算法与数据结构体系课程

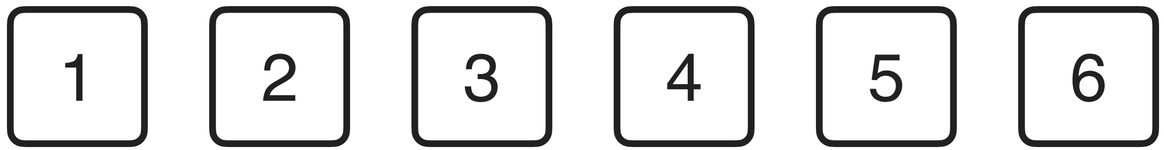
liuyubobobo

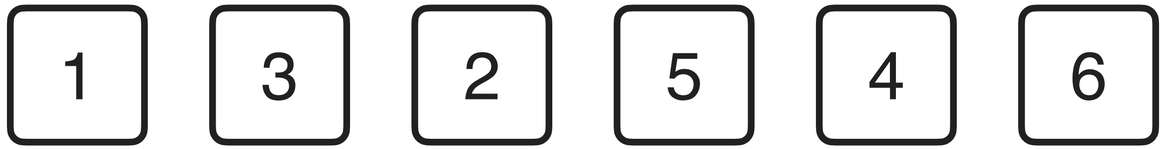
希尔排序法

liuyubobobo

希尔排序法

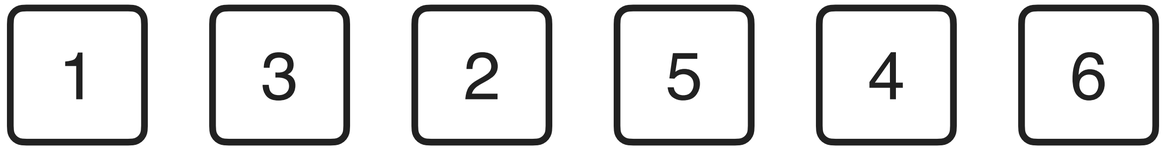
回忆插入排序法





希尔排序法

回忆插入排序法



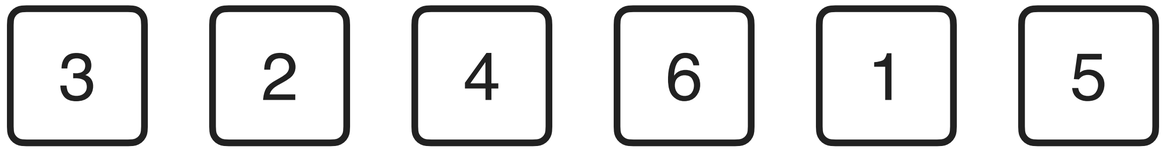
近乎有序的数组更容易排序

希尔排序法

基本思想：让数组越来越有序

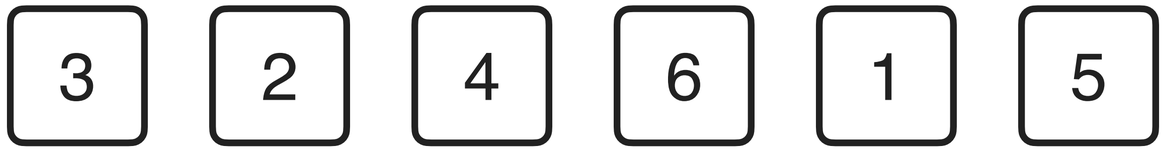
希尔排序法

再回忆冒泡排序法



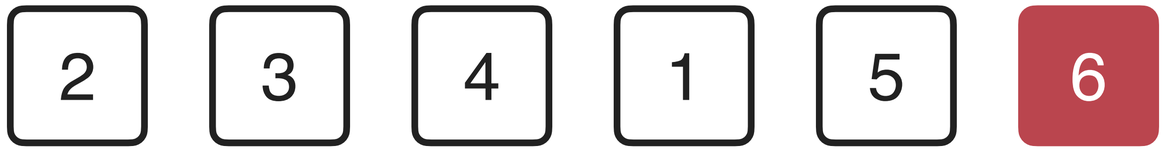
希尔排序法

再回忆冒泡排序法



希尔排序法

再回忆冒泡排序法



希尔排序法

基本思想：让数组越来越有序

不能只处理相邻的逆序对

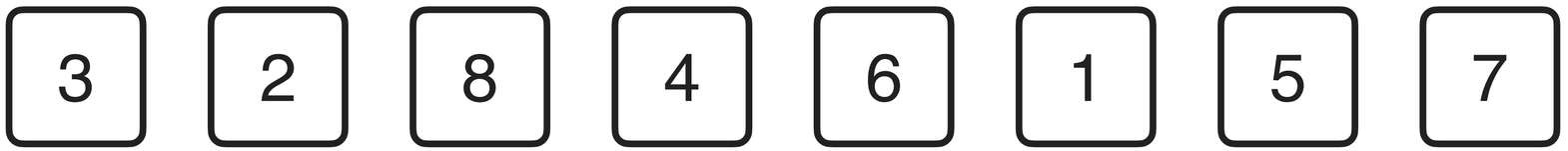
希尔排序法的基本原理

liuyubobobo

