高级代码学习

2021/4/9：自定义sort算法

vector二维数组根据某列排序

原创 zzyzgg 2019-09-17 21:32:19 1027 收藏 2
分类专栏： c++

写一个bool类型的comp函数，比如下面根据第二个元素排序：

```

bool cmp1(const vector<int> &a, const vector<int> &b){
    return a[1] > b[1];
}


sort(allvec.begin(), allvec.end(), cmp1)

```

1. 设置一个bool类型的函数：该函数用来确定两个数的比较方式
2. 当为一维数组时，a<b表示升序排列；a>b表示降序排列；
3. 当为二维数组时，可以根据实际的需要确定时根据第几列的数据进行排序；以及综合几列数据的情况进行排序；
4. 如下举例：

```
static bool cmp1(const vector<int> &a, const vector<int> &b){
        if(a[1]==b[1])return a[0]<b[0];//第二列相同的情况下，按照第一列的数据从小到大进行排序
        return a[1] < b[1];//按照第二列从小到大排序
    }
    static bool cmp2(const vector<int> &a, const vector<int> &b){
        return a[0] < b[0];//按照第一列从小到大排序
    }
```

```
//字符串处理大整数的加法
```

```
string add(string& a,string& b){
        int n1=a.size()-1;
        int n2=b.size()-1;
        int carry=0;
        string ans;
        while(n1>=0||n2>=0||carry>0){
            int t1=n1>=0?a[n1--]-'0':0;
            int t2=n2>=0?b[n2--]-'0':0;
            ans+=(t1+t2+carry)%10+'0';
            carry=(t1+t2+carry)>=10?1:0;
        }
        reverse(ans.begin(),ans.end());
        return ans;
    }
};
```

作者：Over-Lord
链接：https://leetcode-cn.com/problems/additive-number/solution/xia-biao-zuo-wei-fen-duan-dian-dfs-by-over-lord/
来源：力扣（LeetCode）

优先队列的自定义sort算法

```
struct cmp2{
        bool operator()(vector<int>&a,vector<int>&b)
        {
            if(a[0]==b[0])return a[1]>b[1];//小的优先级高，所以是从小到大排序
            else return a[0]>b[0];
        }
    };

    struct cmp3{
        bool operator()(vector<int>&a,vector<int>&b)
        {
            if(a[1]==b[1])return a[2]>b[2];//小的优先级高，所以是从小到大排序
            else return a[1]>b[1];
        }
    };
```

LONG_MAX， LONG_MIN 分别表示长整型的最大值和最小值

8个方向：数组+加法运算表示方向——使代码更加简洁

```
int dir_x[8] = {0, 1, 0, -1, 1, 1, -1, -1};
int dir_y[8] = {1, 0, -1, 0, 1, -1, 1, -1};

 for (int i = 0; i < 8; ++i) {
            int tx = x + dir_x[i];
            int ty = y + dir_y[i];
            if (tx < 0 || tx >= board.size() || ty < 0 || ty >=
board[0].size()) {
                continue;
            }
            // 不用判断 M，因为如果有 M 的话游戏已经结束了
            cnt += board[tx][ty] == 'M';
        }
```

作者：LeetCode-Solution
链接：https://leetcode-cn.com/problems/minesweeper/solution/sao-lei-you-xi-by-leetcode-solution/
来源：力扣（LeetCode）

lambda表达式的使用对运行速度的影响：

```
 sort(pairs.begin(),pairs.end(),[](vector<int>&a,vector<int>&b){return
a[1]<b[1];});//运行快
 sort(pairs.begin(),pairs.end(),[&](vector<int>a,vector<int>b){return
a[1]<b[1];});//运行慢
```