# 实验一 线性表

## 1 实验目的

通过选择下面三个题目之一进行实现，掌握如下内容：

* 熟悉C++语言的基本编程方法，掌握集成编译环境的调试方法
* 学习指针、模板类、异常处理的使用
* 掌握链式结构线性表的操作的实现方法
* 学习使用线性表解决实际问题的能力

## 2 实验内容

### 2.1题目1——基础实验

根据线性表的抽象数据类型的定义，选择下面任一种链式结构实现线性表，并完成线性表的基本功能。

线性表存储结构（五选一）：

1、 带头结点的单链表

2、 不带头结点的单链表

3、 循环链表

4、 双链表

5、 静态链表

线性表的基本功能：

1. 构造函数：包括构造空链表、头插法构造链表、尾插法构造链表
2. 复制构造函数

3、 插入函数：可以在任意指定位置插入元素

4、 删除函数：可以删出任意指定位置的结点

5、 查找：可以输出任意指定位置的元素

6、 获取链表长度

7、 打印链表数据

8、 析构函数

编写测试main()函数测试线性表的正确性。

思考问题（选作）：

1、若输入为乱序的数组，编写一个新的构造函数，使得构造的链表按升序排列？

2、若有两个按升序排列的链表A和B，编写一个合并函数，完成A=AB的功能，使得合并后的链表没有重复数据。

3、编写一个链表逆置函数，若链表中的每一个结点为正序排列，则执行完该函数，链表中的每一个结点倒序排列。

### 2.2题目2——应用实验

利用线性表实现一个通讯录管理，通信录的数据格式如下：

struct DataType

{

int ID; //编号

char name[10]; //姓名

char ch; //性别

char phone[13]; //电话

char addr[31]; //地址

};

实现方式（三选一）：

1 可以在题目一实现的链表基础上实现该通讯录，通讯录就相当于链表类的一个实例。

2 可以使STL中的vector或list实现该通讯录，体会STL使用的好处。

3 可以单独实现一个通讯录类来实现。

功能要求：

* 实现通讯录的建立、增加、删除、修改、查询等功能
* 能够实现简单的菜单交互，即可以根据用户输入的命令，选择不同的操作。
* 能够保存每次更新的数据（选作）
* 能够进行通讯录分类，比如班级类、好友类、黑名单等等（选作）

编写测试main()函数测试通讯录的正确性

### 2.3题目3——应用实验

利用线性表实现一个一元多项式Polynomial类。

Polynomial的结点结构如下：

struct term

{

float coef; //系数

int expn; //指数

};

