

**数 据 结 构 实 验**

**计算机科学与技术学院**

**数组结构课程组**

目录

[1 线性表实验 1](#_Toc427727637)

[1.1 实验目的 1](#_Toc427727638)

[1.2 线性表基本操作定义 1](#_Toc427727639)

[1.3 实验任务 2](#_Toc427727640)

[2 栈与队列实验 3](#_Toc427727641)

[2.1 实验目的 3](#_Toc427727642)

[2.2 栈 3](#_Toc427727643)

[2.2.1 栈基本操作定义 3](#_Toc427727644)

[2.2.2 栈的实验任务 3](#_Toc427727645)

[2.3 队列 5](#_Toc427727646)

[2.3.1 队列基本操作定义 5](#_Toc427727647)

[2.3.2 队列的实验任务 5](#_Toc427727648)

[3 字符串实验 6](#_Toc427727649)

[3.1 实验目的 6](#_Toc427727650)

[3.2 字符串基本操作定义 6](#_Toc427727651)

[3.3 实验任务 7](#_Toc427727652)

[4 数组实验 8](#_Toc427727653)

[4.1 实验目的 8](#_Toc427727654)

[4.2 数组基本操作定义 8](#_Toc427727655)

[4.3 实验任务 8](#_Toc427727656)

[5 二叉树实验 9](#_Toc427727657)

[5.1 实验目的 9](#_Toc427727658)

[5.2 二叉树基本操作定义 9](#_Toc427727659)

[5.3 实验任务 11](#_Toc427727660)

[6 图实验 12](#_Toc427727661)

[6.1 实验目的 12](#_Toc427727662)

[6.2 图基本操作定义 12](#_Toc427727663)

[6.3 实验任务 13](#_Toc427727664)

[7 查找表实验 14](#_Toc427727665)

[7.1 实验目的 14](#_Toc427727666)

[7.2 实验任务 14](#_Toc427727667)

[8 排序算法实验 16](#_Toc427727668)

[8.1 实验目的 16](#_Toc427727669)

[8.2 实验任务 16](#_Toc427727670)

[参考文献 17](#_Toc427727671)

# 1 线性表实验

1.1 实验目的

通过实验达到：（1）加深对线性表的概念、基本运算的理解；（2）熟练掌握线性表的逻辑结构与物理结构的关系；（3）熟练掌握线性表的基本运算在不同的物理结构下的实现。

1.2 线性表基本操作定义

（1）InitaList(&L)

操作结果：构造一个空的线性表。

（2）DestroyList(&L)

初始条件：线性表L已存在。

操作结果：销毁线性表L。

（3）ClearList (&L)

初始条件：线性表L已存在。

操作结果：将L重置为空表。

（4）ListEmpty(L)

初始条件：线性表L已存在。

操作结果：若L为空表，则返回TRUE,否则返回FALSE。

（5）ListLength(L)

初始条件：线性表已存在。

操作结果：返回L中数据元素的个数。

（6）GetElem(L, i,&e)

初始条件：线性表已存在，1≤i≤ListLength(L)。

操作结果：用e返回L中第i个数据元素的值。

（7）LocateElem(L, e，compare（）)

初始条件：线性表已存在。

操作结果：返回L中第1个与e满足关系compare（）关系的数据元素的

位序，若这样的数据元素不存在，则返回值为0。

（8）PriorElem（L，cur\_e，&pre\_e）

初始条件：线性表L已存在。

操作结果：若cur\_e是L的数据元素，且不是第一个，则用pre\_e返回它的 前驱，否则操作失败，pre\_e无定义。

（9）NextElem（L，cur\_e，&next\_e）

初始条件：线性表L已存在。

操作结果：若cur\_e是L的数据元素，且不是最后一个，则用next\_e返回它

的后继，否则操作失败，next\_e无定义。

（10）ListInsert(&L,i,e)

初始条件：线性表L已存在且非空，1≤i≤ListLength(L)+1。

操作结果：在L的第i个位置之前插入新的数据元素e，L的长度加1

（11）ListDelete(&L,i,&e)

初始条件：线性表L已存在且非空，1≤i≤ListLength(L)。

操作结果：删除L的第i个数据元素，用e返回其值，L的长度减1.

