**实验三 C++中的动态内存分配（2学时）**

**一、实验目的**

1．掌握使用new运算符创建对象的方法；

2．掌握使用delete运算符释放对象内存的方法；

3．掌握使用面向对象思想实现链表类并对链表进行操作的方法。

**二、实验内容**

1．设计并编写链表类，能够实现链表的初始化（建立）、输出释放等功能。

**三、实验指导**

（1）链表由若干个结点【Node类对象】构成（每个结点有data域和next域，假设每个结点的data域是一个整数，并且没有重复元素），那么某个链表的逻辑示意图如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 |  |  | 1 |  |  | 2 |  |  | 6 |  |  | 5 | NULL |

图3-1 链表逻辑结构示意图

下面给出链表中结点类的定义：

class Node //结点类

{ public:

Node( ); //构造函数

void SetNext(Node \*p); //设置后继结点

int Getd(); //得到当前结点的数据值

Node \*GetNext( ); //得到后继结点的地址

~Node( ); //析构函数

private:

int data;

Node \*next;

};

Node::Node( )

{ cin>>data; next=NULL;}

void Node::SetNext(Node \*p)

{ next=p;}

int Node::Getd( )

{ return data;}

Node\* Node::GetNext( )

{ return next;}

Node::~Node( )

{ cout<<data<<" 被析构！"<<endl;}

**上面的Node类代码同学们可以直接使用，下面（2）、（3）为必做内容**

1. 利用Node类，设计并编写Link（链表）类，类中包含：

* 数据成员
* Head //头指针
* Num //链表中结点个数 【即链表的长度】
* 函数成员：
* Link( ) //初始化（建立）空链表
* Link(int n) //初始化（建立）一个具有n个结点的链表

【在建立链表的过程中按值从小到大排序】

* LinkPrint( ) //输出链表中所有结点data域的值
* GetHead( ) //返回链表的头指针
* ~Link( ) //释放链表中所有结点的内存空间

1. 编写main( )函数，测试Link类中所有函数

【选作内容：编写Link类的如下成员函数，根据实际完成情况有加分】

* Link\_Insert(int x) //在链表中插入值为x的结点
* Link\_Delete( int x) //删除链表中值为x的结点
* Link\_Connect(Link &p) //两个链表的连接

【即将p链表连接在当前链表的尾部】

* Link( Link &p) //使用深复制编写复制构造函数
* 自己设计Node类中data的类型，使本题实现一个具体的应用

**四、考核标准**

本实验总计10分。

1. Link类的框架 （2分）
2. Link类成员函数的实现（5分）
3. main（）函数编写的合理性及功能演示（3分）