集训课-20：冒泡排序与选择排序

课程目标

01 冒泡排序

02 选择排序

03 总结回顾

Part 1

冒泡排序

冒泡排序

冒泡排序:

重复地走访要排序的元素列,依次比较两个相邻的元素,如果顺序错误就把它们交换过来，直到没有相邻元素需要交换,该元素列排序完成。

冒泡排序

使用冒泡排序,对n个元素完成从小到大的排序

冒泡排序

第一轮
第一次

下标	1	2	3	4
元素	1.64	1.32	1.86	1.21

交换

第一轮
第二次

下标	1	2	3	4
元素	1.32	1.64	1.86	1.21

不换

第一轮
第三次

下标	1	2	3	4
元素	1.32	1.64	1.86	1.21

交换

冒泡排序

第一轮

排序结果

下标	1	2	3	4
元素	1.32	1.64	1.21	1.86

不再参与排序

第一轮比较了__次,数字_____完成了排序

冒泡排序

第二轮
第一次

下标	1	2	3	4
元素	1.32	1.64	1.21	1.86

不换

第二轮
第二次

下标	1	2	3	4
元素	1.32	1.64	1.21	1.86

交换

冒泡排序

第二轮

排序结果

下标	1	2	3	4
元素	1.32	1.21	1.64	1.86

不再参与排序

第一轮比较了3次,数字1.86完成了排序

第二轮比较了_次,数字____完成了排序

冒泡排序

第三轮

第一次

下标	1	2	3	4
元素	1.32	1.21	1.64	1.86

交换

冒泡排序

第三轮

排序结果

下标	1	2	3	4
元素	1.21	1.32	1.64	1.86

不再参与排序

第一轮比较了3次,数字1.86完成了排序

第二轮比较了2次,数字1.64完成了排序

第三轮比较了_次,数字____完成了排序

冒泡排序

第三轮

排序结果

下标	1	2	3	4
元素	1.21	1.32	1.64	1.86

确定了三个元素的顺序后,是否需要继续排序?

A.是　　B.否

冒泡排序

元素个数	比较轮数	第一轮 比较次数	第二轮 比较次数	……	第n-1轮 比较次数
4	3	3	2	……	1
5	4	4	3	……	1
n	n-1	n-1	n-2		1

n个元素进行冒泡排序时比较n-1轮,

每一轮依次比较n-1次,n-2次……1次

冒泡排序

j的值: 1 ~ n－i的值;

```
for (int i = 1; i <= n-1; i++)
{
    for (int j = 1; j <= ____; j++)
    {
        if (a[j] > a[j+1])
        {
            swap(a[j],a[j+1]);
        }
    }
}
```

元素	i的值	j的值
第1轮	1	1~n-1
第2轮	2	1~n-2
第3轮	3	1~n-3
……	……	……
第n-2轮	n-2	1~n-(n-2)
第n-1轮	n-1	1~n-(n-1)

冒泡排序

n个元素,使用冒泡排序从小到大排序

```
for (int i = 1; i <= n-1; i++)
{
    for (int j = 1; j <= n-i; j++)
    {
        if (a[j] > a[j+1])
        {
            swap(a[j],a[j+1]);
        }
    }
}
```

i的范围：1 ~ n-1

j的范围：1 ~ n-i

判断条件：a[j] > a[j+1]

冒泡排序

n个元素,使用冒泡排序从大到小排序

```
for (int i = 1; i <= n-1; i++)
{
    for (int j = 1; j <= n-i; j++)
    {
        if (a[j] < a[j+1])
        {
            swap(a[j],a[j+1]);
        }
    }
}
```

i的范围：1 ~ n-1

j的范围：1 ~ n-i

判断条件：a[j] < a[j+1]