（12）ListTraverse(L，visit（）)

初始条件：线性表L已存在。

操作结果：依次对L的每个数据元素调用函数visit()。一旦调用失败，则操

作失败。

1.3 实验任务

采用不同的物理结构实现线性表的基本操作。其中ElemType为数据元素的类型名，具体含义可自行定义，其它有关类型和常量的定义和引用详见文献[1]的p10。要求构造成具有功能菜单的系统，线性表采用文件形式保存。

实验内容如下：

1. 以顺序表作为物理结构，实现线性表的基本操作。

2. 以带表头结点的单链表作为物理结构，实现线性表的基本操作。

3. (替换选择题，如选择3，则不需要完成1和2)多个数据元素类型相同的线性表的管理。其中每一个线性表采用以带表头结点的单链表作为物理结构，每一个线性表的属性至少包含有表名称。采用顺序表的方式管理多个线性表。其物理结构如图1-1所示。



图1-1 多线性表的物理结构示意图

要求能对每个线性表实现线性表的基本操作；按自己的理解适当增加一些功能，如查找、统计等；多个线性表采用1个文件保存。

# 2 栈与队列实验

2.1 实验目的

通过实验达到：（1）加深对栈与队列的概念、基本运算的理解；（2）熟练掌握栈与队列的逻辑结构与物理结构的关系；（3）熟练掌握栈与队列的基本运算在不同的物理结构下的实现；（4）通过应用体会栈与队列的功能。

2.2 栈

2.2.1 栈基本操作定义

（1）InitStack(&S)

操作结果：构造一个空栈S。

（2）DestroyStack(&S)

初始条件：栈S存在。

操作结果：栈S被销毁，不在存在。

（3）ClearStack(&S)

初始条件：栈S存在。

操作结果：将S清成空栈。

（4）StackEmpty(S)

初始条件：栈S存在。

操作结果：若S为空栈,则返回值为TRUE；否则为FALSE。

（5）StackLength(S)

初始条件：栈S存在。

操作结果：返回栈S的元素个数。

（6）GetTop(S,&e)

初始条件：栈S存在并且非空。

操作结果：将栈顶元素拷贝到e。

（7）Push(&S,e)

初始条件：栈S存在。

操作结果：插入元素e为新的栈顶元素。

（8）Pop(&S,&e)

初始条件：栈S存在并且非空。

操作结果：删除栈S的栈顶元素，并送入e。

（9）StackTravserse(S,visit())

初始条件：栈S存在。

操作结果：从栈底到栈顶依次对栈S中的元素使用函数visit进行访问。

2.2.2 栈的实验任务

采用不同的物理结构实现栈的基本操作。

1. 以顺序栈作为栈的物理结构，实现2.2.1所列栈的全部操作。建议使用模板[3] 的方式实现。

2. 以链式栈作为栈的物理结构，实现2.2.1所列栈的全部操作。建议使用模板的方式实现。

3. 利用栈，实现表达式的求值，这里仅要求对含运算符+、-、\*、/和1位数的操作数的表达式求值。

2.3 队列

2.3.1 队列基本操作定义

（1）InitQueue(&Q)

操作结果：构造一个空队列Q。

（2） DestroyQueue(&Q)

初始条件：队列Q存在。

操作结果：将队列Q销毁，不再存在。

（3）ClearQueue(&Q)

初始条件：队列Q存在。

操作结果：将队列Q清为空队列。

（4）QueueEmpty(Q)

初始条件：队列Q存在。

操作结果：若队列Q为空队列，返回TRUE，否则返回FALSE。

（5）QueueLength(Q)

初始条件：队列Q存在。

操作结果：返回队列元素个数。

（6）GetHead(Q,&e)

