# 

# 《计算机实践》

# 实验报告I—数据结构

班号：0321501 学号：032150115

姓名：易博天

Email：yiboxiaotian@nuaa.edu.cn

签名：

南京航空航天大学

## 实验：查找和排序实验

### 一、实验目的

　　掌握顺序查找和二分查找的操作实现；掌握插入排序、选择排序和冒泡排序的相关操

作实现。

### 二、实验内容

### 实现查找表的顺序查找和二分查找；实现插入排序、选择排序和冒泡排序。三、算法分析与程序设计

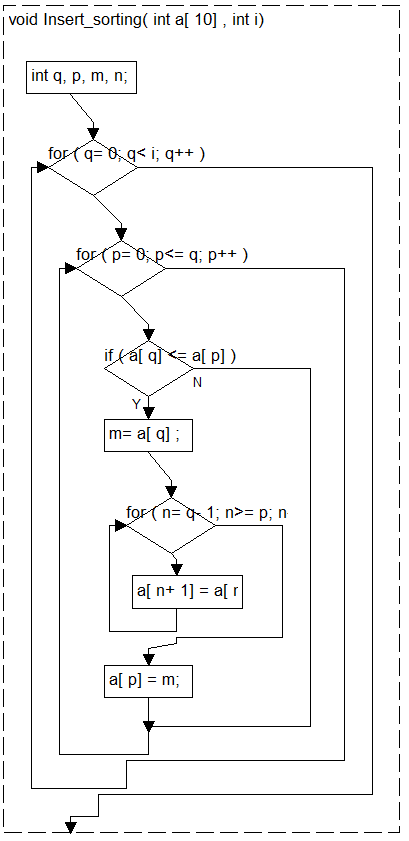
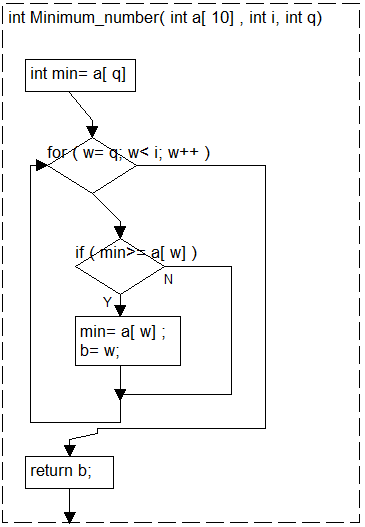
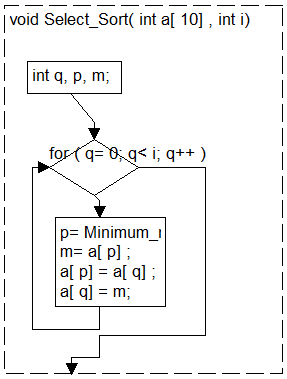
1.顺序查找利用顺序表进行，二分查找利用顺序存储的有序表进行，设计算法并编程实

