《数据结构与算法实验》第 2 次实验

学院： 专业 年级：
姓名： 学号： 日期： 2022 年 2 月 28 日

一、 实验目的

1.学习用时间函数测试算法的执行时间。

2.学习随机数生成的方法生成数据。

3.复习、学习模板函数的编写。

4.完成作业中的六个题目。

二、 实验内容

1. **排序算法与时间测试：**对课本 p19 页习题 6 编写程序，并测试算法的执行时间。
   - 对一个整形数组 A[n] 设计一个排序算法。
   - 找出整形数组 A[n] 中的最大值和次最大值。
2. **排序算法的模板实现：**分模板实现课本 p19 页习题 6 的排序算法。
3. **数组的循环移动：**课本 p53 页习题 5(1),先做整数，然后改用模板函数做并测试。
   - 设计时间复杂度为 O(n) 的算法，实现将数组 A[n] 中所有元素循环左移 k 个位置。
4. **对奇偶数分类：**课本 p53 页习题 5(2)。

   •已知数组 A[n] 的元素为整型，设计算法将其调整为左右两部分，左边所有元素为奇数，右边所有元素为偶数，并要求算法的时间复杂度为 O(n)。
5. **验证实验-验证线性表和模板类：**实验书 P171 顺序表的实现。
   - 加入求线性表的长度等操作： 结合本章习题的算法设计题做。
   - 重新给定测试数据，验证抛出异常机制。
6. **顺序表的逆置：**设计课本 p53 页习题 5(4)的程序，完成顺序表的逆序。
   - 试分别以顺序表作存储结构，实现线性表就地逆置的算法。

三、 设计与编码

1. 本实验用到的理论知识

• 选择排序算法：将整个序列划分为有序区和无序区，初始时有序区为空，无序区含有待排序的所有元素；在无序区中选取最小记录，将它与无序区的第一个记录交换，使得有序区扩展了一个记录，同时无序区减少了一个记录；不断重复以上步骤，直到无序区只剩下一个记录为止，此时所有记录已经按关键码从小到大的顺序排列。

• 计时算法：记录开始时刻与结束时刻，程序所用时长就是结束时刻减去开始时刻。

• 函数模板：建立一个通用函数，其函数类型和形参类型不具体指定，用一个虚拟的类型来代表。

• 顺序表逆置算法：利用数组逆置法实现数组向左边移动 p 位。分别令 i 指向顺序表的表头，j 指向表尾，交换 i，j 位置上的两个数，然后利用 i++, j--, 直到 $i \geq j$。

• 数组循环移动算法：利用数组逆置实现数组整体向左移动 p 位。将数组分为 A、B 两个部分，A 表示前 p 个元素，B 表示剩下的元素；将 A、B 分别逆置得到 A'、B'，再对(A'B') 整体进行逆置，就得到(A'B')'=BA。

• 数组奇偶调整算法：从数组的两端向中间比较，只对需要调整的数字进行交换和插入。

2. 算法设计

（1） **排序算法与时间测试：**用 srand 函数生成随机数。先随机生成 10 个在 1~10000 之间的数，再用选择排序的方法对其从小到大排序并输出结果，在结果序列中可以得到最大数和次大数。再新建一个程序算法用来测试排序时间，通过改变数组的总数测试若干次排序时间，取单次平均值即可。

（2） **排序算法的模板实现：**定义一个模板函数，在随机数取不同类型的情况下采用相同的函数体进行排序。

（3） **数组的循环移动：**使用三-1 中提到的数组循环移动算法，先令移动的位数 k 对数组总长度求余数，降低计算难度，再使用三次逆序算法即可实现数组的移动。

（4） **对奇偶数分类：**使用三-1 中提到的数组奇偶调整算法，初始化两个变量 i=1,j=n，若第 i 位数字是奇数，则 i++，直到第 i 位数字为偶数；若第 j 位数字是偶数，则 j--，直到第 j 位数字为奇数。如果 i<j，则交换 ij 所代表数的位置，重复以上步骤，如果 i≥j，则算法结束。

