

**《数据结构》课程**

**上机实验指导书**

**院 系：** 人工智能学院

**专　　业：** 计算机科学与技术

**课　　程：** 数据结构

**班　　级:**

**制 定 人：**  李言一

**制定时间：**  2024年8月

**目 录**

[实验一 顺序表的定义和运算 1](#_Toc145493647)

[实验二 链表的定义和运算 4](#_Toc145493648)

[实验三 栈的定义和运算 8](#_Toc145493649)

[实验四 队列的定义和操作 12](#_Toc145493650)

[实验五 二叉树的基本操作 17](#_Toc145493651)

[实验六 图的构造与遍历 22](#_Toc145493652)

[实验七 查找算法 28](#_Toc145493653)

[实验八 排序算法 31](#_Toc145493654)

实验一 顺序表的定义和运算

**一、实验目的：**

1.掌握线性表的逻辑特征。

2.掌握线性表顺序存储结构的特点

3.熟练掌握顺序表的基本运算。

4.能编程实现顺序线性表的创建、插入和删除。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现顺序表的创建、插入和删除等基本操作算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

1.顺序表的存储结构定义。

2.创建顺序表，从键盘上依次输入21、18、30、75、42、56，并输出顺序表中的各元素值。

3.分别在顺序表的第3个位置和第9个位置插入67和10，给出插入成功或失败的信息，并输出顺序表中的各元素值。

4.删除顺序表中的第6个数据元素和第8个数据元素，给出删除成功或失败的信息，并输出顺序表中的各元素值。

**五、实施步骤与技术要点**

1.顺序表结构的定义：

#define MaxSize 50

typedef int ElemType;

typedef struct SqList

{

ElemType data[MaxSize];

int length;

}SqList;

2.初始化顺序表

void InitList(SqList \*L) /\* 初始化顺序表 \*/

{

L->data[MaxSize]=(int\*)malloc(sizeof(int)\*MaxSize);

L->length=0;

}

3.顺序表插入（在第i号元素前插入一个新的元素）

int ListInsert(SqList \*L,int i,ElemType e) /\*按i位置插入元素\*/

{

int j;

if(i<1||i>L->length+1)

return 0;

for (j=L->length-1;j>i-1;j--)

L->data[j+1]=L->data[j];

L->data[i-1]=e;

L->length++;

return 1;

}

4.顺序表删除

int ListDelete(SqList \*L,int i) /\*删除某个位置的元素\*/

{

int j;

if (i<1||i>L->length)

return 0;

for(j=i-1;j<L->length-1;j++)

L->data[j]=L->data[j+1];

L->length--;

return 1;

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验二 链表的定义和运算

**一、实验目的：**

1.掌握线性表的逻辑特征。

2.掌握单链表存储结构的特点

3.熟练掌握单链表的基本运算。

4.能编程实现单链表的创建、插入和删除。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现单链表的创建、插入和删除等基本操作算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

1.单链表的存储结构定义。

2.从键盘上依次输入21、18、30、75、42、56，逆序创建单链表（前插法），并输出单链表中的各元素值。

3.分别在单链表的第3个位置和第9个位置插入67和10，给出插入成功或失败的信息，并输出单链表中的各元素值。

4.删除单链表中的第6个数据元素和第8个数据元素，给出删除成功或失败的信息，并输出单链表中的各元素值。

**五、实施步骤与技术要点**

1. 单链表的结构定义：

typedef int ElemType;

//声明节点结构

typedef struct LinkList {

ElemType elem;//存储整形元素

struct LinkList \*next;//指向直接后继元素的指针

}LinkList;

2. 初始化链表

void InitList(LinkList \*L) /\* 初始化链表 \*/

{

L=(struct LinkList\*)malloc(sizeof(struct LinkList\*));

L->next=NULL;