希尔排序法

基本思想：让数组越来越有序

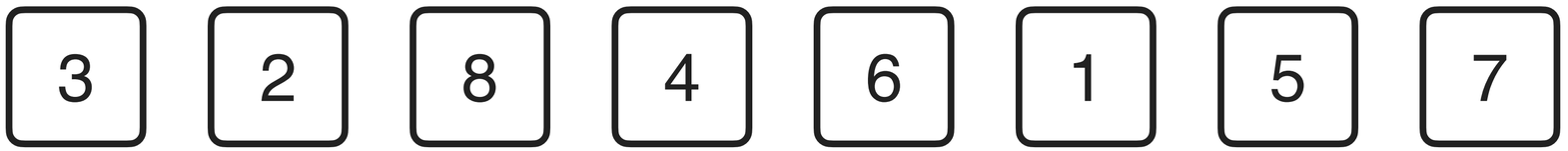
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

不能只处理相邻的逆序对



希尔排序法

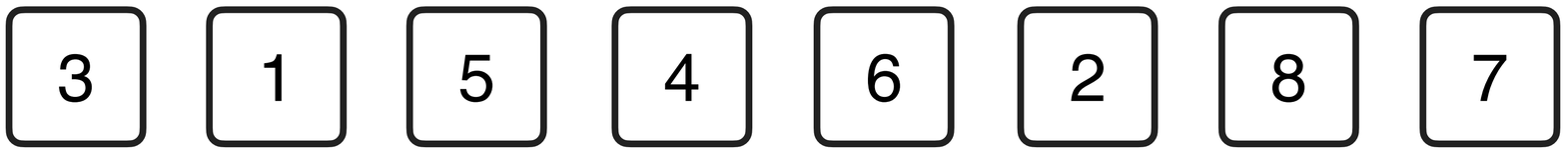
基本思想：让数组越来越有序

不能只处理相邻的逆序对

希尔排序法

基本思想：让数组越来越有序

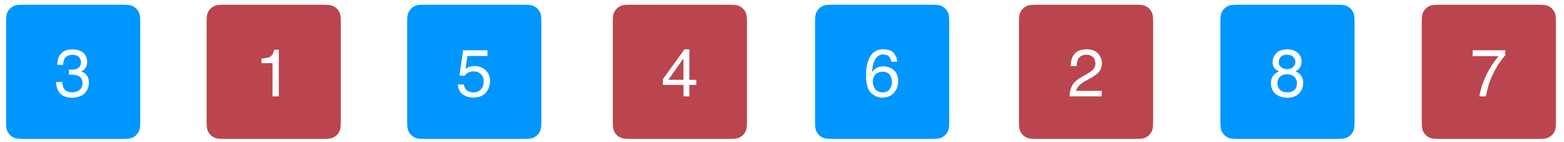
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

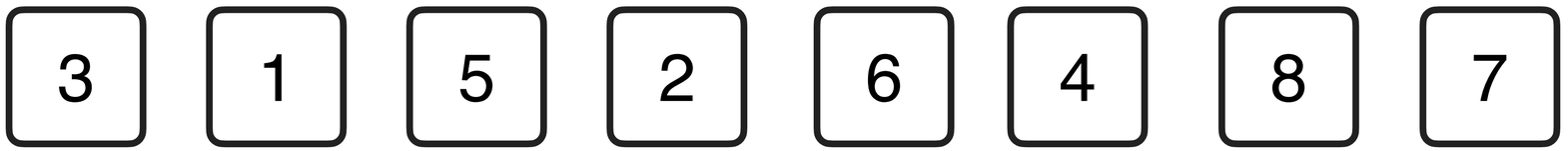
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

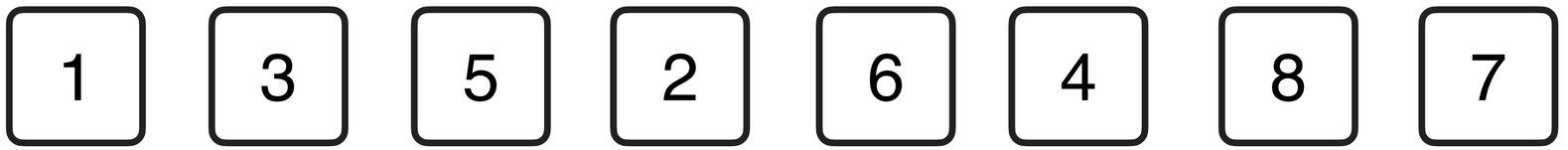
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