（5） **顺序表的逆置：**使用三-1 中提到的顺序表逆置算法，再建立模板函数实现顺序表逆置即可。

3. 编码

(1)排序算法与时间测试-sort.cpp

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
using namespace std;
const int n=10;
void sort(int A[])
{
    int i,j,k;
    int t;
    for ( i = 1; i <= n; i++)
    {
        k=i;
        for ( j = i+1; j <= n; j++)
            if (A[j]<A[k]) k=j;
        t=A[k];
        A[k]=A[i];
        A[i]=t;
    }
}
void show(int A[])
{
    for (int i = 1; i <= n; i++)
        cout<<A[i]<<" ";
    cout<<endl;
}
void get(int A[])
{
    srand(time(NULL));
    for (int i = 1; i <= n; i++)
        A[i]=1+rand()%10000;
}
int main()
{
    int A[n+1];
    get(A);
    sort(A);
    show(A);
    cout<<"max="<<A[n]<<",vice-max="<<A[n-1]<<endl;
    return 0;
}
```

(1) 排序算法与时间测试-sort_time.cpp（算法执行时间）

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#define CLOCKS_PER_SEC 1000
using namespace std;
const int n=10000;
void sort(int A[])
```

```
{
    int i,j,k;
    int t;
    for(i=1;i<=n;i++)
    {
        k=i;
        for(j=i+1;j<=n;j++)
            if(A[j]<A[k]) k=j;
        t=A[k];
        A[k]=A[i];
        A[i]=t;
    }
}
void show(int A[])
{
    for(int i=1;i<=n;i++)
        cout<<A[i]<<" ";
    cout<<endl;
}
void get(int A[])
{
    srand(time(NULL));
    for(int i=1;i<=n;i++)
        A[i]=1+rand()%10000;
}
int main( )
{
    int A[n+1];
    clock_t start_t, end_t;
    double total_t=0;
    int k=10;
    for(int j=1;j<=10;k=k+10,j++)
    {
        for(int i=1;i<=k;i++)
        {
            get(A);
            start_t = clock();  //开始时间
            sort(A);
            end_t = clock();    //结束时间
            total_t=total_t-start_t+end_t;
        }
    total_t = (double) total_t / CLOCKS_PER_SEC;    //运行时间
    cout<<"对"<<k<<"组数排序，每组数有"<<n<<"个，所需时间为："<<total_t<<endl;
    }
    return 0;
}
```

(2) 排序算法的模板实现-sort_template.cpp

```
#include <iostream>
using namespace std;
const int n=10;
template<typename T>
void sort(T A[])
{
```

```
    int i,j,k;
    T t;
    for ( i = 1; i <=n; i++)
    {
        k=i;
        for ( j = i+1; j <= n; j++)
            if (A[j]<A[k]) k=j;
        t=A[k];
        A[k]=A[i];
        A[i]=t;
    }
}
template<typename T>
void show(T A[])
{
    for (int i = 1; i <= n; i++)
        cout<<A[i]<<" ";
    cout<<endl;
    cout<<"max="<<A[n]<<",vice-max="<<A[n-1]<<endl;
}
int main()
{
    int A1[n+1];
    double A2[n+1];
    long A3[n+1];
    cout<<"type in:"<<endl;

    for (int i = 1; i <=n ; i++)
        cin>>A1[i];
    for (int i = 1; i <=n ; i++)
        cin>>A2[i];
    for (int i = 1; i <=n ; i++)
        cin>>A3[i];
    cout<<"output:"<<endl;
    sort(A1);
    show(A1);
    sort(A2);
    show(A2);
    sort(A3);
    show(A3);
    return 0;
}
```

(3)数组的循环移动（整数）-int.cpp

```
#include <iostream>
#include <cstdio>
using namespace std;
const int n = 10;
void inverse(int A[],int sta,int end)
```

```
{
    int i=sta,j=end;
    while (i<j)
    {
        swap(A[i],A[j]);
        i++;
        j--;
    }
}
void move(int A[],int k)
{
    inverse(A,1,k);
    inverse(A,k+1,n);
    inverse(A,1,n);
}
void show(int A[])
{
    for (int i = 1; i <= n; i++)
        cout<<A[i]<<" ";
    cout<<endl;
}
void get(int A[])
{
    cout<<"输入10个数字"<<endl;
    for (int i = 1; i <= n; i++)
        cin>>A[i];
}
int main()
{
    int A[n+1],k;
    get(A);
    cout<<"输入k"<<endl;
        cin>>k;
    k=k%n;
    move(A,k);
    show(A);
    return 0;
}
```