}

3.建立单链表的算法

void CreateList(LinkList \*L,ElemType a[],int n) /\*创建链表 \*/

{

LinkList \*pre=L;

int i;

for(i=n-1;i>=0;i--){

LinkList \*t=(struct LinkList\*)malloc(sizeof(struct LinkList\*));

t->elem=a[i];

t->next=NULL;

pre->next=t;

pre=pre->next;

}

}

3.单链表的插入算法

int ListInsert(LinkList \*L,int i,ElemType e) /\*按i位置插入元素\*/

{

int j=0;

LinkList \*p=L;

while( p &&(j<i-1) ){

p=p->next;

j++;

}

if(!p ||(j>i-1))

return 0;

LinkList \*t=(struct LinkList\*)malloc(sizeof(struct LinkList\*));

t->elem=e;

t->next=p->next;

p->next=t;

return 1;

}

4.单链表的删除算法

int ListDelete(LinkList \*L,int i) /\*删除某个位置的元素\*/

{

int j=0;

LinkList \*p=L;

LinkList \*del=NULL;

while( (p->next) &&(j<i-1) ){

p=p->next;

j++;

}

if(!(p->next) ||(j>i-1))

return 0;

del = p->next;

p->next=del->next;

free(del);

return 1;

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验三 栈的定义和运算

**一、实验目的：**

1.熟练掌握栈的特点。

2.掌握栈的定义和基本操作。

3.熟练掌握顺序栈和链栈的操作及应用。

4.能编程实现顺序栈和链栈的创建、进栈和出栈等。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现栈的基本操作算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

**任务1：栈的顺序表示和实现**

具体要求：编写一个程序实现顺序栈的各种基本运算，完成如下功能：

1.依次进栈数据为1,2,3,4,5，再全部出栈，输出出栈序列。

2.先进栈3,2,1，出栈2次，进栈4,5,6,7，再全部出栈，输出每次入栈，出栈序列。

**任务2：栈的链式表示和实现**

具体要求：编写一个程序实现链栈的各种基本运算，完成功能同顺序栈。

**五、实施步骤与技术要点**

1.顺序栈的结构定义

#define MAXSIZE 100

typedef struct {

int data[MAXSIZE];

int top;

} Stack;

2.顺序栈的初始化算法

void InitStack(Stack \*S) {

S->top = -1;

}

3.顺序栈进栈操作算法

int Push(Stack \*S, int x) {

if (S->top == MAXSIZE - 1) {

return 0;

}

S->data[++S->top] = x;

return 1;

}

4.顺序栈出栈操作算法

int Pop(Stack \*S, int \*x) {

if (StackEmpty(S)) {

return 0;

}

\*x = S->data[S->top--];

return 1;

}

5.链栈的结构定义

typedef struct StackNode {

int data;

struct StackNode\* next;

} StackNode, \*LinkStack;

6.链栈的初始化

LinkStack CreateStack() {

LinkStack s = (LinkStack)malloc(sizeof(StackNode));

s->next = NULL;

return s;

}

7.链栈的进栈操作算法

int Push(LinkStack s, int x) {

StackNode\* node = (StackNode\*)malloc(sizeof(StackNode));

node->data = x;

node->next = s->next;

s->next = node;

return 1;

}

8.链栈的出栈操作算法

int Pop(LinkStack s, int\* x) {

if (s->next == NULL) {

return 0;

}

StackNode\* node = s->next;

\*x = node->data;

s->next = node->next;

free(node);

return 1;

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验四 队列的定义和操作

**一、实验目的：**

1.熟练掌握队列的特点（先进先出）。

2.掌握队列的定义和基本操作，如入队、出队等。

3.熟练掌握顺序（循环）队列和链队列的操作及应用。

4.能编程实现顺序（循环）队列和链队列的创建、入队和出队等。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现队列的基本操作算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