初始条件：队列Q存在并且非空。

操作结果：读取队列Q的首元素,送e返回其值。

（7）EnQueue(&Q,e)

初始条件：队列Q存在。

操作结果：插入元素e到队列Q中作为尾元素。

（8）DeQueue(&Q,&e)

初始条件：队列Q存在并且非空。

操作结果：删除队列Q的首元素,并且用e返回其值。

（9）QueueTravserse(Q,visit())

初始条件：队列Q存在并且非空。

操作结果：从队首到队尾依次对队列中的元素使用函数visit进行访问。

2.3.2 队列的实验任务

采用不同的物理结构实现队列的基本操作。

1. 以链式队列作为队列的物理结构，实现2.3.1所列队列的全部操作。建议使用模板的方式实现。

2. 以循环队列作为队列的物理结构，实现2.3.1所列队列的全部操作。建议使用模板的方式实现。

# 3 字符串实验

3.1 实验目的

通过实验达到：（1）加深对字符串的概念、基本运算的理解；（2）熟练掌握字符串的逻辑结构与物理结构的关系；（3）熟练掌握字符串的基本运算在定长物理结构下的实现。

3.2 字符串基本操作定义

（1）StrAssign(&T,chars)

初始条件：chars是字符串常量。

操作结果：生成一个其值等于chars的字符串T。

（2）StrCopy(&T,S)

初始条件：字符串S存在。

操作结果：由字符串S赋值得到字符串T。

（3）StrEmpty(S)

初始条件：字符串S存在。

操作结果：字符串S为空返回TRUE，否则返回FALSE。

（4）StrCompare(S,T)

初始条件：字符串S、T存在。

操作结果：若S>T，则返回值>0若S=T，则返回值=0若S<T，则返回值<0。

（5）StrLength(S)

初始条件：字符串S存在。

操作结果：返回字符串元素个数。

（6）ClearString(&S)

初始条件：字符串S存在。

操作结果：将S清为空串。

（7） Concat(&T,S1,S2)

初始条件：字符串S1、S2存在。

操作结果：用T返回S1和S2连接的新串。

（8）SubString(&sub,S,pos,len)

初始条件：字符串S存在，**1≤pos≤StrLength(S)+1**

**且0≤len≤StrLength(S)-pos+1**。

操作结果：用Sub返回S从pos开始的长度为len的子串。

（9）Index(S,T,pos)

初始条件：字符串S、T存在，T是非空串，**1≤pos≤StrLength(S)。**

操作结果：若主串S存在与T相同的子串，则返回T在主串S的第pos个

字符之后第一次出现的位置，否则返回0。

（10）Replace(&S,T,V)

初始条件：字符串S、T和V存在，T是非空串。

操作结果：用V替换主串S中所有与T相等的不重叠子串。

（11） StrInsert(&S,pos,T)

初始条件：字符串S、T存在，**1≤pos≤StrLength(S)+1。**。

操作结果：在字符串S的第pos个字符前插入T。

（12） StrDelete(&S,pos,len)

初始条件：字符串S、T存在，1≤pos≤StrLength(S)-len+1。

操作结果：删除S从第pos个字符开始，长度为len的子串。

（13）DestroyString(&S)

初始条件：字符串S存在。

操作结果：字符串S被销毁。

3.3 实验任务

采用定长字符串S物理结构，管理多个字符串，实现字符串的基本操作。要求构造成具有功能菜单的系统。

# 4 数组实验

4.1 实验目的

通过实验达到：（1）加深对数组的概念、基本运算的理解；（2）熟练掌握数组的逻辑结构与物理结构的关系；（3）熟练掌握数组的基本运算在不同的物理结构下的实现。

4.2 数组基本操作定义

（1）InitArray(&A，n，bound1，… ，boundn)

操作结果：若维数n个各维长度合法，（这里约定下界为0）创建数组A，

并返回OK。

（2）DestroyArray(&A)

操作结果：销毁数组A。

（3）Value(A，index1，…，indexn)

