第 15 课 排序

薛浩

xuehao0618@outlook.com

阅读

• Programming Abstraction in C++ Chapter 10

复习

今日话题

• 排序

排序

一种将列表元素按顺序(升/降)排列的算法,最常用的顺序是数字顺序和字典顺序

常见排序算法比较

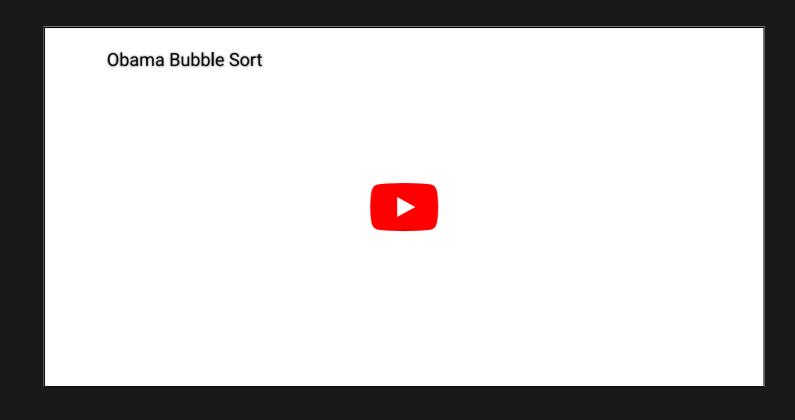
https://www.toptal.com/developers/sortingalgorithms

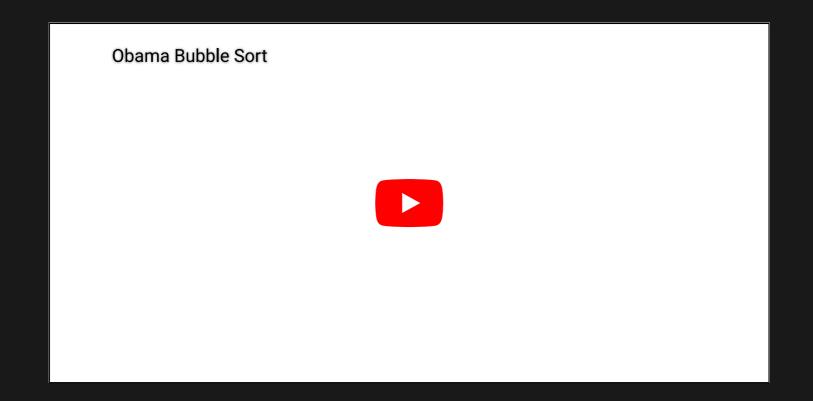
排序

- ☐ Bubble Sort
- ☐ Selection Sort
- ☐ Insertion Sort
- ☐ Merge Sort
- □ Quick Sort

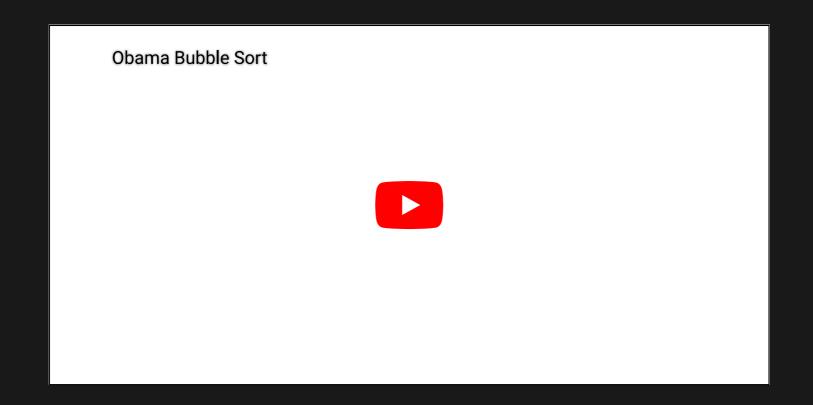


What is the most efficient way to sort a million 32-bit integers?





OBAMA: The bubble sort would be



OBAMA: The bubble sort would be the wrong way to go.

BUBBLE SORT

BUBBLE SORT

一种仅出现在课堂上的排序算法等

BUBBLE SORT

- 1. 重复遍历列表
 - 将当前元素与后面的进行比较
 - 如果当前元素较大就进行交换
- 2. 每次遍历结束,最大元素会移动到末端
- 3. 列表为空时,排序完成

算法复杂度

冒泡排序的算法复杂度

$$O(n^2)$$

排序

- Bubble Sort
- ☐ Selection Sort
- ☐ Insertion Sort
- ☐ Merge Sort
- □ Quick Sort

SELECTION SORT

SELECTION SORT

- 1. 遍历列表,找出最小的元素,放到第1个位置
- 2. 重复遍历,找出最小的元素,放到第2个位置
- 3. 重复遍历,找出最小的元素,放到第3个位置
- 4.