**任务1：队列的顺序表示和实现**

具体要求：编写一个程序实现循环队列的各种基本运算，完成如下功能：

1.依次入队数据为1,2,3,4,5，再全部出队，输出出队队列序列。

2.先入队数据3,2,1，出队2次，入队4,5,6,7，最后所有数据出队，输出每次入队和出队序列。

**任务2：队列的链式表示和实现**

具体要求：编写一个程序实现链队列的各种基本运算，完成功能同顺序（循环）队列。

**五、实施步骤与技术要点**

1.循环队列的结构定义：

#define MAXSIZE 100

typedef struct {

int data[MAXSIZE];

int front;

int rear;

} Queue;

2.初始化循环队列

void CreateQueue(Queue \*Q) {

Q->front = Q->rear = 0;

}

3.循环队列的进队列操作算法

int EnQueue(Queue \*Q, int x) {

if (IsFull(\*Q)) {

return 0;

}

Q->data[Q->rear] = x;

Q->rear = (Q->rear + 1) % MAXSIZE;

return 1;

}

4. 循环队列的出队列操作算法

int DeQueue(Queue \*Q, int \*x) {

if (IsEmpty(\*Q)) {

return 0;

}

\*x = Q->data[Q->front];

Q->front = (Q->front + 1) % MAXSIZE;

return 1;

}

5. 链队列的结构定义

typedef struct Node {

int data;

struct Node\* next;

} Node, \*LinkQueue;

6. 链队列的初始化

LinkQueue CreateQueue() {

LinkQueue q = (LinkQueue)malloc(sizeof(Node));

q->next = NULL;

return q;

}

7. 链队列的入队操作

int EnQueue(LinkQueue\* q, int x) {

Node\* node = (Node\*)malloc(sizeof(Node));

node->data = x;

node->next = (\*q)->next;

(\*q)->next = node;

return 1;

}

8. 链队列的出队操作

int DeQueue(LinkQueue\* q, int\* x) {

if ((\*q)->next == NULL) {

return 0;

}

Node\* node = (\*q)->next;

\*x = node->data;

(\*q)->next = node->next;

free(node);

return 1;

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验五 二叉树的基本操作

**一、实验目的：**

1.熟练掌握二叉树的二叉链表存储结构。

2.掌握二叉树的非线性和递归性特点。

3.熟练掌握二叉树的递归遍历操作的实现方法，掌握二叉树的非递归遍历操作的实现。

4.加深对二叉树结构和性质的理解。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现二叉树的基本操作算法。

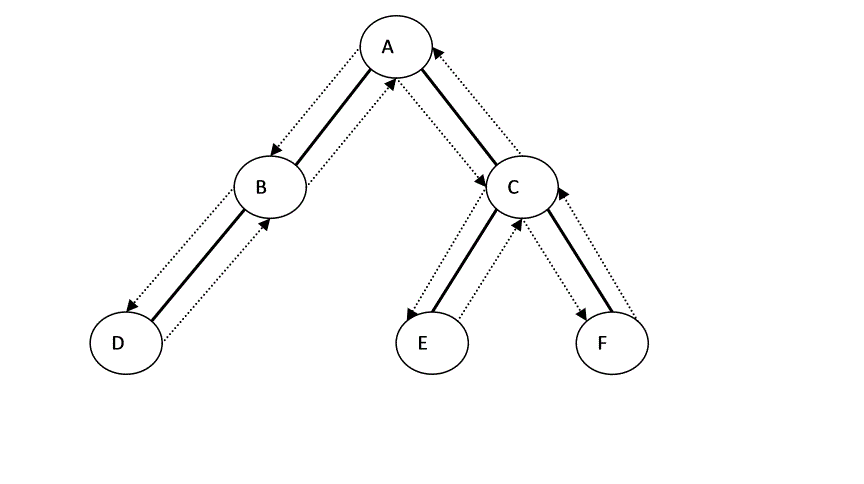
2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

具体要求如下：



二叉树示意图

采用二叉树链表作为存储结构，设计一棵二叉树，输入完全二叉树的先序序列，用 # 代表虚结点（空指针），如 ABD###CE##F## ，建立二叉树，求出先序、中序和后序遍历序列，再求出所有叶子及结点总数。