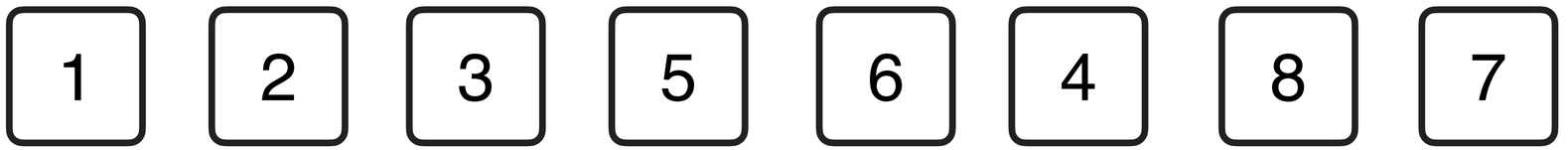
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

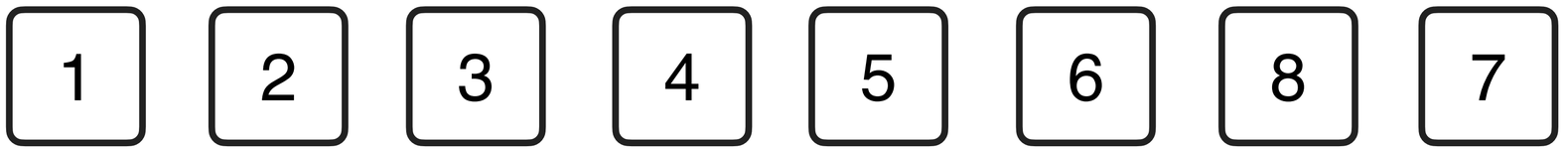
不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

不能只处理相邻的逆序对



希尔排序法

基本思想：让数组越来越有序

不能只处理相邻的逆序对

希尔排序法

对元素间距为 n / 2 的所有数组做插入排序

对元素间距为 n / 4 的所有数组做插入排序

对元素间距为 n / 8 的所有数组做插入排序

…

对元素间距为 1 的所有数组做插入排序

实现希尔排序

liuyubobobo

实现希尔排序

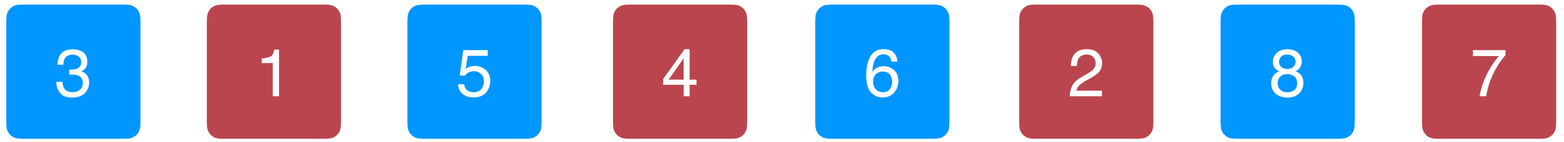
实践：实现希尔排序

希尔排序的性能

liuyubobobo

希尔排序的性能

```
for(int start = 0; start < h; start ++){

    // 对以 start 为起始，间隔为 h 的数组进行插入排序
    for(int i = start + h; i < data.length; i += h){
        E t = data[i];
        int j;
        for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
            data[j] = data[j - h];
        data[j] = t;
    }
}
```



希尔排序的性能

```
for(int start = 0; start < h; start ++){

    // 对以 start 为起始，间隔为 h 的数组进行插入排序
    for(int i = start + h; i < data.length; i += h){
        E t = data[i];
        int j;
        for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
            data[j] = data[j - h];
        data[j] = t;
    }
}
```

希尔排序的性能

```
for(int start = 0; start < h; start ++){

    // 对以 start 为起始，间隔为 h 的数组进行插入排序
    for(int i = start + h; i < data.length; i += h){
        E t = data[i];
        int j;
        for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
            data[j] = data[j - h];
        data[j] = t;
    }
}
```

