[0]-a[3]进行反转

//如果使用vector，则reverse(a.begin(),a.end());

或者是下面的：

```
auto vit = v.begin();
reverse(vit, vit + 6);
for (; vit != v.end(); vit++)
    cout << *vit << " ";
cout << endl;
```

3.next_permutation(a,a+n)

a可以是数组也可以vector，然后n就是从a的第一位到a的第n位之间的元素进行全排序。

基本操作如下：

```
do
{
    for (int i = 0; i < v.size(); i++)
        cout << v.at(i) << " ";
    cout << endl;
} while (next_permutation(v.begin(), v.end()));
```

一下是对1, 2, 3进行全排序的结果：

```
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
```

4.sort (a, a+n) 一般为升序排序

a可以是数组，也可以是vector等类型。

(a, a+n) 就是对a中的第一个元素到第n个元素进行排序。

5. lower_bound()和upper_bound()

lower_bound(first, last, val)用来寻找在数组或容器的[first,last)范围内第一个值**大于等于val**的元素的位置，如果是**数组**，则**返回该位置的指针**；如果是**容器**，则**返回该位置的迭代器**。

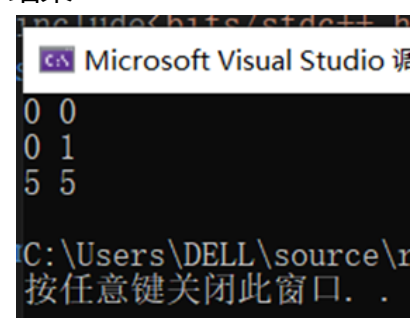
upper_bound(first, last, val)用来寻找在数组或容器的[first,last)范围内第一个值**大于val**的元素的位置，如果是**数组**，则**返回该位置的指针**；如果是**容器**，则**返回该位置的迭代器**。

显然，如果数组或容器中没有需要寻找的元素，则lower_bound()和upper_bound()均返回可以插入该元素的位置的指针或迭代器(即假设存在该元素时，该元素应当在的位置)。lower_bound()和upper_bound()的复杂度均为 $O(\log(\text{last} - \text{first}))$ 。

下面是例子：

```
int a[5] = { 1, 2, 3, 5, 6 };
//寻找-1
int* lowerPos = lower_bound(a, a + 5, -1);
int* upperPos = upper_bound(a, a + 5, -1);
cout << lowerPos - a << " " << upperPos - a << endl;
//寻找1
lowerPos = lower_bound(a, a + 5, 1);
upperPos = upper_bound(a, a + 5, 1);
cout << lowerPos - a << " " << upperPos - a << endl;
//寻找10
lowerPos = lower_bound(a, a + 5, 10);
upperPos = upper_bound(a, a + 5, 10);
cout << lowerPos - a << " " << upperPos - a << endl;
```

结果：



```
Microsoft Visual Studio 运行
0 0
0 1
5 5
C:\Users\DELL\source\run
按任意键关闭此窗口. .
```

6. pow()函数：求一个数的次方

1 **pow(x,y)**； 返回x的y次方