＝…A[index1，…，indexn] …

初始条件：A是n维数组，e是元素变量，随后是n个下标值。

操作结果：若n个下标值都没越界，将对应的下标变量值赋值给e，并

返回OK。

（4）Assign(A，e ，index1，…，indexn)

初始条件：A是n维数组，e是元素变量，随后是n个下标值。

操作结果：若n个下标值都没越界，将e赋值给对应的下标变量，并

返回OK。

4.3 实验任务

采用不同的物理结构，实现稀疏矩阵的基本操作。要求构造成具有功能菜单的系统，稀疏矩阵采用文件形式保存。实验内容：

1. 以三元组表作为稀疏矩阵的物理结构，实现稀疏矩阵的全部基本操作。

2. 以十字链表作为稀疏矩阵的物理结构，实现稀疏矩阵的全部基本操作。

# 5 二叉树实验

5.1 实验目的

通过实验达到：（1）加深对二叉树的概念、基本运算的理解；（2）熟练掌握二叉树的逻辑结构与物理结构的关系；（3）熟练掌握以二叉链表作为物理结构的二叉树基本运算的实现。

5.2 二叉树基本操作定义

（1）InitBiTree(&T)

操作结果：构造空二叉树T。

（2）DestroyBiTree(&T)

初始条件：二叉树T已存在。

操作结果：销毁二叉树T。

（3）CreateBiTree(&T,definition)

初始条件：definition 给出二叉树T的定义。

操作结果：按definition构造二叉树T。

（4）ClearBiTree (&T)

初始条件：二叉树T存在。

操作结果：将二叉树T清空。

（5）BiTreeEmpty(T)

初始条件：二叉树T存在。

操作结果：若T为空二叉树，则返回TRUE,否则返回FALSE.

（6）BiTreeDepth(T)

初始条件：二叉树T存在。

操作结果：返回T的深度。

（7）Root(T)

初始条件：二叉树T已存在。

操作结果：返回T的根。

（8）Value(T，e)

初始条件：二叉树T已存在，e是T中的某个结点。

操作结果：返回e的值。

（9）Assign(T，&e，value)

初始条件：二叉树T已存在，e是T中的某个结点。

操作结果：结点e赋值为value。

（10）Parent(T，e)

初始条件：二叉树T已存在，e是T中的某个结点。

操作结果：若e是T的非根结点，则返回它的双亲，否则返回空。

（11）LeftChild(T，e)

初始条件：二叉树T存在，e是T中某个节点。

操作结果：返回e的左孩子。若e无左孩子，则返回“空”。

（12）RightChild(T，e)

初始条件：二叉树T已存在，e是T中某个结点。

操作结果：返回e的右孩子。若e无右孩子，则返回“空”。

（13）LeftSibling(T，e)

初始条件：二叉树T存在，e是T中某个节点。

操作结果：返回e的左兄弟。若e是T的左孩子或者无左兄弟，则返回“空”。

（14）RightSibling(T，e)

初始条件：二叉树T已存在，e是T中某个结点。

操作结果：返回e的右兄弟。若e是T的右孩子或者无有兄弟，则返回“空”。

（15）InsertChild(T,p,LR,c)

初始条件：二叉树T存在，p指向T中的某个结点，LR为0或1，,非空二叉树c与T不相交且右子树为空。

操作结果：根据LR为0或者1，插入c为T中p所指结点的左或右子树，p所指结点的原有左子树或右子树则为c的右子树。

（16）DeleteChild(T，p，LR)