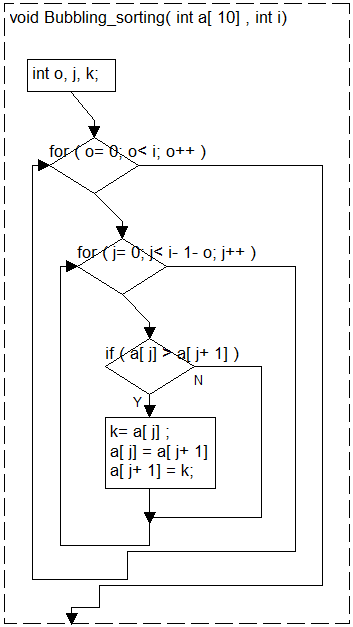
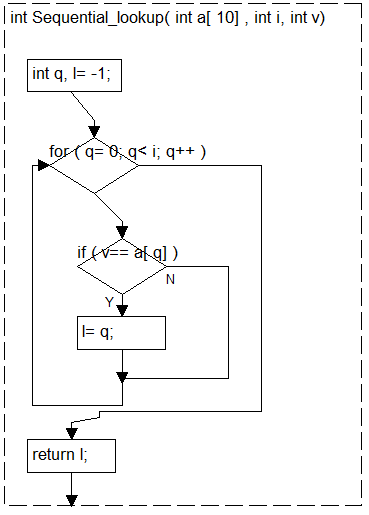
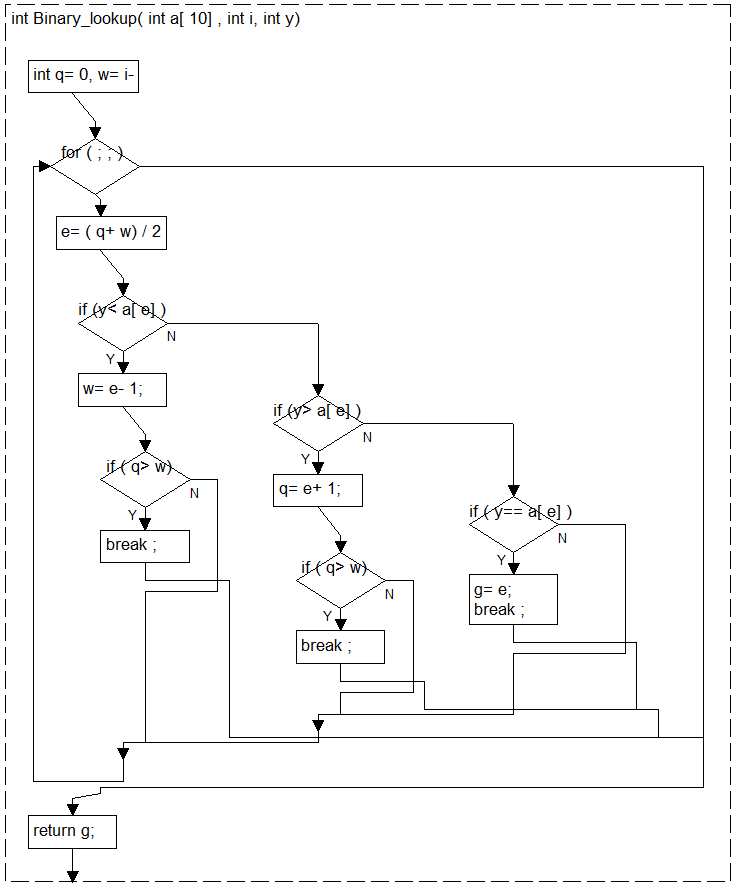
现此过程。设计菜单，选择排序方法对查找表的数据进行排序。实现 3 种排序算法时，可以使用随机数，或者键盘输入数据。

2. 输入数据：查找操作时，建立输入处理，输入查找表里的数据；待查找的关键字要求从键盘输入。排序操作时，可随机生成待排序列，或者键盘输入待排序列，数据元素 10 个以内。

输出形式：分别输出查找表元素以及查找结果（找到的话返回关键字的位置，没找到

返回相应的查找失败信息）。对排序操作，输出原始数据序列和每一趟排序序列。3.算法描述

给出算法描述的程序流程图

4. 程序设计

给出实现程序功能的函数及说明如下：

(1) 本程序包含的函数

主函数 main()

