排序：

要创建.cpp 文件而不是.c 文件，sort 函数在 C++标准模板库（STL）中，要包含头文件#include <algorithm>，同时还要打开命名空间 using namespace std;。即以下代码：

```
#include <algorithm>
using namespace std;
```

sort 的时间复杂度为 O(nlongn)，是一种混合排序，主要为快速排序，同时还结合了插入排序和堆排序，sort 函数会自动根据待排序列的情况选择合适的排序方式。当数据量较大时 sort 采用的是快速排序，此处加上快速排序的思想，当在快排递归到某一区间时，如果数据量小于某个门槛的话，为减小递归层数，会采用插入排序，此处加上插入排序的思想，当数据量过大或者待排序列有序性高时，都会使递归层数过深，带来大量的内存消耗，有出现最坏情况的倾向，这时就会采用堆排序来优化，此处加上堆排序的思想。

sort 的具体使用方法，类似于这样：sort(begin,end,cmp)，cmp 参数可以省略。其中 begin 为待排序列的第一个元素的地址，end 为待排序列的最后一个元素的下一个元素的地址，比如对 number[1]到 number[5]排序，可以这样写：sort(number+1,number+6)，sort 的排序区间为左闭右开，即包括 begin 这个元素但是不包括 end 这个元素。再来讲下 cmp 参数，cmp 参数可以不加，不加默认为从小到大排序，如果有其它想自定义的排序方式，可以加上 cmp 参数，参数名随便起，不一定为 cmp。比如你想从大到小排序，可以这样写：

```
#include <stdio.h>
#include <algorithm>
```

```
using namespace std;
#define N 1005
int number[N];
bool cmp(int x, int y)
{
    return x > y;
}
int main()
{
    int n;
    scanf("%d", &n);
    for(int i = 1; i <= n; i++)
        scanf("%d", &number[i]);
    sort(number + 1, number + 1 + n, cmp);
    for(int i = 1; i <= n; i++)
        printf("%d ", number[i]);
    return 0;
}
```

在这里引出 cmp 函数的写法，cmp 函数的返回值为 bool 型，bool 型为 C++专有，C 没有，取值只有 true 和 false 两种，任何非 0 值都为 true，0 为 false，两者等价，所以返回类型写 int 也可。参数有两个，要比较的两个变量是什么类型就写什么类型（包括结构体），变量名称随意。cmp 函数的执行过程是，cmp 函数的判断结果为真，返回 true，代表要比较的两个值符合判断条件，不需要交换位置；cmp 函数的判断结果为假，返回 false，代表要比较的两个值不符合判断条件，需要交换位置。

结构体 cmp 函数的编写方式自学，现在暂时用不到但是不久就会用到，这点自学能力应该有吧，我们去年就没讲，都是自己学的。

二分：

二分就是二分查找，也称折半查找，查找的时间复杂度为O(longn)。注意，用二分查找的前提是**数组已有序**，升序排序降序排序均可，但是必须要有序（用到了前面学习的 sort）。此处加上二分的思想。

二分查找的代码如下：

```
#include <stdio.h>
int bin_search(int number[], int l, int r, int key)
{
    while(l <= r)
    {
        int m = l + r >> 1;
        if(number[m] == key)
            return m;
        else if(number[m] < key)
            l = m + 1;
        else
            r = m - 1;
    }
    return 0; //代表未查找到目标元素
}
int main()
{
    int number[] = {0, 1, 3, 5, 7, 9, 10, 11, 13, 15, 17, 19};
    int pos = bin_search(number, 1, 11, 10);
    printf("%d\n", number[pos]);
    return 0;
}
```