冒泡排序

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    double a[1001];
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin >> a[i];
    }
    for (int i = 1; i <= n-1; i++)
    {
        for (int j = 1; j <= n-i; j++)
        {
            if (a[j] > a[j+1])
            {
                swap(a[j],a[j+1]);
            }
        }
    }
    for (int i = 1; i <= n; i++)
    {
        cout << a[i] << " " ;
    }
    return 0;
}
```

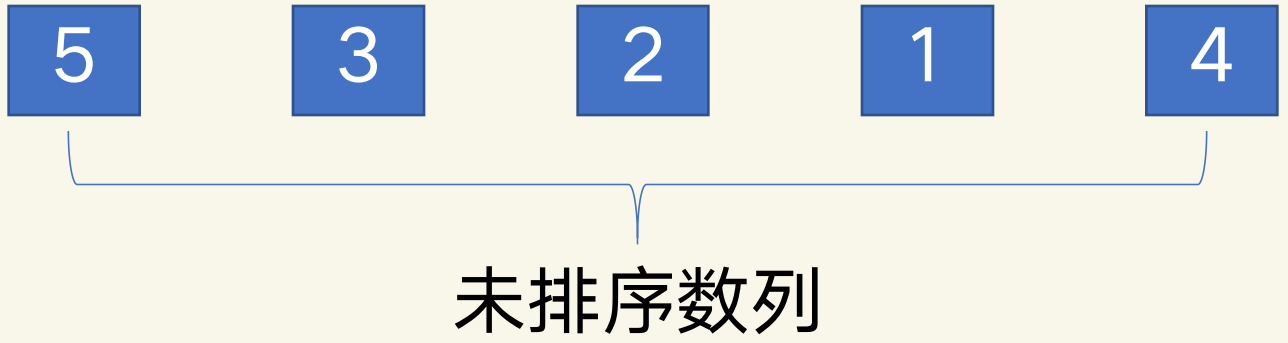
Part 2

选择排序

选择排序

简单描述:(升序)

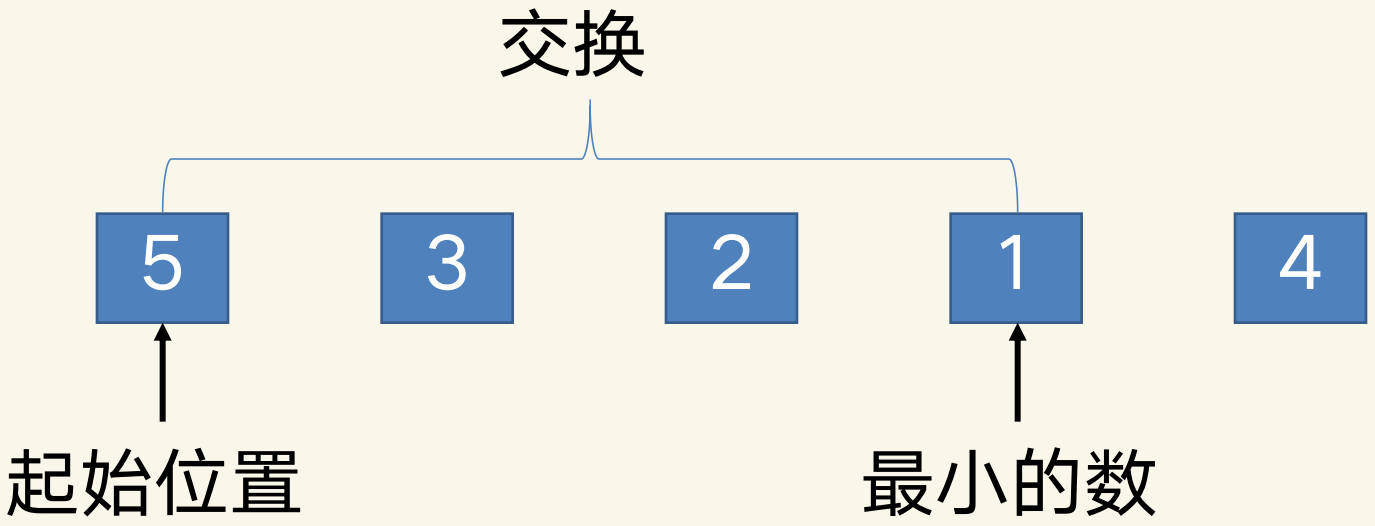