实现方式（三选一）：

1 可以在题目一实现的链表基础上实现该一元多项式，一元多项式就相当于链表类的一个派生类。

2 可以使STL中的vector或list实现该一元多项式，体会STL使用的好处。

3 可以单独实现一个一元多项式类来实现。

功能要求：

* 能够实现一元多项式的输入和输出
* 能够进行一元多项式相加
* 能够计算一元多项式在x处的值
* 能够计算一元多项式的导数（选作）

编写测试main()函数测试线性表的正确性

## 3代码要求

1、必须要有异常处理，比如删除空链表时需要抛出异常；

2、保持良好的编程的风格：

* 代码段与段之间要有空行和缩近
* 标识符名称应该与其代表的意义一致
* 函数名之前应该添加注释说明该函数的功能
* 关键代码应添加注释说明其功能

3、代码中需要标注每一个函数的时间复杂度

# 实验二 树

## 1 实验目的

通过选择下面两个题目之一进行实现，掌握如下内容：

* 掌握二叉树基本操作的实现方法
* 了解赫夫曼树的思想和相关概念
* 学习使用二叉树解决实际问题的能力

## 2 实验内容

### 2.1 题目1——基础实验

根据二叉树的抽象数据类型的定义，使用二叉链表实现一个二叉树。

完成二叉树的基本功能：

1、二叉树的建立

2、二叉树的复制

3、前序遍历二叉树

4、中序遍历二叉树

5、后序遍历二叉树

6、按层序遍历二叉树

7、求二叉树的结点数

8、二叉树的销毁

编写测试main()函数测试线性表的正确性。

思考问题（选作）：

1、若数据量非常大，如何使得构造二叉树时栈不溢出？使用非递归方式编写新的二叉树的构造函数，建立二叉树。提示：可以使用STL中的stack来辅助实现。

2、若二叉树的每一个结点具有数值，如何搜索二叉树，找到指定值的叶子结点？

3、若已知叶子结点的指针，如何输出从根到该叶子的路径？

### 2.2 题目2——基础实验

根据二叉排序树的抽象数据类型的定义，使用二叉链表实现一个二叉排序树。提示：二叉排序树的理论内容见教材第7章。

二叉排序树的基本功能：

1、二叉排序树的建立

2、二叉排序树的查找

3、二叉排序树的插入

4、二叉排序树的删除

5、二叉排序树的销毁

编写测试main()函数测试二叉排序树的正确性。

思考问题（选作）：

1. 如何判断建立的二叉排序树是否平衡？编写一个函数，完成该功能。

## 3代码要求

1、必须要有异常处理，比如删除空链表时需要抛出异常；

2、保持良好的编程的风格：

* 代码段与段之间要有空行和缩近
* 标识符名称应该与其代表的意义一致
* 函数名之前应该添加注释说明该函数的功能
* 关键代码应添加注释说明其功能

3、递归程序注意调用的过程，防止栈溢出

4、代码中需要标注每一个函数的时间复杂度

# 实验三 图

## 1 实验目的

通过完成图的相关算法的实现，掌握如下内容：

* 掌握图的两种基本的存储结构，以及图的基本算法的实现方法
* 了解最小生成树的思想和相关概念
* 了解最短路径的思想和相关概念
* 学习使用图解决实际问题的能力

## 2 实验内容

### 2.1 题目1——基础实验

根据图的抽象数据类型的定义，使用邻接矩阵或邻接表实现一个图。

图的基本功能：

1、图的建立

2、图的销毁

3、深度优先遍历图

4、广度优先遍历图

5、使用普里姆算法生成最小生成树

6、使用克鲁斯卡尔算法生成最小生成树

7、求指定顶点到其他各顶点的最短路径

编写测试main()函数测试图的正确性

思考问题（选作）：

1、若测试数据量较大，如何使得栈不溢出？使用非递归方式编写新的深度优先遍历函数。提示：可以使用STL中的stack来辅助实现。

2、最短路径D算法，是否可以优化？请写出优化的思路并计算时间复杂度，同时实现一个新的优化的最短路径算法。

## 3代码要求

1、必须要有异常处理，比如删除空链表时需要抛出异常；

2、保持良好的编程的风格：

* 代码段与段之间要有空行和缩近
* 标识符名称应该与其代表的意义一致
* 函数名之前应该添加注释说明该函数的功能
* 关键代码应添加注释说明其功能

3、递归程序注意调用的过程，防止栈溢出

# 实验四

## 1 实验目的

本实验为可选实验，用于提高同学们使用数据结构解决实际问题的能力。通过选择下面5个题目之一进行实现，掌握如下内容：

* 深度了解内存分配和释放的原理
* 熟练掌握栈和递归的进一步应用
* 学习矩阵的相关算法在BMP图像中的应用
* 深入掌握二叉树在哈夫曼编码中的算法实现
* 了解图的相关算法的应用
* 进一步提高编程能力

### 4.1题目1——动态内存管理

动态内存管理是操作系统的基本功能之一，用于响应用户程序对内存的申请和释放请求。初始化时，系统只有一块连续的空闲内存；然后，当不断有用户申请内存时，系统会根据某种策略选择一块合适的连续内存供用户程序使用；当用户程序释放内存时，系统将其回收，供以后重新分配，释放时需要计算该内存块的左右是否也为空闲块，若是，则需要合并变成更大的空闲块。

试设计用于模拟动态内存管理的内存池类。

基本要求：

1. 实现内存池MemoryPool(int size)的初始化
2. 实现Allocate(int size)接口
3. 实现Free(void \*p)接口
4. 实现内存池的析构
5. 在分配内存空间时，可选择不同的内存分配策略：最佳拟合策略、最差拟合策略或最先拟合策略。实现其中至少两种分配策略。

编写测试main()函数对类中各个接口和各种分配策略进行测试，并实时显示内存池中的占用块和空闲块的变化情况。

### 4.2题目2——表达式求值

表达式求值是程序设计语言编译中最近本的问题，它要求把一个表达式翻译成能够直接求值的序列。例如用户输入字符串“14+((13-2)\*2-11\*5)\*2”，程序可以自动计算得到最终的结果。在这里，我们将问题简化，假定算数表达式的值均为非负整数常数，不包含变量、小数和字符常量。

