# 实验一：线性表的物理实现

计科1903陈旭 201914020128

**日期**：**2020.10.13**

**当天任务**：**实现一个顺序表。**

**产生问题**：为什么在基类中需要用纯虚函数来声明

**解决办法**：多重继承可能会在构造的时候，同一个基类被构造多次。

**产生问题**：void clear() 函数在实现时，起初考虑使用for循环遍历顺序表，使用delete删除

每一个元素，并将当前位置赋值0，线性表元素个数赋值为0。此方法时间复杂

度太高，并且可读性很差。

**解决办法**：直接用delete删除原有线性表，再重新调用构造函数建个新的线性表。

**日期：202010.15**

**当天任务**：**实现一个单向链表。**

**产生问题**：



纯虚函数E getValue() 未在子类中被实现。

具体代码：

     E& getvalue ()

        {

            if (curr==tail) return;

            return curr->next->element;

        }

基类中的声明：

virtual const E& getvalue() = 0; //返回当前元素值

**解决方式**：第一行加const (声明时必须加const)。

**产生问题**：实现之后，增加了一个函数用来方便调试：

输出链表中每一个元素的值。

        void fordisplay ()

        {

            assert(cnt > 0);

            link<E> \*ltemp;

            ltemp=curr;

            curr=head;

            while (curr!=NULL)

            {

                cout << curr->element << ' ';

                curr=curr->next;

            }

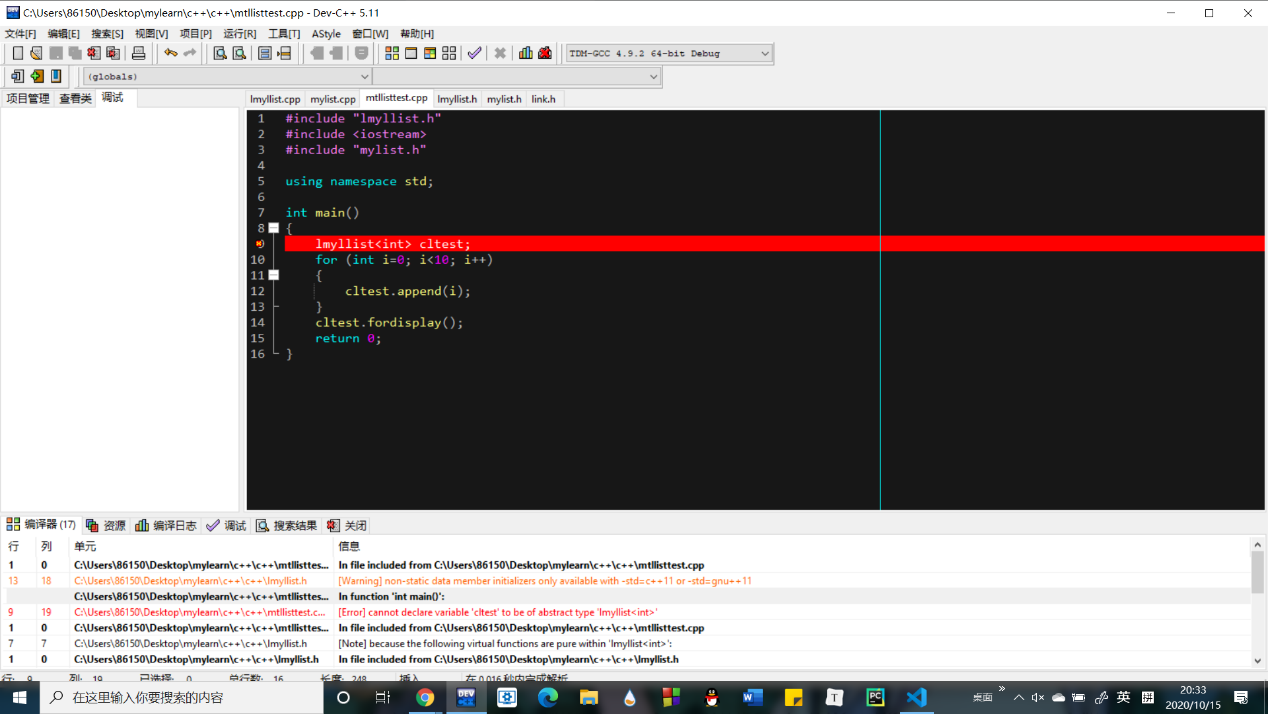
            cout << endl;

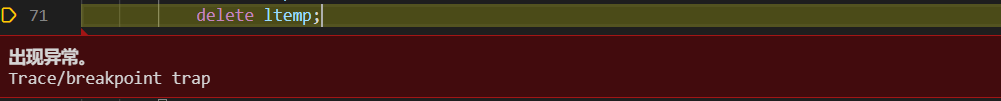
            curr=ltemp;

            delete ltemp;

        }

在调用中产生了很多不可名状的错误，如溢出，空指针报错以及下图几个错误。





代码如下：

       void append(const E& it)

        {

            tail=tail->next=new link<E>(it, NULL);

            cnt++;

        }

最终调试了很久，得出结论。

**解决方法**：

       void append(const E& it)

        {

            if (cnt==0)

            {

                head->element=it;

                cnt++;

            } //错误所在

            else

            {

                tail=tail->next=new link<E>(it, NULL);

                cnt++;

            }

        }

之前append、insert等函数均没有考虑cnt==0，curr,next,tail指向同一个位置且都是空指针的情况。加了if判断后恢复正常。

**产生问题：**在链表实现完成后，对照老师的代码，发现以下部分不相同：

 LList(int size=100) { init(); }

产生疑问，链表使用的是动态数组来存放数据的，为什么在构造函数里声明它的

最大尺寸，而且在之后的实现文件里未再见size被使用过。

**问题解答**：size作为一个默认的大小，如果存储数据量比较少，则使用size作为最大尺寸，

防止造成内存的浪费。