在未排序的数列中，每次将其中最小的数据与起始位置进行交换。



选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

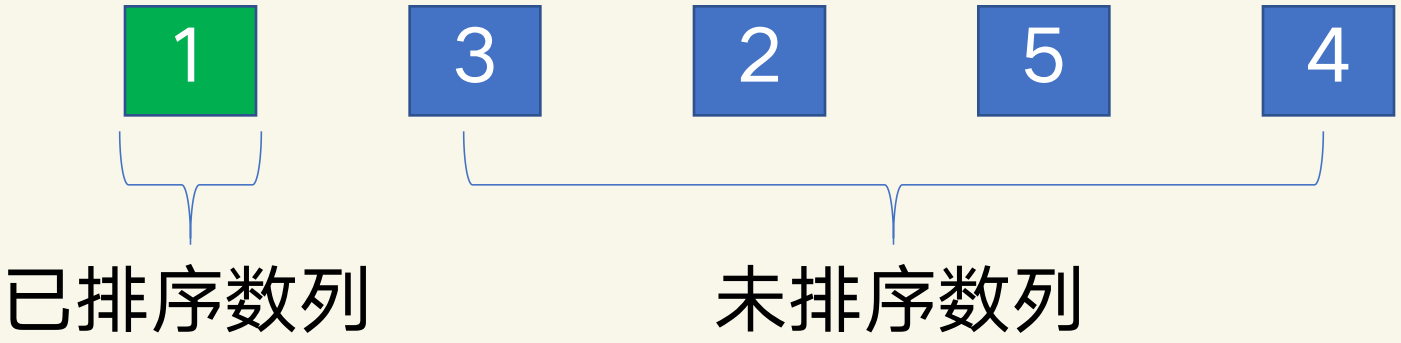


选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

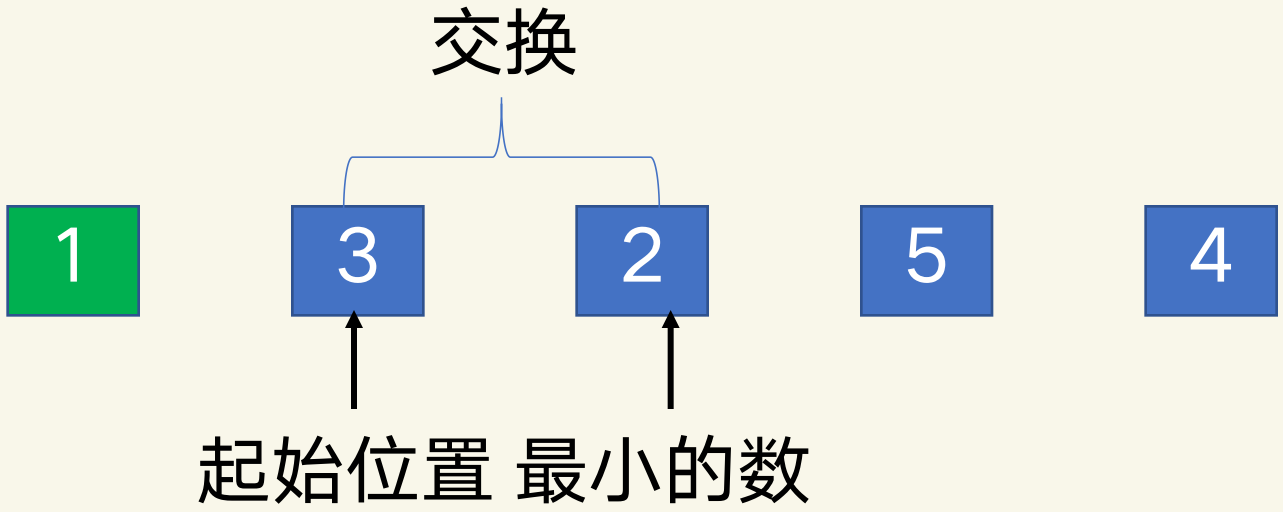
这就是第一轮选择排序



选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

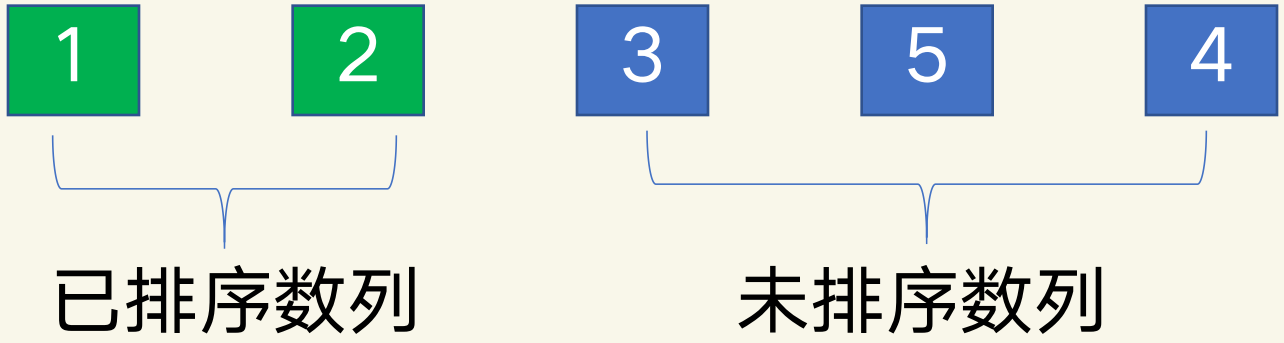


选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

这就是第二轮选择排序



选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

选择排序

简单描述:(升序)

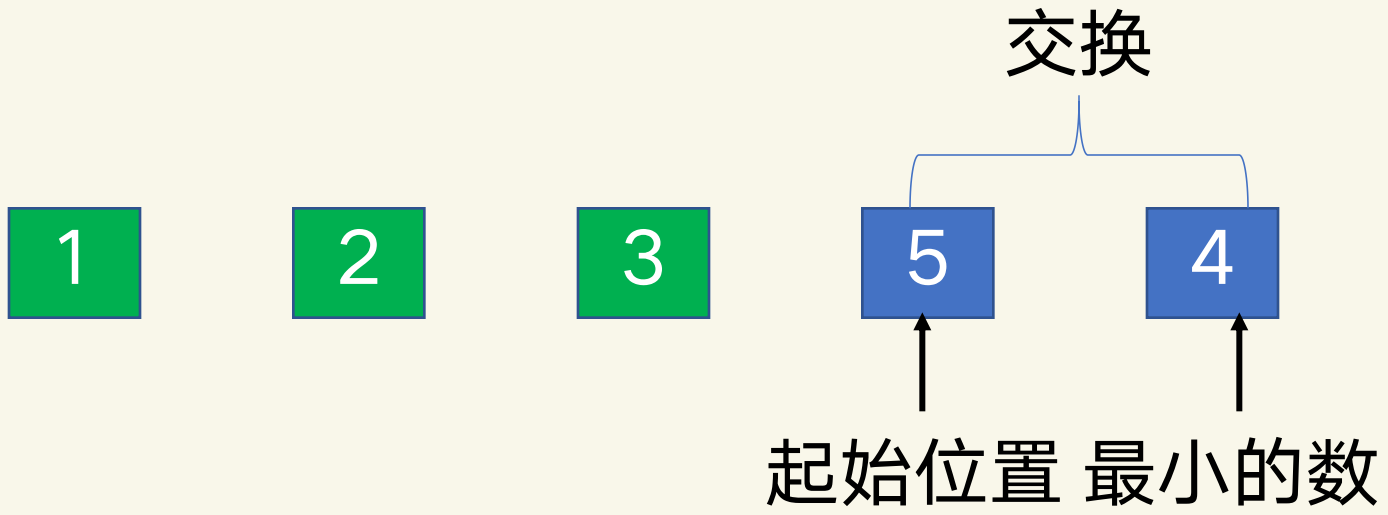