试设计一个算术四则运算表达式求值的简单计算器。

基本要求：

1、 操作数均为非负整数常数，操作符仅为+、-、\*、/、（和）；

2、 编写main函数进行测试。

### 4.3题目3——bmp图像处理

实现一个识别BMP文件的图像类，能够进行以下图像处理。

基本要求：

1、能够将24位真彩色Bmp文件读入内存；

2、能够将24位真彩色Bmp文件重新写入文件；

3、能够将24位真彩色Bmp文件进行24位灰度处理；

4、能够将24位灰度Bmp文件进行8位灰度处理；

5、能够将8位灰度Bmp文件转化成黑白图像；

6、能够将图像进行平滑处理；

7、其他：自定义操作，比如翻转、亮度调节、对比度调节、24位真彩色转256色等。

提示：

1、参考教材《数据结构与STL》第四章4.4小节。

2、灰度处理的转换公式

Grey=0.3\*Red+0.59\*Blue+0.11\*Green

3、平滑处理采用邻域平均法进行，分成4邻域和8邻域平滑，基本原理就是将每一个像素点的值设置为其周围各点像素值得平均值。

4、亮度调节公式，a为亮度调节参数，0 < a < 1, 越接近 0，变化越大

R = pow (R, a) \* pow (255, 1 - a)

G = pow (G, a) \* pow (255, 1 - a)

B = pow (B, a) \* pow (255, 1 - a)

5、对比度调节公式，a为对比度调节参数，-1< a < 1，（中间值一般为128）

R=中间值+(R-中间值)\*(1+a)

G=中间值+(G-中间值)\*(1+a)

B=中间值+(B-中间值)\*(1+a)

注意：调整对比度的时候容易发生越界，需要进行边界处理

6、24位真彩色转256色，需要手动添加颜色表在BMP头结构中，可以使用位截断法、流行色算法、中位切分算法、八叉树算法等方法实现。

### 4.4题目4——哈夫曼树

利用二叉树结构实现哈夫曼编/解码器。

基本要求：

1、 初始化(Init)：能够对输入的任意长度的字符串s进行统计，统计每个字符的频度，并建立赫夫曼树

2、 建立编码表(CreateTable)：利用已经建好的赫夫曼树进行编码，并将每个字符的编码输出。

3、 编码(Encoding)：根据编码表对输入的字符串进行编码，并将编码后的字符串输出。

4、 译码(Decoding)：利用已经建好的赫夫曼树对编码后的字符串进行译码，并输出译码结果。

5、 打印(Print)：以直观的方式打印赫夫曼树（选作）

6、 计算输入的字符串编码前和编码后的长度，并进行分析，讨论赫夫曼编码的压缩效果。

测试数据：

I love data Structure, I love Computer。I will try my best to study data Structure.

提示：

1、用户界面可以设计为“菜单”方式：能够进行交互。

2、根据输入的字符串中每个字符出现的次数统计频度，对没有出现的

字符一律不用编码。

### 4.5题目5——运动会比赛问题

问题：设某个田径运动会共有七个项目的比赛，分别为100米、200米、跳高、跳远、铅球、铁饼和标枪。每个选手最多参加3个项目，现有六名选手参赛，他们选择的项目如表1-1所示。考虑到每个选手的参加的各个项目不能同时进行，则如何设计合理的比赛日程，使运动会在尽可能短的时间内完成？

|  |  |  |  |
| --- | --- | --- | --- |
| **姓名** | **项目1** | **项目2** | **项目3** |
| 张凯 | 跳高 | 跳远 |  |
| 王刚 | 100m | 200m | 铁饼 |
| 李四 | 跳高 | 铅球 |  |
| 张三 | 跳远 | 标枪 |  |
| 王峰 | 铅球 | 标枪 | 铁饼 |
| 李杰 | 100m | 跳远 |  |

测试数据：

提示：

1、利用图的着色思想解决该问题

2、可以使用STL相关内容辅助解决该问题

## 3代码要求

1、必须要有异常处理，比如删除空链表时需要抛出异常；

2、保持良好的编程的风格：

* 代码段与段之间要有空行和缩近
* 标识符名称应该与其代表的意义一致
* 函数名之前应该添加注释说明该函数的功能
* 关键代码应添加注释说明其功能

3、递归程序注意调用的过程，防止栈溢出