

┆ 今天学习目标

* 掌握冒泡排序

* 思路：从前到后两两比较，假如前比后大的话，交换位置

* 掌握选择排序

* 思路：从当前中选出最小的。（怎么样查找最小值的下标）

* 能够比较排序算法效率

* 冒泡排序，选择排序，系统提供排序

* 总结性能最好的是：系统提供排序>选择排序>冒泡排序

* 掌握二分查找

* 拆半查找

* 先排序好的

* from,to,mid

mid= (from+to) /2

* key

if(data[mid]==key){

return mid;

}else if(data[mid]>key){

* 左

to=mid-1;

mid=(from+to)/2;

} else if(data[mid]<key){

* from=mid+1;

mid=(from+to)/2;

}

* 找不到返回-1

* 什么情况下，才不找（终止循环）from>to

* 掌握冒泡排序

* 思路：从前向后两两比较，如果前面的数比后面的数大就交换

* 分析：

```
int [] data={666,8,88,23,12,98};
```

冒泡排序（从小到大）：

思路：从前向后两两比较，如果前面的数比后面的数大就交换

```
666, 8, 88, 23, 12, 98
8, 666, 88, 23, 12, 98
8, 88, 666, 23, 12, 98
8, 88, 23, 666, 12, 98
8, 88, 23, 12, 666, 98
8, 88, 23, 12, 98, 666
```

```
8, 88, 23, 12, 98, 666
8, 88, 23, 12, 98, 666
8, 23, 88, 12, 98, 666
8, 23, 12, 88, 98, 666
8, 23, 12, 88, 98, 666
```

```
8, 23, 12, 88, 98, 666
8, 23, 12, 88, 98, 666
8, 12, 23, 88, 98, 666
8, 12, 23, 88, 98, 666
```

```
8, 12, 23, 88, 98, 666
8, 12, 23, 88, 98, 666
8, 12, 23, 88, 98, 666
```

```
8, 12, 23, 88, 98, 666
8, 12, 23, 88, 98, 666
```

```
if(data[0]>data[1]){
    data[0]交换data[1]
}

if(data[1]>data[2]){
    data[1]交换data[2]
}

if(data[2]>data[3]){
    data[2]交换data[3]
}

if(data[3]>data[4]){
    data[3]交换data[4]
}

if(data[4]>data[5]){
    data[4]交换data[5]
}
```

```
for(int i=0;i<data.length-1;i++){
    if(data[i]>data[i+1]){
        交换
    }
}

for (int i=0;i<data.length-1;i++) {
    if(data[i]>data[i+1]){
        交换
    }
}

for (int i=0;i<data.length-1-1;i++) {
    if(data[i]>data[i+1]){
        交换
    }
}

for (int i=0;i<data.length-1-2;i++) {
    if(data[i]>data[i+1]){
        交换
    }
}

for (int i=0;i<data.length-1-3;i++) {
    if(data[i]>data[i+1]){
        交换
    }
}

for (int i=0;i<data.length-1-4;i++) {
    if(data[i]>data[i+1]){
        交换
    }
}

for (int x=0;x<x.data.length-1;x++)
{
    for(int i=0;i<data.length-1-x;i++)
    {
        if(data[i]>data[i+1]){
            交换
        }
    }
}
```

交换前：

[666, 8, 88, 23, 12, 98]

交换后：

[8, 12, 23, 88, 98, 666]

```
1 public static void main(String[] args) {
2     int [] data={666,8,88,23,12,98};
3     System.out.println("交换前：");
4     System.out.println(Arrays.toString(data));
5     for(int x=0;x<data.length-1;x++) {
6         for (int i = 0; i < data.length-1-x; i++) {
7             if(data[i]>data[i+1]) {
```

```

8         int temp=data[i];
9         data[i]=data[i+1];
10        data[i+1]=temp;
11    }
12
13    }
14 }
15 System.out.println("交换后: ");
16 System.out.println(Arrays.toString(data));
17 }

```

* 断点调试查看过程替换过程

rest1.java

```

4
5 public class Test1 {
6     public static void main(String[] args) {
7         int [] data={666,8,88,23,12,98};
8         System.out.println("交换前: ");
9         System.out.println(Arrays.toString(data));
10        for(int x=0;x<data.length-1;x++) {
11            for (int i = 0; i < data.length-1-x; i++) {
12                if(data[i]>data[i+1]) {
13                    int temp=data[i];
14                    data[i]=data[i+1];
15                    data[i+1]=temp;
16                }
17            }
18        }
19    }
20 }

```

Name	Value
no method return value	
args	String[0] (id=359)
data	(id=360)
data[0]	8
data[1]	666
data[2]	88
data[3]	23
data[4]	12
data[5]	98
x	0
i	0

* 掌握选择排序

* 思路：从当前的数中找到最小的数，交换到指定位置

* int [] data={666,8,88,23,12,98};

* 分析：

选择排序（从小到大）

* 思路：从当前的数中找到最小的数，交换到指定位置

666, 8, 88, 23, 12, 98

如何从当前的数中找到最小的数？下标

666, 8, 88, 23, 12, 98

666, 8, 88, 23, 12, 98

8, 666, 88, 23, 12, 98

min

8, 12, 88, 23, 666, 98

if(data[1]<data[min]){

min=1;

8, 12, 23, 88, 666, 98

}

8, 12, 23, 88, 666, 98

if(data[2]<data[min]){

min=2;

8, 12, 23, 88, 666, 98

}

8, 12, 23, 88, 98, 666

if(data[3]<data[min]){

min=3;

if(data[4]<data[min]){

min=4;

if(data[5]<data[min]){

min=5;

