**实验五 查找和排序算法的设计**

**位序号（两位）：55 班级：创新2201 组号：11 姓名： 孙文杰**

1. **实验目的**

1、熟悉静态查找的实现过程和基本思想，掌握各种查找算法的实现。

2、熟悉各种排序的实现过程和基本思想

3、掌握直接插入排序的算法的实现。

1. **实验内容**

（1）基础实验：查找和排序基础应用

实验内容

（1）建立顺序表，输入n个整数，实现顺序查找，输出查找结果及该整数在表中位置号。

（2）建立顺序表，输入n个整数，采用直接插入排序从小到大排序后输出，并应用折半查找方法查给定值，输出查找结果及该整数在表中位置号。

1. **算法设计描述**

**顺序查找：**

- 定义一个顺序表结构，包含一个数组和一个长度属性。

- 定义一个初始化顺序表的函数，根据输入的n个整数，为数组分配空间，并将整数依次存入数组，同时更新长度属性。

- 定义一个顺序查找的函数，接受一个顺序表和一个待查找的整数作为参数，返回一个整数表示查找结果。

- 在顺序查找函数中，从数组的第一个元素开始，依次与待查找的整数比较，如果相等，则返回当前的下标，表示查找成功；如果不相等，则继续比较下一个元素，直到数组的最后一个元素为止。

- 如果遍历完数组后仍未找到相等的元素，则返回-1，表示查找失败。

- 在主函数中，调用初始化顺序表的函数，输入n个整数，建立顺序表。

- 然后，输入一个待查找的整数，调用顺序查找的函数，得到查找结果。

- 如果查找结果不等于-1，则输出“查找成功，该整数在表中的位置号为：”和查找结果；否则，输出“查找失败，该整数不在表中。”

**插入排序+折半查找**

- 定义一个顺序表结构，包含一个数组和一个长度属性。

- 定义一个初始化顺序表的函数，根据输入的n个整数，为数组分配空间，并将整数依次存入数组，同时更新长度属性。

- 定义一个直接插入排序的函数，接受一个顺序表作为参数，对其进行从小到大的排序。

- 在直接插入排序函数中，从数组的第二个元素开始，依次将每个元素插入到前面已经有序的子数组中，使得子数组仍然有序，直到数组的最后一个元素为止。

- 在插入过程中，先将当前元素暂存到一个变量中，然后从后往前比较已经有序的子数组中的元素，如果比当前元素大，则将其后移一位，为当前元素腾出空间；如果比当前元素小或等于，则停止比较，将当前元素插入到空位中。

- 定义一个折半查找的函数，接受一个顺序表和一个待查找的整数作为参数，返回一个整数表示查找结果。

- 在折半查找函数中，设置两个变量，分别表示查找区间的左右边界，初始时为数组的第一个和最后一个元素的下标。

- 在查找区间不为空时，计算查找区间的中间位置，将待查找的整数与中间位置的元素比较，如果相等，则返回中间位置的下标，表示查找成功；如果不相等，则根据比较结果，缩小查找区间，如果待查找的整数小于中间位置的元素，则将右边界设为中间位置减一；如果待查找的整数大于中间位置的元素，则将左边界设为中间位置加一。

- 如果查找区间为空时仍未找到相等的元素，则返回-1，表示查找失败。

- 在主函数中，调用初始化顺序表的函数，输入n个整数，建立顺序表。

- 然后，调用直接插入排序的函数，对顺序表进行从小到大的排序，并输出排序后的顺序表。

- 接着，输入一个待查找的整数，调用折半查找的函数，得到查找结果。

- 如果查找结果不等于-1，则输出“查找成功，该整数在表中的位置号为：”和查找结果；否则，输出“查找失败，该整数不在表中。”