(3)数组的循环移动（模板函数）-template.cpp

```
#include <iostream>
#include <cstdio>
using namespace std;
const int n=10;
template<typename T>
void inverse(T A[],int sta,int end)
```

```
{
    int i=sta,j=end;
    while(i<j)
    {
        swap(A[i],A[j]);
        i++;
        j--;
    }
}
template<typename T>
void move(T A[],int k)
{
    inverse(A,1,k);
    inverse(A,k+1,n);
    inverse(A,1,n);
}
template<typename T>
void show(T A[])
{
    for (int i = 1; i <= n; i++)
        cout<<A[i]<<" ";
    cout<<endl;
}
int main()
{
    int A1[n+1];
    double A2[n+1];
    long long A3[n+1];
    int k;
    cout<<"输入数据：  "<<endl;
    for (int i = 1; i <= n; i++)
        cin>>A1[i];
    for (int i = 1; i <= n; i++)
        cin>>A2[i];
    for (int i = 1; i <= n; i++)
        cin>>A3[i];
    cout<<"输入左移位数k"<<endl;
        cin>>k;
    k=k%n;
    move(A1,k);
    move(A2,k);
    move(A3,k);
    show(A1);
    show(A2);
    show(A3);
    return 0;
}
```

(4)对奇偶数分类-sort4.cpp

```
#include <iostream>
#include <cstdio>
using namespace std;
const int n=10;
void inverse(int A[],int sta,int end)//逆向函数
{
    int i=sta,j=end;
    while(i<j)
    {
        swap(A[i],A[j]);
        i++;
        j--;
    }
}
void move(int A[])//移动函数
{
    int i=1,j=n;
    while(i<j)
    {
        while(A[i]%2==1)
            i++;
        while(A[j]%2==0)
            j--;
        if(i<j) swap(A[i],A[j]);
    }
}
void show(int A[])//输出结果
{
    for(int i=1;i<=n;i++)
        cout<<A[i]<<" ";
    cout<<endl;
}
void get(int A[])
{
    cout<<"输入10个数字"<<endl;
    for(int i=1;i<=n;i++)
        cin>>A[i];
}
int main( )
{
    int A[n+1],k;
    get(A);
    move(A);
    show(A);
    return 0;
}
```

(5)验证实验：验证线性表和模板类-SeqList.h（头文件）

```
#ifndef SeqList_H
#define SeqList_H
const int MaxSize=10;
class SeqList
{
public:
   SeqList( ){length=0;}        //无参构造函数，创建一个空表
   SeqList(int a[ ], int n);    //有参构造函数
   ~SeqList( ){ }               //析构函数为空
   void Insert(int i, int x);
  //在线性表中第i个位置插入值为x的元素
   int Delete(int i);           //删除线性表的第i个元素
   int Locate(int x);
          //按值查找，求线性表中值为x的元素序号
   void PrintList( );
          //遍历线性表，按序号依次输出各元素
   int Length( );               //求性线表的长度
   int Get(int i);
             //按位查找，在线性表中查找第i个元素

private:
   int data[MaxSize];           //存放数据元素的数组
   int length;                  //线性表的长度
};
#endif
```