**五、实施步骤与技术要点**

1.二叉树的链式存储结构定义

typedef char ElemType;

typedef struct BinTreeNode

{ ElemType data;

struct BinTreeNode \*lchild;

struct BinTreeNode \*rchild;

}BinTreeNode,\*Tree;

2.二叉树的建立算法

void CreateTree(Tree \*t)

{

ElemType ch;

scanf("%c",&ch);

if(ch!='#')

{

\*t=(Tree)malloc(sizeof(BinTreeNode));

(\*t)->data=ch;

CreateTree(&(\*t)->lchild);

CreateTree(&(\*t)->rchild);

}

else

\*t=NULL;

}

3.二叉树的先序遍历算法

void DLRTraverse(BinTreeNode \*t) //先序遍历

{

if(t)

{

printf("%c",t->data);

DLRTraverse(t->lchild);

DLRTraverse(t->rchild);

}

}

4.二叉树的中序遍历算法

void LDRTraverse(BinTreeNode \*t) //中序遍历

{

if(t)

{

LDRTraverse(t->lchild);

printf("%c",t->data);

LDRTraverse(t->rchild);

}

}

5.二叉树的后序遍历算法

void LRDTraverse(BinTreeNode \*t) //后序遍历

{

if(t)

{

LRDTraverse(t->lchild);

LRDTraverse(t->rchild);

printf("%c",t->data);

}

}

6.二叉树的叶子结点计算算法

int countLeaves(BinTreeNode \*root) {

if (root == NULL) return 0;

if (root->lchild == NULL && root->rchild == NULL) return 1;

return countLeaves(root->lchild) + countLeaves(root->rchild);

}

7.二叉树的所有结点计算算法

int countNodes(BinTreeNode \*root) {

if (root == NULL) return 0;

return 1 + countNodes(root->lchild) + countNodes(root->rchild);

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验六 图的构造与遍历

**一、实验目的：**

1.了解图的基本概念以及逻辑结构。

2.理解图的两种存储结构（邻接矩阵、邻接表）。

3.掌握图的两种遍历方法（深度优先搜索和广度优先搜索）。

4.掌握图的应用和经典算法，如最小生成树等。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现图的构造、遍历等算法。

2.演示程序运行结果。

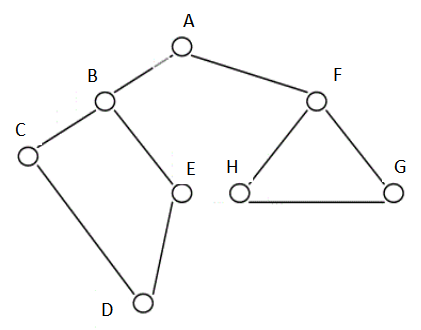
3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

具体要求：构造无向连通图，采用邻接矩阵作为图的存储结构，输出邻接矩阵，完成图的 DFS （深度优先遍历）和 BFS （广度优先遍历）的操作。

实验无向图 G如图：



**五、实施步骤与技术要点**

1.图的结构定义（这里使用数组表示法）

typedef char VextexType;

typedef int EdgeType;

#define MaxNum 15 //邻接矩阵的最大顶点数

typedef struct adjacency\_matrix

{

VextexType MGVexTexArray[MaxNum]; //顶点表

EdgeType Edge[MaxNum][MaxNum];

int VertexNum,EdgeNum; //顶点数和边数

}MGraph;

2. 邻接矩阵的创建

void Creat\_Adjacency\_Matrix(MGraph \*MG)

{

int i,j,x,y;

//先输入顶点数和边数

printf("输入邻接矩阵的顶点数和边数:");

scanf("%d %d",&MG->VertexNum,&MG->EdgeNum);

//初始化顶点表

printf("输入邻接矩阵各个顶点的信息:");

for(i=0; i<MG->VertexNum; i++)

{

getchar();

scanf("%c",&MG->MGVexTexArray[i]);

}

//初始化边表

for(x=0; x<MG->VertexNum; x++)