void Insert\_sorting(int a[10], int i); // 插入排序

int Minimum\_number(int a[10], int i, int q); // 最小数

void Select\_Sort(int a[10], int i); // 选择排序

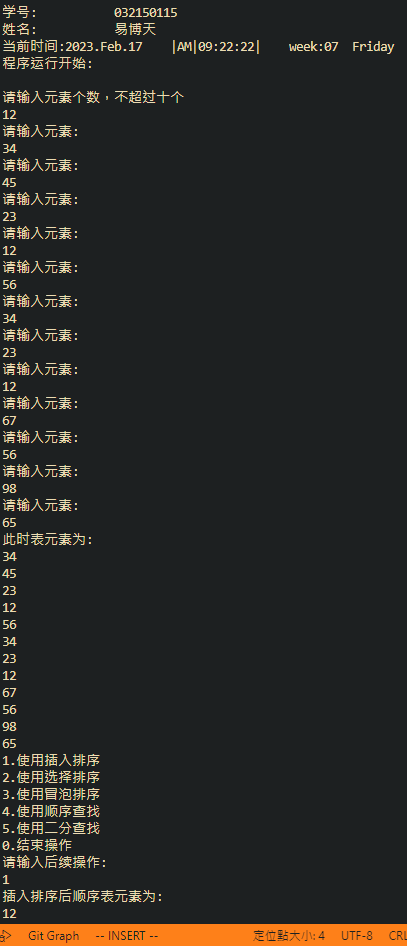
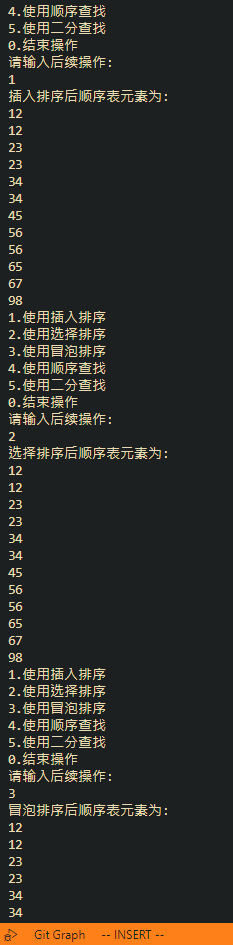
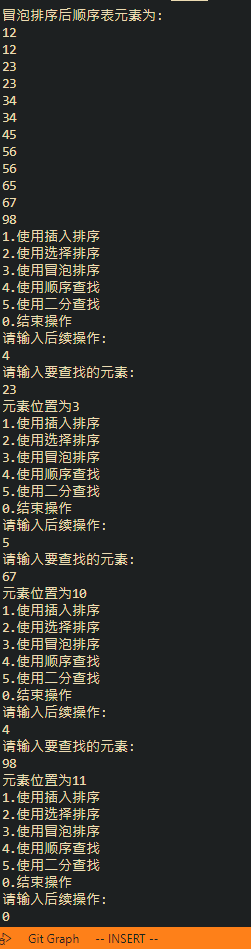
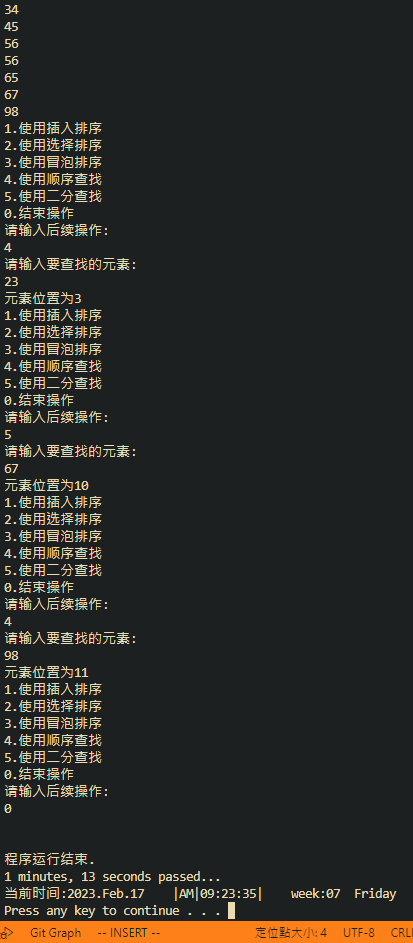
void Bubbling\_sorting(int a[10], int i); // 冒泡排序

int Sequential\_lookup(int a[10], int i, int v); // 顺序查找

int Binary\_lookup(int a[10], int i, int y); // 二分查找

### 四、调试分析

#### 1、调试过程及说明

* 以截屏图像的方式给出，包括程序运行开始和结束时的系统时间、姓名、学号、实验名称等信息.