初始条件：二叉树T存在，p指向T中的某个结点，LR为0或1。

操作结果：根据LR为0或者1，删除c为T中p所指结点的左或右子树。

（17）PreOrderTraverse（T,Visit（））

初始条件：二叉树T存在，Visit是对结点操作的应用函数。

操作结果：先序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

（18）InOrderTraverse（T,Visit（））

初始条件：二叉树T存在，Visit是对结点操作的应用函数。

操作结果：中序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

（19）PostOrderTraverse（T,Visit（））

初始条件：二叉树T存在，Visit是对结点操作的应用函数。

操作结果：后序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

（20）LevelOrderTraverse（T,Visit（））

初始条件：二叉树T存在，Visit是对结点操作的应用函数。

操作结果：层序遍历t，对每个结点调用函数Visit一次且一次，一旦调用失败，则操作失败。

5.3 实验任务

实现对多个结点元素类型相同的二叉树的管理。其中每一个二叉树采用二叉链表作为物理结构，每一个二叉树的属性至少包含有二叉树名称。采用顺序表的方式管理多个线性表。其物理结构如图5-1所示。



图5-1 多二叉树管理的物理结构示意图

要求构造成具有功能菜单的系统，能对每棵二叉树实现二叉树的基本操作；按自己的理解适当增加一些功能，如查找、统计等；多个二叉树采用文件保存。

# 6 图实验

6.1 实验目的

通过实验达到：（1）加深对图的概念、基本运算的理解；（2）熟练掌握图的逻辑结构与物理结构的关系；（3）熟练掌握图的基本运算在不同的物理结构下的实现。

6.2 图基本操作定义

（1）CreateCraph(&G,V,VR)

初始条件：V是图的顶点集，VR是图的关系集。

操作结果：按V和VR的定义构造图G。

（2）DestroyCraph(&G)

初始条件：图G存在。

操作结果：销毁图G。

（3）LocateVex(G,u)

初始条件：图G存在，u和G中的顶点具有相同特征。

操作结果：若u在图G中存在，返回顶点u的位置信息，否则返回其它信息。

（4）FirstAdjVex(&G, v)

初始条件：图G存在，v是G的一个顶点。

操作结果：返回v的第一个邻接顶点，如果v没有邻接顶点，返回空。

（5）NextAdjVex(&G, v, w)

初始条件：图G存在，v是G的一个顶点,w是v的邻接顶点。

操作结果：返回v的（相对于w）下一个邻接顶点，如果w是最后一个邻接

顶点，返回空。

（6）InsertVex(&G,v)

初始条件：图G存在，v和G中的顶点具有相同特征。

操作结果：在图G中增加新顶点v。

（7）DeleteVex(&G,v)

初始条件：图G存在，v是G的一个顶点。

操作结果：在图G中删除顶点v和与v相关的弧。

（8）InsertArc(&G,v,w)

初始条件：图G存在，v、w是G的顶点。

操作结果：在图G中增加弧<v,w>，如果图G是无向图，还需要增加<w,v>。

（9）DeleteArc(&G,v,w)

初始条件：图G存在，v、w是G的顶点。

操作结果：在图G中删除弧<v,w>，如果图G是无向图，还需要删除<w,v>。

（10）DFSTraverse(G,visit())

初始条件：图G存在。

操作结果：对图G进行深度优先搜索遍历，依次对图中的每一个顶点使用函

数visit访问一次，且仅访问一次。

（11）BFSTraverse(G,visit())

初始条件：图G存在。

操作结果：对图G进行广度优先搜索遍历，依次对图中的每一个顶点使用函

数visit访问一次，且仅访问一次。

6.3 实验任务

采用图的不同物理结构表示法（Representation），实现无向（或有向）图（或网）的基本运算。构造成具有功能菜单的系统，采用文件形式保存图。以下实验任务任选其一。

1. 采用邻接矩阵法，实现无向（或有向）图（或网）的基本运算。

2. 采用邻接表，实现无向（或有向）图（或网）的基本运算。

3. 采用十字链表，实现有向图（或网）的基本运算。

4. 采用邻接多种表，实现无向图（或网）的基本运算。

# 7 查找表实验

7.1 实验目的

通过实验达到：（1）加深对查找表的概念、基本运算的理解；（2）熟练掌握不同类型查找表的基本运算的实现。

7.2 实验任务

要求将查找表的数据元素定义成结构型数据，具体形式为:

typedef 关键字的类型 KeyType；

typedef struct[] {

KeyType key；

其它成员部分；

} ElemType；

以下实验任务任选其二。要求构造成具有功能菜单的系统，采用文件形式保存查找表。

1. 采用有序表，实现如下查找表的基本运算。

（1）Create(&ST,n)

