天津大学

数据结构实验报告

实验名称：排序

学院名称 智能与计算学部

专　　业 软件工程

学生姓名 陈昊昆

学　　号 3021001196

年　　级 2021 级

班　　级 软工 3 班

时　　间 2023 年 5 月 26 日

1. 实验内容

实现排序类 MySort ，其中 num 存储要排序的数目， origin 存储原始数据，完成以下功能：
类初始化和销毁、私有属性存取等基本功能
排序类的带参数初始化 MySort(int, int*)
参数列表为： 要排序的数目 int 原始数据 int*

奇偶交换排序 string paritySort() ，将原始数据排为升序。每一次进行交换操作，都应输 出当前的序列，最终合成并返回一个 string 类型字符串

双向冒泡排序 string bubbleSort() ，将原始数据排为升序，输出格式同上。

2. 程序实现

```
#include "mySort.h"
#include <iostream>
#include <string>

MySort::MySort(){
    num = 0;
    origin = nullptr;
}

MySort::MySort(int num, int* origin){
    this->num = num;
    this->origin = origin;
}

MySort::~MySort(){
    num = 0;
    origin = nullptr;
}

void MySort::changeSwap(int& a, int& b){
    int temp = a;
    a = b;
    b = temp;
}

string MySort::printNum(){
```

```
    string res;
    for (int i = 0; i < num; i++) {
        res += to_string(origin[i]) + " ";
    }
    res += "\n";
    return res;
}

// 奇偶交换排序
string MySort::paritySort(){
    int n = num;
    bool sorted = false;
    string res;
    while (!sorted) {
        sorted = true;
        for (int i = 1; i < n - 1; i += 2) {
            if (origin[i] > origin[i + 1]) {
                swap(origin[i], origin[i + 1]);
                sorted = false;
                res += printNum();
            }
        }
        for (int i = 0; i < n - 1; i += 2) {
            if (origin[i] > origin[i + 1]) {
                swap(origin[i], origin[i + 1]);
                sorted = false;
                res += printNum();
            }
        }
    }
    return res;
}

// 双向冒泡排序
string MySort::bubbleSort(){
    int n = num;
    int left = 0, right = n - 1;
    bool sorted = false;
    string res;
    while (!sorted) {
        sorted = true;
        for (int i = left; i < right; i++) {
            if (origin[i] > origin[i + 1]) {
                swap(origin[i], origin[i + 1]);
                sorted = false;
```

```
                res += printNum();
            }
        }
        right--;
        for (int i = right; i > left; i--) {
            if (origin[i] < origin[i - 1]) {
                swap(origin[i], origin[i - 1]);
                sorted = false;
                res += printNum();
            }
        }
        left++;
    }
    return res;
}
```

3. 实验结果

奇偶交换排序

```
4 6 5 6 1 6
4 6 5 1 6 6
4 6 1 5 6 6
4 1 6 5 6 6
1 4 6 5 6 6
1 4 5 6 6 6
```

双向冒泡排序

```
1 2 3 5 7 3
1 2 3 5 3 7
1 2 3 3 5 7
```

4. 实验中遇到的问题及解决方法

熟悉了两种排序算法:奇偶交换排序和双向冒泡排序

奇偶交换排序:

通过交换奇数位和偶数位的元素,实现排序。每次交换之后重新判断排序是否完成,如果未完成继续交换,直到排序完成。

双向冒泡排序:

从左到右和从右到左两次冒泡,每次记录最后交换的位置,作为下次冒泡的起点。通过这种双向冒泡,可以减少不必要的比较,提高排序效率。

同时也练习了字符串的拼接和 to_string 的使用