1. **程序清单**

**顺序查找.c**

*#include <stdlib.h>*

*#include <stdio.h>*

*#include <math.h>*

*#include <string.h>*

*#include <stdbool.h>*

*#define MaxSize 110*

*typedef struct SeqList {*

*int \*data;*

*int length, maxSize;*

*}SeqList;*

*bool initSeqList(SeqList \*L) {*

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

*if(L->data == NULL) return false;*

*L->length = 0;*

*L->maxSize = MaxSize;*

*return true;*

*}*

*bool insertSeqList(SeqList \*L, int e) {*

*if(L->length + 1 == L->maxSize) {*

*printf("The SeqList is full!\n");*

*return false;*

*}*

*L->data[L->length++] = e;*

*return true;*

*}*

*bool querySeqList(SeqList \*L, int value, int \*pos) {*

*for(int i = 0; i < L->length; ++ i) {*

*if(L->data[i] == value) {*

*\*pos = i + 1;*

*return true;*

*}*

*}*

*return false;*

*}*

*int main() {*

*SeqList L;*

*initSeqList(&L);*

*printf("Please enter n:\n");*

*int n;*

*scanf("%d", &n);*

*printf("Please enter n(1 <= n <= 100) integers:\n");*

*for(int i = 1; i <= n; ++ i) {*

*int e;*

*scanf("%d", &e);*

*insertSeqList(&L, e);*

*}*

*printf("Please enter the integer you want to query:\n");*

*int ans, pos; scanf("%d", &ans);*

*bool flag = querySeqList(&L, ans, &pos);*

*if(!flag) {*

*printf("The integer you want to query does not exist.\n");*

*} else {*

*printf("The position of the integer you want to query is the %dth\n", pos);*

*}*

*return 0;*

*}*

**直接插入排序+折半查找.c**

*#include <stdlib.h>*

*#include <stdio.h>*

*#include <math.h>*

*#include <string.h>*

*#include <stdbool.h>*

*#define MaxSize 110*

*typedef struct SeqList {*

*int \*data;*

*int length, maxSize;*

*}SeqList;*

*bool initSeqList(SeqList \*L) {*

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

*if(L->data == NULL) return false;*

*L->length = 0;*

*L->maxSize = MaxSize;*

*return true;*

*}*

*bool insertSeqList(SeqList \*L, int e) {*

*if(L->length + 1 == L->maxSize) {*

*printf("The SeqList is full!\n");*

*return false;*

*}*

*L->data[L->length++] = e;*

*return true;*

*}*

*void insertSortSeqList(SeqList \*L) {*

*for(int i = 0; i < L->length; ++ i) {*

*int j = i - 1;*

*for(; j >= 0; -- j)*

*if(L->data[j] < L->data[i]) break;*

*if(j != i - 1) {*

*int temp = L->data[i];*

*for(int k = i; k > j + 1; -- k)*

*L->data[k] = L->data[k-1];*

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

*}*

*}*

*}*

*void printSeqList(SeqList \*L) {*

*for(int i = 0; i < L->length; ++ i) {*

*printf("%d ", L->data[i]);*

*}*

*printf("\n");*

*}*

*bool querySeqList(SeqList \*L, int value, int \*pos) {*

*int l = 0, r = L->length - 1;*

*while(l <= r) {*

*int mid = (l + r) >> 1;*

*if(L->data[mid] == value) {*

*\*pos = mid + 1;*

*return true;*

*} else if(L->data[mid] < value) {*

*l = mid + 1;*

*} else {*

*r = mid - 1;*

*}*

*}*

*return false;*

*}*

*int main() {*

*SeqList L;*

*initSeqList(&L);*

*printf("Please enter n:\n");*

*int n;*

*scanf("%d", &n);*

*printf("Please enter n(1 <= n <= 100) integers:\n");*

*for(int i = 1; i <= n; ++ i) {*

*int e;*

*scanf("%d", &e);*

*insertSeqList(&L, e);*

*}*

*insertSortSeqList(&L);*

*printf("The sorted SeqList:\n");*

*printSeqList(&L);*

*printf("Please enter the integer you want to query:\n");*

*int ans, pos; scanf("%d", &ans);*

*bool flag = querySeqList(&L, ans, &pos);*

*if(!flag) {*

*printf("The integer you want to query does not exist.\n");*

*} else {*

*printf("The position of the integer you want to query is the %dth\n", pos);*

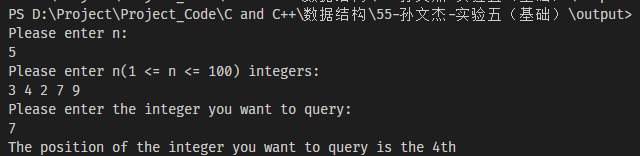
*}*

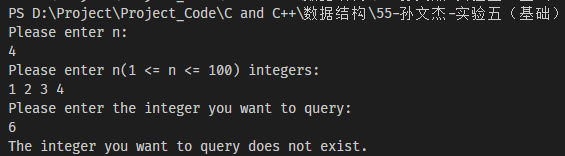
*return 0;*

*}*

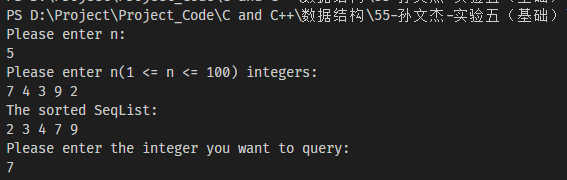
1. **实验结果及分析**

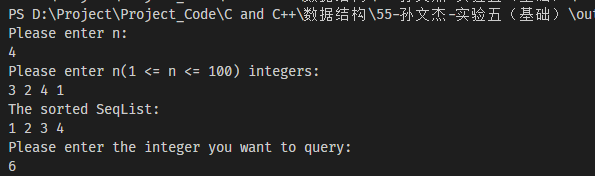
**顺序查找**





**直接插入排序+折半查找**





**六．总结**

**我熟悉了静态查找的实现过程和基本思想，也掌握直接插入排序的算法的实现。我对排序和查找有了新的认识。**