}
把min元素换到0上。

8, 666, 88, 23, 12, 98

|

min

if(data[2]<data[min]){

min=2;

}
if(data[3]<data[min]){

min=3;

}
if(data[4]<data[min]){

min=4;

}
if(data[5]<data[min]){

min=5;

}

把min元素换到1上

for(int x=0;x<data.length-1;x++){

min=x;

for(int i=min+1;i<data.length;i++){

if(data[i]<data[min]){

min=i;

}

}

}

}

terminated: test1 (1) java Application: C:\Program Files\Java\jre1.6.0_10

[666, 8, 88, 23, 12, 98]

[8, 666, 88, 23, 12, 98]

[8, 12, 88, 23, 666, 98]

[8, 12, 23, 88, 666, 98]

[8, 12, 23, 88, 666, 98]

[8, 12, 23, 88, 98, 666]

```
1 public static void main(String[] args) {
2     int [] data={666,8,88,23,12,98};
3     System.out.println(Arrays.toString(data));
4     System.out.println("-----");
```

```

5      for (int x = 0; x < data.length-1; x++) {
6          int min=x;
7          for (int i = min+1; i < data.length; i++) {
8              if(data[i]<data[min]) {
9                  min=i;
10             }
11         }
12         int temp=data[x];
13         data[x]=data[min];
14         data[min]=temp;
15         System.out.println(Arrays.toString(data));
16         System.out.println("-----");
17     }
18 }

```

* 能够比较排序算法效率

* 随记产生10万条数据，通过不同排序所花的时间

冒泡排序所花的时间：18106

选择排序所花的时间：3268

系统排序所花的时间：15

```

1  package com.lg.test;
2
3  import java.util.Arrays;
4  import java.util.Random;
5
6  public class Test3 {
7      public static void main(String[] args) {

```

```
8      Random r = new Random();
9      int arrLength = 100000;
10     int arr[] = new int[arrLength];
11     for (int i = 0; i < arrLength; i++) {
12         int rand = r.nextInt(arrLength);
13         arr[i] = rand;
14     }
15     long start = System.currentTimeMillis();
16     bubbleSort(arr);
17     long end = System.currentTimeMillis();
18     System.out.println("冒泡排序所花的时间: " + (end - start));
19
20     for (int i = 0; i < arrLength; i++) {
21         int rand = r.nextInt(arrLength);
22         arr[i] = rand;
23     }
24     start = System.currentTimeMillis();
25     selectionSort(arr);
26     end = System.currentTimeMillis();
27     System.out.println("选择排序所花的时间: " + (end - start));
28
29     for (int i = 0; i < arrLength; i++) {
30         int rand = r.nextInt(arrLength);
31         arr[i] = rand;
32     }
33     start = System.currentTimeMillis();
34     Arrays.sort(arr);
35     end = System.currentTimeMillis();
36     System.out.println("系统排序所花的时间: " + (end - start));
37 }
38
39 public static void bubbleSort(int arr[]) {
40     for (int x = 0; x < arr.length - 1; x++) {
41         for (int i = 0; i < arr.length - 1 - x; i++) {
42             if (arr[i] > arr[i + 1]) {
43                 int temp = arr[i];
44                 arr[i] = arr[i + 1];
45                 arr[i + 1] = temp;
46             }
47         }
48     }
```

```
48     }
49 }
50
51 public static void selectionSort(int arr[]) {
52     for (int x = 0; x < arr.length - 1; x++) {
53         int min = x;
54         for (int i = min + 1; i < arr.length; i++) {
55             if (arr[i] < arr[min]) {
56                 min = i;
57             }
58         }
59         int temp = arr[x];
60         arr[x] = arr[min];
61         arr[min] = temp;
62     }
63 }
64 }
65
```

* 掌握二分查找

* 前提数组中元素从小到大排序

* 例子：3 , 8 , 12 , 23 , 88 , 128 , 666

二分查找（折半查找）：
前提是数组从小到大排序

```
if(data[mid]==key){
    return mid;
}else if(data[mid]>key){
    to=mid-1;mid=(from+to)/2
} else { from=mid+1;mid=(from+to)/2
```

3)key=100

from mid to
[3 , 8 , 12 , 23 , 88 , 128 , 666]

1) key=23

```
if(data[mid]==key){
    return mid;
}
```

mid
|
from

mid
|
to

2) key=12

from mid to

[3 , 8 , 12 , 23 , 88 , 128 , 666]

to

[3 , 8 , 12 , 23 , 88 , 128 , 666]

from

[3 , 8 , 12 , 23 , 88 , 128 , 666]

```
if(data[mid]==key){
    return mid;
}else if(data[mid]>key){
    to=mid-1;mid=(from+to)/2
} else { from=mid+1;mid=(from+to)/2
```

找不到就返回-1

```
1 public static void main(String[] args) {
2     int[] arr = { 3, 8, 12, 23, 88, 128, 666 };
3     int index = binarySearch(arr, 23);
4     System.out.println(index);
5     System.out.println(binarySearch(arr, 12));
6     System.out.println(binarySearch(arr, 128));
7     System.out.println(binarySearch(arr, 100));
8 }
9
10 public static int binarySearch(int[] arr, int key) {
11     int from = 0;
12     int to = arr.length - 1;
13     int mid = (to - from) / 2;
14     while (from <= to) {
15         if (arr[mid] == key) {
16             return mid;
17         } else if (arr[mid] > key) {
18             to = mid - 1;
19             mid = (from + to) / 2;
20         } else {
21             from = mid + 1;
22             mid = (from + to) / 2;
23         }
24     }
25     return -1;
26 }
```

3

2

5

-1