计算机学院 23336003 陈政宇

Homwork1

> I.试分别举出实例说明，在对包含n个元素的序列做冒泡排序的过程中，可能发生的情况，并分析元素移动的时间复杂度。

最佳情况

在最佳情况下，序列已经是有序的。例如：

```
序列：1, 2, 3, 4, 5
```

在这种情况下，冒泡排序只需要进行一次遍历，每次比较相邻元素时都不需要交换。因此，时间复杂度为$O(n)$。

最差情况

在最差情况下，序列是逆序的。例如：

```
序列：5, 4, 3, 2, 1
```

在这种情况下，冒泡排序需要进行n-1次遍历，每次遍历都需要进行多次交换。具体来说，第i次遍历需要进行n-i次比较和交换。因此，时间复杂度为$O(n^2)$。

平均情况

在平均情况下，序列是随机排列的。例如：

```
序列：3, 1, 4, 5, 2
```

在这种情况下，冒泡排序的时间复杂度仍然是$O(n^2)$，因为在大多数情况下，序列中的元素都需要进行多次比较和交换。

> II.对n个整数的排序，能否保证在最坏的情况下，仍可在少于O(n)的时间内完成，为什么？

不能保证在最坏情况下对 n 个整数的排序在少于 $O(n \log n)$ 的时间内完成。原因如下：

比较排序的下界

对于基于比较的排序算法（如快速排序、归并排序、堆排序等），在最坏情况下的时间复杂度下界是 $O(n \log n)$。这是由比较排序的决策树模型决定的。

决策树模型

- 在比较排序中，每次比较可以将问题的规模减少一半。
- 决策树的高度代表了算法的最坏情况时间复杂度。
- 对于 n 个元素的排序，决策树的高度至少为 $\log_2(n!)$ 。
- 通过斯特林公式近似 n!，可以得到 $\log_2(n!) \approx n \log_2(n) - n \log_2(e)$，其中 e 是自然对数的底。
- 因此，比较排序的最坏情况时间复杂度下界是 $O(n \log n)$ 。

非比较排序

对于某些特定情况下的非比较排序算法（如计数排序、基数排序和桶排序），可以在 $O(n)$ 时间内完成排序，但这些算法有特定的限制条件：

- **计数排序**：适用于已知范围内的整数排序，时间复杂度为 $O(n+k)$，其中 K 是整数的范围。
- **基数排序**：适用于整数或字符串排序，时间复杂度为 $O(n \cdot d)$，其中 d 是数字的位数或字符串的长度。
- **桶排序**：适用于均匀分布的实数排序，时间复杂度为 $O(n)$（在理想情况下）。

这些非比较排序算法在特定条件下可以达到线性时间复杂度，但它们并不适用于所有情况，尤其是当数据不满足这些特定条件时。

> III.分析以下程序段的时间复杂度

A : $O(n^2)$

```
for(i = 1; i < n; i ++) {
    y = y + 1;
    for(j = 0; j < 2 * n; j ++)
        x ++;
}
```

B : $O(n^3)$

```
for(i = 1; i <= n; i ++) {
    for(j = 1; j <= i; j ++) {
        for(k = 1; k <= j; k ++)
            x = x + 2;
    }
}
```

C : $O(\sqrt[3]{n})$

```
int i = 0;
while(i * i * i <= n)
    i ++;
return i;
```