在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

这就是第三轮选择排序

选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

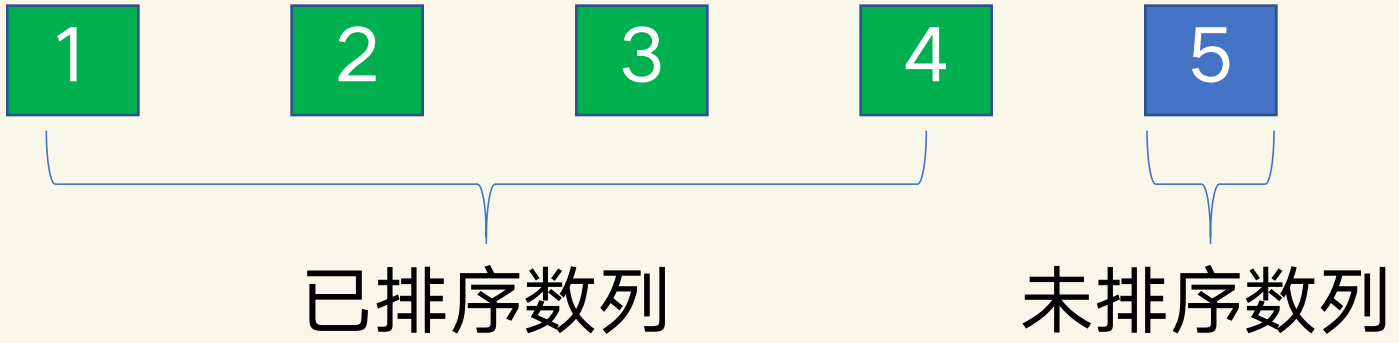


选择排序

简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

这就是第四轮选择排序



选择排序

五个元素确定了四个，最后一个元素无需在进行排序

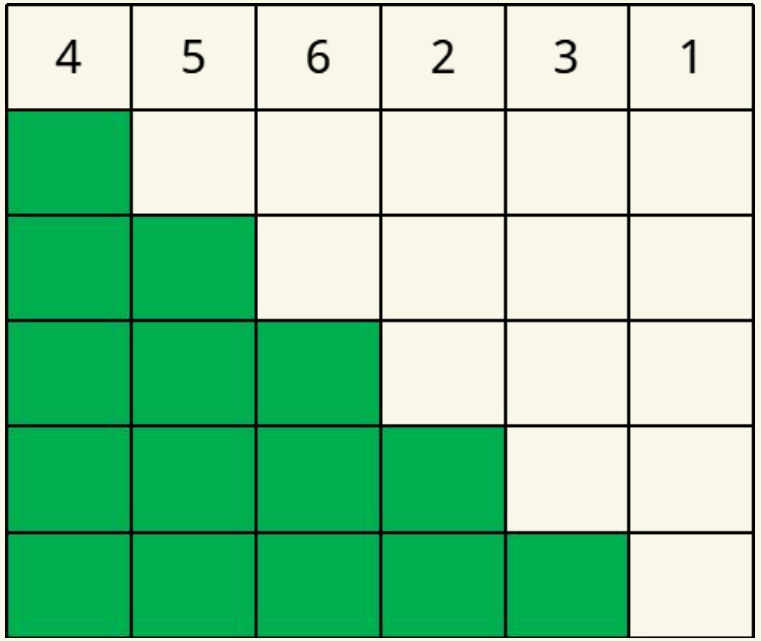
选择排序

选择排序(升序)

待排序数列

4	5	6	2	3	1

排序过程



选择排序

选择排序过程

重复执行n-1轮

1、每一轮起始位置minn = i

2、每一轮查找位置范围i+1~n

3、查找过程中更新minn的位置

4、每一轮查找结束交换位置i与位置minn上的元素

选择排序

Step1：初始化数据，并输入。

Step2：进行从小到大选择排序。

Step3：输出排序之后的结果。

选择排序

从大到小选择排序