{

for(y=0; y<MG->VertexNum; y++)

{

MG->Edge[x][y] = 0; //初始化边表的各项为0，表示两者还没有直达的边

}

}

//开始构造邻接矩阵

for(j=0; j<MG->EdgeNum; j++)

{

printf("请输入边两个顶点的下标:");

scanf("%d %d",&x,&y);

MG->Edge[x][y] = 1;

//由于无向网和无向图是对称的所以

MG->Edge[y][x] = MG->Edge[x][y];

}

printf("\n临接矩阵输出如下：\n");

for(x=0; x<MG->VertexNum; x++)

{

for(y=0; y<MG->VertexNum; y++)

{

printf("%d ", MG->Edge[x][y] );

}

printf("\n");

}

}

3.深度优先遍历

void DFS\_MG(MGraph \*MG,int i)

{

int k;

visited[i] = 1;

printf("%c ",MG->MGVexTexArray[i]);

for(k=0; k<MG->VertexNum; k++)

{

if(MG->Edge[i][k]==1 && !visited[k])

{

DFS\_MG(MG, k);

}

}

}

4.广度优先遍历

void BFS\_Traverse\_MG(MGraph \*MG)

{

int i,j;

Queue q;

CreatQueue(&q);

//初始化标志数组

for(i=0; i<MG->VertexNum; i++)

visited[i] = 0;

//开始构建广度优先算法遍历

for(i=0; i<MG->VertexNum; i++)//若是连通图只执行一次即可遍历完

{

if(!visited[i])

{

visited[i] = 1;

EnQueue(&q, i);

while(!Queue\_isEmpty(&q))

{

i = DeQueue(&q);

printf("%c ",MG->MGVexTexArray[i]);

for(j=0; j<MG->VertexNum; j++)

{

if(!visited[j] && MG->Edge[i][j]==1)

{

visited[j] = 1;

EnQueue(&q, j);

}

}

}

}

}

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验七 查找算法

**一、实验目的：**

1.理解查找的基本概念。

2.重点掌握线性表的查找。

3.了解树表的查找。

4.了解散列(Hash)技术。

5.逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现顺序查找、二分查找算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

具体要求：

1.从键盘输入11个整数；5，16，20，27，30，36，44，55，60，67，71。

2.编程实现顺序查找，输入27和65，验证是否找到。

3.编程实现二分查找，输入27和65，验证是否找到。

4.如果查找成功，需要给出查找成功和比较次数的信息，并确定元素所在的位置。

5.如果查找失败，需要给出查找失败和比较次数的信息。

**五、实施步骤与技术要点**

1.顺序表结构体定义

typedef int KeyType;

typedef struct

{

KeyType key;

}ElemType;

typedef struct //顺序表

{

ElemType \*R;

int length;

}SSTable;

2. 顺序查找

int Search\_Seq(SSTable ST,int key) //顺序查找

{

int i;

ST.R[0].key = key; //将待查找的数值放在0号位置

for(i = ST.length;ST.R[i].key!=ST.R[0].key;i--); //依次与标志各元素进行比较

return i; //返回i，即位置

}

3.二分查找

int BinarySearch(SSTable ST,int key) //二分查找

{

int low=0,high=ST.length-1,mid;

while(low<=high)

{

mid=(low+high)/2;

if(ST.R[mid].key ==key)

return mid;

if(ST.R[mid].key>key)

high=mid-1;

else

low=mid+1;

}

return 0;

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。

实验八 排序算法

**一、实验目的：**

1.理解排序的基本概念。

2.理解各种排序算法的基本思想及特点。

3.掌握各种排序算法的程序设计方法。

4.熟练掌握常用排序算法的实现。

5.加深对查找和排序的理解，逐步培养解决实际问题的编程能力。

**二、实验设备**

安装有能够编译运行C或C++语言的电脑。

**三、实验要求**

1.编写程序，实现直接插入排序、冒泡排序、快速排序算法。

2.演示程序运行结果。

3.分析调试过程中出现的现象。

4.完成实验报告。

**四、实验内容**

具体要求：

1.从键盘输入关键字序列k={ 49，38，65，97，76，13，27，49 }。

2.编程实现直接插入排序、冒泡排序、简单选择排序和快速排序。

3.输出各种排序算法每一趟排序的结果，观察关键字次序的变化。

**五、实施步骤与技术要点**

1. 直接插入排序代码

void InsertSort(SeqList \*sqList,int n) //插入排序

{ int i,j,t;

printf("插入排序过程\n");

for (i = 1; i < n; i++)