选择排序的算法复杂度

选择排序的算法复杂度

$$O(n^2)$$

排序

- Bubble Sort
- **■** Selection Sort
- ☐ Insertion Sort
- ☐ Merge Sort
- □ Quick Sort

INSERTION SORT

INSERTION SORT

- 1. 遍历列表,依次取出当前元素,插入到前面已排序部分
- 2. 插入元素的时候,依次向前比较
 - 如果元素较大,则向后移动一个位置
 - 否则,将元素插入到空位

最差 $O(n^2)$

最差 $O(n^2)$

最好 O(n)

 $\overline{| 最差 O(n^2) |}$

最好 O(n)

平均 $O(n^2)$

排序

- Bubble Sort
- **■** Selection Sort
- **Insertion Sort**
- ☐ Merge Sort
- □ Quick Sort

MERGE SORT

MERGE

- 1. 输入为 2 个有序列表,输出为 1 个排序列表
- 2. 两个输入列表都不为空时,比较首个元素
 - 将更小的元素取出,添加到输出列表
- 3. 当一个输入为空时,
 - 将另一个列表所有元素,添加到输出列表

MERGE SORT

```
Queue<int> merge(Queue<int> one, Queue<int> two);
Queue<int> mergeSort(Vector<Queue<int>>& all) {
    int length = all.size();
    if (length == 1)
        return all[0];
    auto left = all.subList(0, length / 2);
    auto right = all.subList(length / 2);
    return merge(mergeSort(left), mergeSort(right));
```

MERGE SORT

```
Queue<int> merge (Queue<int> one, Queue<int> two);
Queue<int> mergeSort(Vector<Queue<int>>& all) {
    int length = all.size();
    if (length == 1)
        return all[0];
    auto left = all.subList(0, length / 2);
    auto right = all.subList(length / 2);
    return merge(mergeSort(left), mergeSort(right));
```

O(nlogn)

- 16 个元素的列表
 - 插入排序需要执行 256 次

- 16 个元素的列表
 - 插入排序需要执行 256 次
- 拆分成 4 个列表,每个列表包含 4 个元素
 - 插入排序需要执行 64 次
 - 排序好的 4 个列表,再执行 merge

- 16 个元素的列表
 - 插入排序需要执行 256 次
- 拆分成4个列表,每个列表包含4个元素
 - 插入排序需要执行 64 次
 - 排序好的 4 个列表,再执行 merge
- 获得了 256 ÷ 64 = 4 倍的提升

$$O(n) \cdot O(n)$$

$$O(n) \cdot O(n)$$
 \downarrow

$$O(n) \cdot O(n)$$
 \downarrow

$$4 \cdot O(\frac{1}{4}n) \cdot O(\frac{1}{4}n)$$

$$O(n)\cdot O(n)$$
 \downarrow
 $4\cdot O(rac{1}{4}n)\cdot O(rac{1}{4}n)$
 \downarrow

$$O(n) \cdot O(n)$$

 \downarrow

$$4 \cdot O(\frac{1}{4}n) \cdot O(\frac{1}{4}n)$$

 \downarrow

$$logn \cdot O(\frac{1}{logn}n) \cdot O(\frac{1}{logn}n)$$

排序

- Bubble Sort
- **■** Selection Sort
- **Insertion Sort**
- Merge Sort |
- □ Quick Sort

QUICK SORT

QUICK SORT

- 1. 选择列表第一个元素作为"基准" (pivot)
- 2. 根据基准将列表拆分成 3 个
 - 小于基准的元素
 - 等于基准的元素
 - 大于基准的元素
- 3. 不等于基准的两个列表进行递归操作;
- 4. 递归结束后,将1个列表连接成1个列表

快速排序的算法复杂度

快速排序的算法复杂度

O(nlogn)

✔ 小试牛刀

基准问题: Random Quick Sort

排序

- Bubble Sort
- **■** Selection Sort
- **Insertion Sort**
- Merge Sort
- **■** Quick Sort

Extra Practice 1
Counting Sort

可能用到的 ADT 有哪些?

Extra Practice 2 Radix Sort

可能用到的 ADT 有哪些?

今日话题

排序

下一次课

• 二叉搜索树

THE END