```
for (int i = 1; i <= n - 1; i++)
{
    int maxn = i;
    for (int j = i + 1; j <= n; j++)
    {
        if (a[j] > a[maxn])
        {
            maxn = j;
        }
    }
    swap(a[i],a[maxn]);
}
```

从小到大选择排序

```
for (int i = 1; i <= n - 1; i++)
{
    int minn = i;
    for (int j = i + 1; j <= n; j++)
    {
        if (a[j] < a[minn])
        {
            minn = j;
        }
    }
    swap(a[i],a[minn]);
}
```

选择排序

```
#include <iostream>
using namespace std;

int main()
{
    int n, a[101];
    cin >> n;
    for (int i = 1; i <= n; i++)
    {
        cin  >> a[i];
    }
    for (int i = 1; i <= n - 1; i++)
    {
        int minn = i;
        for (int j = i + 1; j <= n; j++)
        {
            if (a[j] < a[minn])
            {
                minn = j;
            }
        }
        swap(a[i],a[minn]);
    }

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

    {

        cout  << a[i] << " ";

    }
    return 0;
}
```

Part 3

总结回顾

总结回顾

冒泡排序:

重复地走访要排序的元素列,依次比较两个相邻的元素,如果顺序错误就把它们交换过来，直到没有相邻元素需要交换,该元素列排序完成。

冒泡排序对n个元素从小到大排序代码

```
for (int i = 1; i <= n-1; i++)
{
    for (int j = 1; j <= n-i; j++)
    {
        if (a[j] > a[j+1])
        {
            swap(a[j],a[j+1]);
        }
    }
}
```

总结回顾

选择排序简单描述:(升序)

在未排序的数列中，每次将其中最小的数据与起始位置进行交换。

从大到小选择排序

```
for (int i = 1; i <= n - 1; i++)
{
    int maxn = i;
    for (int j = i + 1; j <= n; j++)
    {
        if (a[j] > a[maxn])
        {
            maxn = j;
        }
    }
    swap(a[i],a[maxn]);
}
```

从小到大选择排序

```
for (int i = 1; i <= n - 1; i++)
{
    int minn = i;
    for (int j = i + 1; j <= n; j++)
    {
        if (a[j] < a[minn])
        {
            minn = j;
        }
    }
    swap(a[i],a[minn]);
}
```

Class is over