# 算法复杂度

## 时间复杂度

print('Hello World')

for i in range(n):
 print('Hello World')

for i in range(n):
 for j in range(n):
 print('Hello World')

for i in range(n):
 for j in range(n):
 for k in range(n):
 print('Hello World')

- ▶ 左面四组代码,哪组运行 时间最短?
- ▶ 用什么方式来体现算法运 行的快慢?

时间表示复杂度时,不同的机器设备之间不同,没办法归一化比较---使用几秒。

▶ 类比生活中的一些事件, 估计时间:

▶ 眨一下眼

一瞬间/几毫秒

▶ □算"29+68"

几秒

▶ 烧一壶水

几分钟

▶ 睡一觉

几小时

▶ 完成一个项目

几天/几星期/几个月

▶ 飞船从地球飞出太阳系

几年

时间复杂度也参照这个

▶ 时间复杂度:用来评估算法运行效率的一个式子

 print('Hello World')
 O(1)

 for i in range(n):
 print('Hello World')
 O(n)

 for i in range(n):
 for j in range(n):
 print('Hello World')
 O(n²)

 for i in range(n):
 for j in range(n):
 for k in range(n):
 print('Hello World')
 O(n³)

print('Hello World')
print('Hello Python')
print('Hello Algorithm')

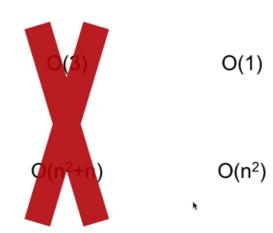
for i in range(n):
 print('Hello World')
 for j in range(n):
 print('Hello World')



1 n n^2 ......是一个单位 上升到计算机只看规模

print('Hello World')
print('Hello Python')
print('Hello Algorithm')

for i in range(n):
 print('Hello World')
 for j in range(n):
 print('Hello World')



**while** n > 1: print(n) n = n // 2

 $2^6 = 64$   $\log_2 64 = 6$ 

- ▶ 时间复杂度记为O(log<sub>2</sub>n)或 O(logn)
- ▶ 当算法过程出现循环折半的时候, 复杂度式子中会出现logn.

### 小结

- ▶ 时间复杂度是用来估计算法运行时间的一个式子(单位)。
- ▶ 一般来说, 时间复杂度高的算法比复杂度低的算法慢。
- ▶ 常见的时间复杂度(按效率排序)
  - $O(1)<O(\log n)<O(n)<O(n\log n)<O(n^2)<O(n^2\log n)<O(n^3)$
- ▶ 复杂问题的时间复杂度
  - ▶ O(n!) O(2<sup>n</sup>) O(n<sup>n</sup>) ...

#### 如何简单快速地判断算法复杂度

- ▶ 快速判断算法复杂度(适用于绝大多数简单情况):
  - ▶ 确定问题规模n
  - ▶ 循环减半过程→logn
  - ▶ k层关于n的循环→n<sup>k</sup>
- ▶ 复杂情况:根据算法执行过程判断

https://www.bilibili.com/video/BV1uA411N7c5?p=2&vd\_source=e1de9f6d02128b9c85f5fdd03c7e\_72fc\_

## 空间复杂度

- ▶ 空间复杂度: 用来评估算法内存占用大小的式子
- ▶ 空间复杂度的表示方式与时间复杂度完全一样
  - ▶ 算法使用了几个变量: O(1)
  - ▶ 算法使用了长度为n的一维列表: O(n)
  - ▶ 算法使用了m行n列的二维列表: O(mn)
- ▶ "空间换时间"

n不同则这个度量就没有办法衡量

研究算法时,时间远远重要于空间,内存已经不值钱了。

https://www.bilibili.com/video/BV1uA411N7c5?p=4&vd\_source=e1de9f6d02128b9c85f5fdd03c7e\_72fc\_