(5)验证实验：验证线性表和模板类-SeqList.cpp

```
#include <iostream>
using namespace std;
#include "SeqList.h"

SeqList::SeqList(int a[],int n)
{
    if (n>MaxSize) throw "参数非法";
    for (int i=0;i<n;i++)
        data[i]=a[i];
    length=n;
}

void SeqList::Insert(int i,int x)
{
    if (length>=MaxSize) throw "上溢";
    if (i<1||i>length+1) throw "位置非法";
    for (int j=length;j>=i;j--)
```

```
        data[j]=data[j-1];
    data[i-1]=x;
    length++;
}

int SeqList::Delete(int i)
{
    if (length==0) throw "下溢";
    if (i<1||i>length+1) throw "位置非法";
    int x=data[i-1];
    for (int j=i;j<length;j++)
        data[j-1]=data[j];
    length--;
    return x;
}

int SeqList::Locate(int x)
{
    for (int i=0;i<length;i++)
        if (data[i]==x) return i+1;
    return 0;
}

void SeqList::PrintList()
{
    for (int i=0;i<length;i++)
        cout<<data[i]<<" ";
    cout<<endl;
}
```

(5)验证实验：验证线性表和模板类- SeqList_main.cpp

```
#include<iostream>
using namespace std;
#include "SeqList.h"

int main()
{
    int r[5]={29,31,3,67,2};
    int a,b;
    SeqList L(r,5);
    cout<<"执行插入操作前数据为："<<endl;
    L.PrintList();
    cout<<"此时数组长度为："<<sizeof(r)/sizeof(r[0])<<endl;
    cout<<"请输入插入的位置："<<endl;
    cin>>a;
    cout<<"请输入插入的数据："<<endl;
    cin>>b;
```

```
    try
    {
        L.Insert(a,b);
    }
    catch(char*s)
    {
        cout<<s<<endl;
    }
    cout<<"执行插入操作后数据为："<<endl;
    L.PrintList();
    cout<<"值为3的元素位置为：";
    cout<<L.Locate(3)<<endl;
    cout<<"执行删除第一个元素操作，删除前数据为："<<endl;
    L.PrintList();
    try
    {
        L.Delete(1);
    }
    catch(char*s)
    {
        cout<<s<<endl;
    }
    cout<<"删除后数据为："<<endl;
    L.PrintList();
    return 0;
}
```

(6)顺序表的逆置- change. cpp

```
#include<cstdio>
#include<cstring>
#include<iostream>
using namespace std;
const int MaxSize=100;
template <class DataType>
class SeqList
{
public:
    SeqList ();
    SeqList (DataType a[],int n);
    ~SeqList(){;}
    int Length(){return length;}
    void Inverse();
    void PrintList();
private:
    DataType data[MaxSize];
    int length;
};
```

```
template <class DataType>
SeqList<DataType>::SeqList()
{
    length=0;
}

template <class DataType>
SeqList<DataType>::SeqList(DataType a[],int n)
{
    if (n>MaxSize) throw "参数非法";
    for(int i=0;i<n;i++)
        data[i]=a[i];
    length=n;
}

template <class DataType>
void SeqList<DataType>::PrintList()
{
    for (int i = 0; i < length; i++)
        cout<<data[i]<<" ";
    cout<<endl;
}

template <class DataType>
void SeqList<DataType>::Inverse()
{
    int i=0,j=length-1;
    DataType t;
    while (i<j)
    {
        t=data[i];
        data[i]=data[j];
        data[j]=t;
        i++;
        j--;
    }
}

int main()
{
    int Ta[1000],Tn;
    cout<<"请输入顺序表的长度（不超过100）："<<endl;
    cin>>Tn;
    cout<<"请逐个输入顺序表中的数字（整数）："<<endl;
    for (int i = 0; i < Tn; i++)
        cin>>Ta[i];
```

```
    SeqList<int> A(Ta,Tn);
    A.Inverse();
    cout<<"逆置后的顺序表为: "<<endl;
    A.PrintList();
    return 0;
}
```

四、 运行与测试

(1)排序算法与时间测试

```
676 765 871 1550 5026 5502 6132 7527 8214 9093
max=9093,vice-max=8214
请按任意键继续. . .
```