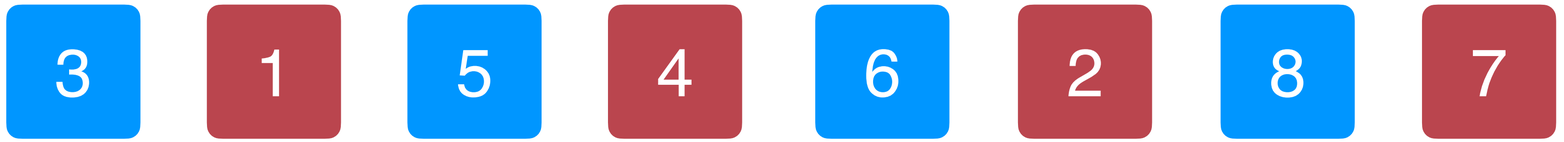
希尔排序的性能

```
for(int start = 0; start < h; start ++){

    // 对以 start 为起始，间隔为 h 的数组进行插入排序
    for(int i = start + h; i < data.length; i += h){
        E t = data[i];
        int j;
        for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
            data[j] = data[j - h];
        data[j] = t;
    }
}
```

h 轮

插入排序：(n / h) ^ 2

总共 n^2 / h



希尔排序的性能

h 轮

插入排序：(n / h) ^ 2

总共 n^2 / h

$$\frac{n^2}{1}+\frac{n^2}{2}+\frac{n^2}{4}+\ldots+\frac{n^2}{n/2} = n^2(\frac{1}{1}+\frac{1}{2}+\frac{1}{4}+\ldots+\frac{1}{n/2})$$

$$= n^2\frac{1-\frac{1}{2}^{logn}}{1-\frac{1}{2}} = O(n^2-\frac{n^2}{2^{logn}})$$

换个方式实现希尔排序

liuyubobobo

换个方式实现希尔排序

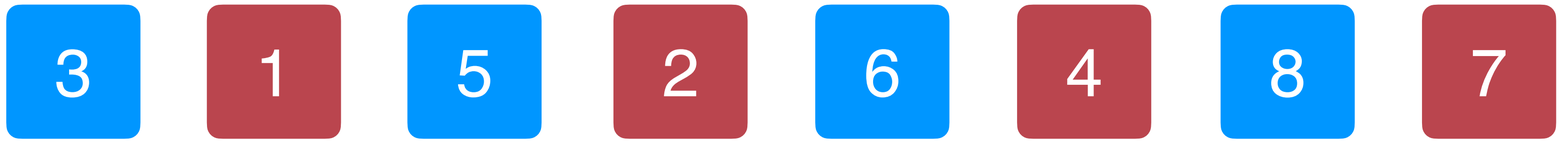
```
for(int start = 0; start < h; start ++){

    // 对以 start 为起始，间隔为 h 的数组进行插入排序
    for(int i = start + h; i < data.length; i += h){
        E t = data[i];
        int j;
        for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
            data[j] = data[j - h];
        data[j] = t;
    }
}
```

h 轮

插入排序：(n / h) ^ 2

总共 n^2 / h

把三重循环变成两重



换个方式实现希尔排序

实践：换个方式实现希尔排序

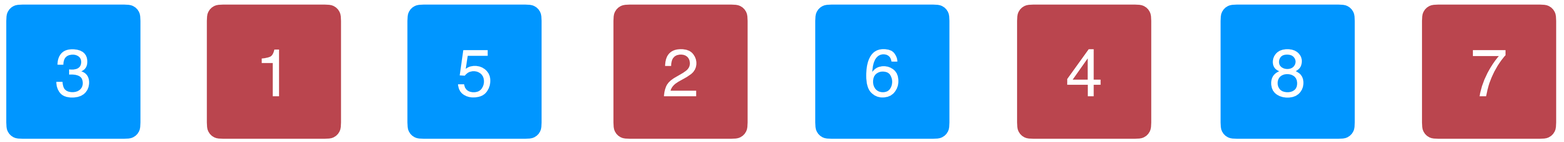
换个方式实现希尔排序

```
for(int i = h; i < data.length; i ++){
    E t = data[i];
    int j;
    for(j = i; j - h >= 0 && t.compareTo(data[j - h]) < 0; j -= h)
        data[j] = data[j - h];
    data[j] = t;
}
```

n 个元素

n / h 次比较

总共 n^2 / h



步长序列

liuyubobobo

步长序列

h ： 1， 2， 4， 8， …

h ： 1， 4， 13， 40， …

h = 3 * h + 1

步长序列

实践：新的步长序列实现希尔排序

希尔排序和超参数

liuyubobobo

希尔排序算法

h ： 1，2，4，8，…

h ： 1，4，13，40，…

不同的步长序列，复杂度分析不同

超参数

希尔排序算法

不同的步长序列，复杂度分析不同

超参数

什么步长序列最好?

https://hbfs.wordpress.com/2011/03/01/shellsort/

其他

欢迎大家关注我的个人公众号：是不是很酷

算法与数据结构体系课程

liuyubobobo