操作结果：构造一个有n个元素的有序表ST。

（2）Destroy(&ST)

初始条件：有序表ST存在。

操作结果：销毁有序表ST。

（3）SearchElem(&ST, k)

初始条件：有序表ST存在。

操作结果：在有序表ST中查找关键字为k的数据元素。

（4）ASL(&ST)

初始条件：有序表ST存在。

操作结果：计算有序表ST查找成功的ASL。

2. 采用二叉排序树，实现如下查找表的基本运算。

（1）Create(&ST,n)

操作结果：构造一个有n个元素的二叉排序树ST。

（2）Destroy(&ST)

初始条件：二叉排序树ST存在。

操作结果：销毁二叉排序树ST。

（3）InsertNode(&ST,e)

初始条件：二叉排序树ST存在。

操作结果：增加一个数据元素e到二叉排序树ST。

（4）DeleteNode(&ST,k)

初始条件：二叉排序树ST存在。

操作结果：从二叉排序树ST中删除关键字为k的数据元素e。

（5）SearchBiTree(&ST,k)

初始条件：二叉排序树ST存在。

操作结果：在二叉排序树ST中查找关键字为k的数据元素。

（6）ASL(&ST)

初始条件：二叉排序树ST存在。

操作结果：计算二叉排序树ST查找成功的ASL。

3. 采用HASH表，实现如下查找表的基本运算。

（1）Create(&ST,n)

操作结果：构造一个有n个元素的HASH表ST。

（2）Destroy(&ST)

初始条件：HASH表ST存在。

操作结果：销毁HASH表ST。

（3）InsertElem(&ST,e)

初始条件：HASH表ST存在。

操作结果：增加一个数据元素e到HASH表ST。

（4）DeleteElem(&ST,k)

初始条件：HASH表ST存在。

操作结果：从HASH表ST中删除关键字为k的数据元素e。

（5）SearchHashTable(&ST,k)

初始条件：二叉排序树ST存在。

操作结果：在二叉排序树ST中查找关键字为k的数据元素。

（6）ASL(&ST)

初始条件：HASH表ST存在。

操作结果：计算HASH表ST查找成功的ASL。

# 8 排序算法实验

8.1 实验目的

通过实验达到：（1）加深对排序的概念的理解；（2）熟练掌握几种排序算法的实现。

8.2 实验任务

要求将待排序的数据元素定义成结构型数据，具体形式为:

typedef 关键字的类型 KeyType；

typedef struct[] {

KeyType key；

其它成员部分；

} ElemType；

下列排序算法中，选取4个（含4个）以上实现，其中至少包括2个时间复杂度为O(n log n)的。

1. 简单插入排序

2. 希尔排序

3. 简单选择排序

4. 堆排序

5. 冒泡排序

6. 快速排序

7. 归并排序

8. 基数排序

要求：（1）构造成具有功能菜单的系统；（2）能随机生成一定规模的待排序记录序列；（3）在各规模下，统计各排序算法完成排序的实测数据（平均值），在报告中，结合理论分析，对各算法的实测数据进行对比分析。

参考文献

[1] 严蔚敏等.数据结构（C语言版）.清华大学出版社

[2] [Larry Nyhoff](http://www.calvin.edu/~nyhl/index.html). [ADTs, Data Structures, and Problem Solving with C++.](http://vig.prenhall.com/catalog/academic/product/0,1144,0131409093,00.html)Second Edition, [Calvin College](http://cs.calvin.edu/),2005

[3] 殷立峰. Qt C++跨平台图形界面程序设计基础. 清华大学出版社,2014:192～197

[4] 严蔚敏等.数据结构题集（C语言版）.清华大学出版社