随机生成的 10 个数排序结果以及求出的最大数、次最大数如上图。随机生成 k 组数进行排序所用时间如下图所示，根据计算，对 10000 个随机数进行排序，每次所需时间大约为 0.1 秒。

```
对10组数排序, 每组数有10000个, 所需时间为: 1.049
对20组数排序, 每组数有10000个, 所需时间为: 2.16305
对30组数排序, 每组数有10000个, 所需时间为: 3.13816
对40组数排序, 每组数有10000个, 所需时间为: 4.19114
对50组数排序, 每组数有10000个, 所需时间为: 5.16819
对60组数排序, 每组数有10000个, 所需时间为: 6.38317
对70组数排序, 每组数有10000个, 所需时间为: 7.23438
对80组数排序, 每组数有10000个, 所需时间为: 8.17623
对90组数排序, 每组数有10000个, 所需时间为: 9.20318
对100组数排序, 每组数有10000个, 所需时间为: 10.3732

--------------------------------
Process exited after 57.3 seconds with return value 0
请按任意键继续. . .
```

(2)排序算法的模板实现

```
type in:
17 22 93 65 44 32 1 82 99 36
1.2 3.1 5.7 9.3 10.4 7.6 0.5 4.8 6.5 4.4
11111 31245 72834 91236 67293 93527 60384 12043 74832 93156
output:
1 17 22 32 36 44 65 82 93 99
max=99,vice-max=93
0.5 1.2 3.1 4.4 4.8 5.7 6.5 7.6 9.3 10.4
max=10.4,vice-max=9.3
11111 12043 31245 60384 67293 72834 74832 91236 93156 93527
max=93527,vice-max=93156
请按任意键继续. . .
```

如图所示，使用模板函数可以对各种数据类型（整型、双精度型、长整型）的数组进行排序，并且求出最大值和次大值。

(3)数组的循环移动

对整型数组循环移动实验结果如图。可以看出使用模板函数可以对对各种数据类型（整型、双精度型、长整型）的数组进行循环移动。

```
输入10个数字
93 26 55 18 47 82 45 22 80 65
输入k
2
55 18 47 82 45 22 80 65 93 26
请按任意键继续. . .
```

```
输入数据:
29 35 44 96 10 33 57 63 82 73
1.6 3.7 9.8 4.0 2.3 5.5 7.1 5.6 8.2 9.1
11111 56731 28309 55543 27839 10925 37465 99102 83742 30951
输入左移位数k
3
96 10 33 57 63 82 73 29 35 44
4 2.3 5.5 7.1 5.6 8.2 9.1 1.6 3.7 9.8
55543 27839 10925 37465 99102 83742 30951 11111 56731 28309
请按任意键继续. . .
```

(4)对奇偶数分类

奇偶数分类实验中输入的数据和分类结果如图所示。

```
输入10个数字
19 83 76 54 30 39 66 51 72 11
19 83 11 51 39 30 66 54 72 76
请按任意键继续. . .
```

(5)验证实验：验证线性表和模板类

顺序表基本操作和求线性表长度、查找元素位置的操作结果如图所示。

```
执行插入操作前数据为:
29 31 3 67 2
此时数组长度为: 5
请输入插入的位置:
3
请输入插入的数据:
304
执行插入操作后数据为:
29 31 304 3 67 2
值为3的元素位置为: 4
执行删除第一个元素操作，删除前数据为:
29 31 304 3 67 2
删除后数据为:
31 304 3 67 2

--------------------------------
Process exited after 5.423 seconds with return value 0
请按任意键继续. . .
```

(6) 顺序表的逆置

```
请输入顺序表的长度（不超过100）：
15
请逐个输入顺序表中的数字（整数）：
10 22 93 54 77 69 32 30 55 81 24 46 37 71 38
逆置后的顺序表为：
38 71 37 46 24 81 55 30 32 69 77 54 93 22 10
请按任意键继续. . .
```

五、 总结与心得

通过这次实验，我学习了数组奇偶调整算法和数组循环移位算法这两个重要的算法。这两种算法对比其他可行的算法而言，在空间和时间的复杂度更小，可以达到省时间、内存的目的，也是我们使用计算机的主旨所在。我们知道，完成相同的工作有许多种方式，但我们必须从空间和时间两个维度去设计算法，选择合理的数据存储方式，才能高效完成任务。

本次实验特别注重对于模板函数的应用，在同时需要设计很多结构体相同而对象不同的算法时，模板函数能够很大程度上为我们节省代码时间。在某些特定情况下，使用模板函数十分必要，需要我们谨慎判断。