#### 2、调试分析

照着书上写的，没什么问题。也可以调整为随机数给出的排序。

### 五、使用说明

按照提示运行即可

**六、测试结果**

内容：给出测试用例测试程序。写出测试的情况，使用的输入以及得到的结果。

要求：测试的情况越全面越好。

示例：

测试例子如上图所示

**七、实验的收获和感想**

深刻理解了几种排序和查找操作的巧妙，比对了它们之间的异同，体会了其中深刻的思想

**八、附录：源代码（双栏排版，加必要的注释）**

#include "homework.h"

void Insert\_sorting(int a[10], int i); // 插入排序

int Minimum\_number(int a[10], int i, int q); // 最小数

void Select\_Sort(int a[10], int i); // 选择排序

void Bubbling\_sorting(int a[10], int i); // 冒泡排序

int Sequential\_lookup(int a[10], int i, int v); // 顺序查找

int Binary\_lookup(int a[10], int i, int y); // 二分查找

void Insert\_sorting(int a[10], int i)

{

int q, p, m, n;

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

{

for (p = 0; p <= q; p++)

{

if (a[q] <= a[p])

{

m = a[q];

for (n = q - 1; n >= p; n--)

{

a[n + 1] = a[n];

}

a[p] = m;

}

}

}

}

int Minimum\_number(int a[10], int i, int q)

{

int min = a[q], w, b;

for (w = q; w < i; w++)

{

if (min >= a[w])

{

min = a[w];

b = w;

}

}

return b;

}

void Select\_Sort(int a[10], int i)

{

int q, p, m;

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

{

p = Minimum\_number(a, i, q);

m = a[p];

a[p] = a[q];

a[q] = m;

}

}

void Bubbling\_sorting(int a[10], int i)

{

int o, j, k;

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

{

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

{

if (a[j] > a[j + 1])

{

k = a[j];

a[j] = a[j + 1];

a[j + 1] = k;

}

}

}

}

int Sequential\_lookup(int a[10], int i, int v)

{

int q, l = -1;

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

{

if (v == a[q])

{

l = q;

}

}

return l;

}

int Binary\_lookup(int a[10], int i, int y)

{

int q = 0, w = i - 1, e, g = -1;

for (;;)

{

e = (q + w) / 2;

if (y < a[e])

{

w = e - 1;

if (q > w)

{

break;

}

}

else if (y > a[e])

{

q = e + 1;

if (q > w)

{

break;

}

}

else if (y == a[e])

{

g = e;

break;

}

}

return g;

}

int main()

{

homework\_start();

int a[10];

int i, j;

printf("请输入元素个数，不超过十个\n");

scanf("%d", &i);

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

{

printf("请输入元素:\n");

scanf("%d", &a[j]);

}

int s;

printf("此时表元素为:\n");

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

{

printf("%d\n", a[s]);

}

int k;

for (;;)

{

printf("1.使用插入排序\n");

printf("2.使用选择排序\n");

printf("3.使用冒泡排序\n");

printf("4.使用顺序查找\n");

printf("5.使用二分查找\n");

printf("0.结束操作\n");

printf("请输入后续操作:\n");

scanf("%d", &k);

if (k == 1)

{

Insert\_sorting(a, i);

int z;

printf("插入排序后顺序表元素为:\n");

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

{

printf("%d\n", a[z]);

}

}

if (k == 2)

{

Select\_Sort(a, i);

int z;

printf("选择排序后顺序表元素为:\n");

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

{

printf("%d\n", a[z]);

}

}

if (k == 3)

{

Bubbling\_sorting(a, i);

int z;

printf("冒泡排序后顺序表元素为:\n");

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

{

printf("%d\n", a[z]);

}

}

if (k == 4)

{

printf("请输入要查找的元素:\n");

int v, x;

scanf("%d", &v);

x = Sequential\_lookup(a, i, v);

if (x == -1)

{

printf("查找元素失败,无此元素\n");

}

else

{

printf("元素位置为%d\n", x);

}

}

if (k == 5)

{

printf("请输入要查找的元素:\n");

int y, u;

scanf("%d", &y);

u = Binary\_lookup(a, i, y);

if (u == -1)

{

printf("查找元素失败,无此元素\n");

}

else

{

printf("元素位置为%d\n", u);

}

}

if (k == 0)

{

break;

}

}

homework\_done();

return 0;

}