priority_queue默认创建大堆

```
void main()
{
	int ar[] = {3, 2, 7, 6, 0, 4, 1, 9, 8, 5};
	int n = sizeof(ar) / sizeof(int);
	priority_queue<int> q;
	priority_queue<int> q(ar, ar+n); //默认创建大堆


	cout<<"size = "<<q.size()<<endl;
	cout<<"top = "<<q.top()<<endl;
	q.pop();
	cout<<"top = "<<q.top()<<endl;

}
```

创建小堆

```
void main()
{
	int ar[] = {3, 2, 7, 6, 0, 4, 1, 9, 8, 5};
	int n = sizeof(ar) / sizeof(int);
 //创建小堆
	priority_queue<int,vector<int>, greater<int> > q(ar, ar+n);

	cout<<"size = "<<q.size()<<endl;
	cout<<"top = "<<q.top()<<endl;
	q.pop();
	cout<<"top = "<<q.top()<<endl;

}
```

优先级队列的好处是可以拿到排序的值（每拿到堆顶元素就出一次堆顶）

```
void main()
{
	int ar[] = {3, 2, 7, 6, 0, 4, 1, 9, 8, 5};
	int n = sizeof(ar) / sizeof(int);
	//priority_queue<int> q;
	//priority_queue<int> q(ar, ar+n); //大堆
```

```
	priority_queue<int,vector<int>, greater<int> > q(ar, ar+n);

	cout<<"size = "<<q.size()<<endl;
	cout<<"top = "<<q.top()<<endl;
	q.pop();
	cout<<"top = "<<q.top()<<endl;

}
```

联想到也可以用链表调用sort方法排序，也同样可以能得到排序好的值

```
void main()
{
	int ar[] = {3, 2, 7, 6, 0, 4, 1, 9, 8, 5};
	int n = sizeof(ar) / sizeof(int);
	list<int> mylist(ar, ar+n);

	mylist.sort();

	for(auto e : mylist)
		cout<<e<<" ";
	cout<<endl;
}
```

仿函数

```
template<class _Ty = void>
struct Plus
{
	_Ty operator()(const _Ty& _Left, const _Ty& _Right) const
	{
		return (_Left + _Right);
	}
};

void main()
{
	plus<int> pl;
	cout<<pl(1,2)<<endl;
	cout<<pl.operator()(1,2)<<endl;
}
```

比较两个日期大小

```
class Date
{
public:
	Date(int year, int month, int day)
		: m_year(year), m_month(month), m_day(day)
	{}
	~Date()
	{}
public:
	bool operator<(const Date &d)const
	{
		if(m_year < d.m_year)
			return true;
		else if(m_year > d.m_year)
			return false;

		if(m_month < d.m_month)
			return true;
		else if(m_month > d.m_month)
			return false;

		if(m_day < d.m_day)
			return true;
		else if(m_day > d.m_day)
			return false;
		return false;
	}
private:
	int m_year;
	int m_month;
	int m_day;
};

void main()
{
	priority_queue<Date> q;
	q.push(Date(2015,11,13));
	q.push(Date(2019,3,5));
}
```