{

if (sqList[i] < sqList[i - 1])

{ t = sqList[i];

j = 0;

for (j = i - 1; j >= 0 && t < sqList[j]; j--)

sqList[j + 1] = sqList[j];

sqList[j + 1] = t;

}

printf("第%d趟排序过程：",i);

printSeqList(sqList,n);

printf("\n");

}

}

2.冒泡排序代码

void BubbleSort(SeqList \*sqList,int n) //冒泡排序

{ int i,j,t;

printf("\n冒泡排序过程\n");

for (i = 0; i < n; i++)

{ for (j = 0; j < n-i-1; j++)

if (sqList[j + 1] < sqList[j])

{

t = sqList[j + 1];

sqList[j + 1] = sqList[j];

sqList[j] = t;

}

printf("第%d趟排序过程：",i+1);

printSeqList(sqList,n);

printf("\n");

}

}

3.快速排序代码

int partitions(SeqList \*sqList,int low,int high,int n)

{ int pivotkey=sqList[low];

while(low<high)

{ while(low<high && sqList[high]>=pivotkey)

high--;

sqList[low]=sqList[high];

while(low<high && sqList[low]<=pivotkey)

low++;

sqList[high]=sqList[low];

}

sqList[low]=pivotkey;

printSeqList(sqList,n);

return low;

}

void qSort(SeqList \*sqList,int low,int high,int n)//快速排序

{ int pivottag;

static int i=1;

if(low<high)

{ printf("第%d次快速排序：",i++);

//递归调用

pivottag=partitions(sqList,low,high,n);

printf("\n");

qSort(sqList,low,pivottag-1,n);

qSort(sqList,pivottag+1,high,n);

}

}

六、考核或评价标准

实验成绩将主要根据学生对待实验的态度、对关键知识点和编程技巧的掌握程度、实验报告的内容、答辩情况等进行综合评定。最后的成绩将分优秀、良好、中等、及格和不及格五个等级。具体评判标准如下：

优秀：实验认真、刻苦，有钻研精神，不无故缺席。熟练掌握了本实验的关键知识点，具有良好的独立思考问题和解决问题的能力，具备了较好的C语言编程能力，编制的程序运行正确。实验记录内容丰富、齐全，答辩时能清晰明了地阐明问题，回答问题反映敏捷、思路清晰。

良好：能认真对待实验，不无故缺席。掌握了本实验的关键知识点，具备了较好的C语言编程能力，编写的程序运行正确。实验记录内容齐全，答辩时能清晰明了地阐明问题，能正确回答全部问题。

中等：能认真对待实验，不无故缺席。基本掌握了本实验的关键知识点，具备了一定的C语言编程能力，编写的程序运行基本正确，无致命错误。实验记录内容较齐全，答辩时能正确回答大部分问题。

及格：对待实验不够认真，有少量迟到、早退或无故缺席现象。基本掌握了本实验的主要内容，具有了用C语言编程的基本能力，但掌握不全面、扎实，编写的程序总体结构符合要求，基本能正常运行，但还存在少量错误。实验记录内容基本齐全，答辩时能在教师提示下正确回答大部分问题。

不及格：对待实验马虎、敷衍，经常迟到、早退或无故缺席。不能正确理解本实验的主要内容，不具备基本的C语言编程能力，编制的程序不能正常运行，或是抄袭他人程序，应付答辩。答辩时即使经教师提示仍不能正确回答大部分问题。