# 1.评判一个算法的好坏？

## 大O表示法（Big O）

它表示的是数据规模为 n时所 对应的复杂度。

**原则：忽略常数、系数、阶数：**

|  |
| --- |
| 9 >> O(1)  2n + 3 >> O(n)  n2 + 2n + 6 >> O(n2)  4n3 + 3n2 + 22n + 100 >> O(n3)  对数阶一般省略底数，对数阶乘以一个常数，可以替换底数。  log29 ∗ log9n= log2n  所以 O(log2n) 、O(log9n) 统称为 O(logn) |

## 常见的复杂度





## 多个数据规模的情况

时间复杂度：O(n + k)

|  |
| --- |
| public static void test(int n, int k){  for(int i = 0; i < n; i++){  System.*out*.println("test");  }  for (int i = 0; i < k; i++){  System.*out*.println("test");  } } |

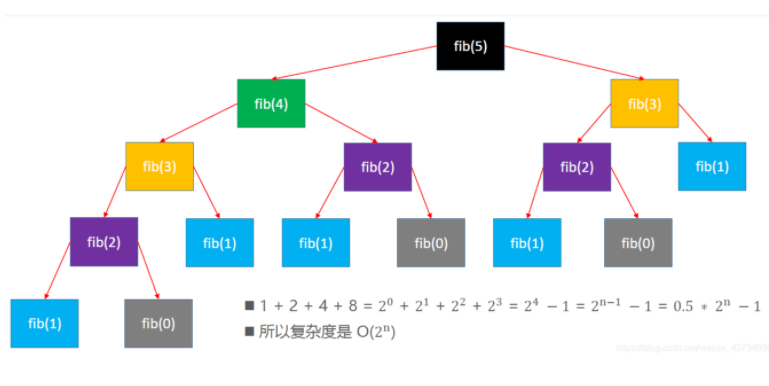
## 递归情况复杂度分析

|  |
| --- |
| public static int fib1(int n) {  if (n <= 1) return n;  return *fib1*(n - 1) + *fib1*(n - 2); } |

1个调两个

两个调四个

四个调八个

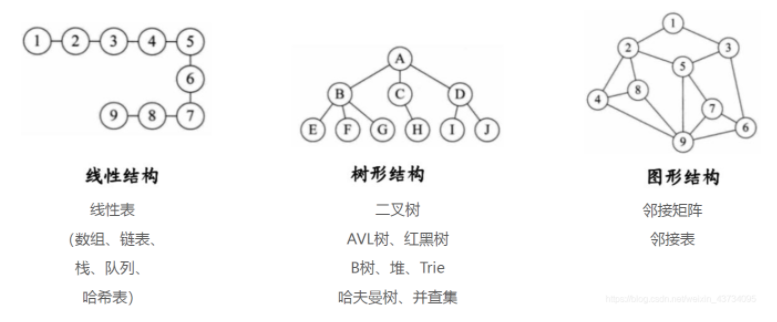


## 算法的优化方向

* 用尽量少的存储空间
* 用尽量少的执行步骤（执行时间）
* 根据情况，可以
  + 空间换时间
  + 时间换空间

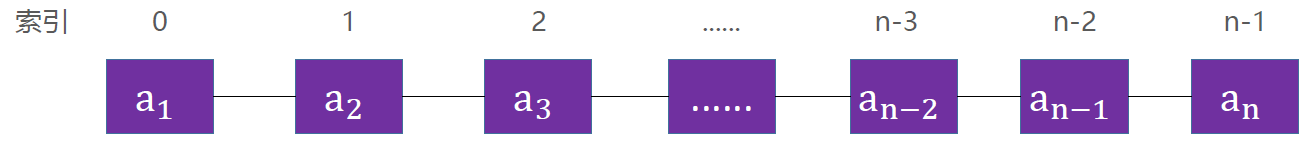
# 2.什么是数据结构？

数据结构是计算机存储、组织数据的方式；



在实际应用中，根据使用场景来选择最合适的数据结构；

## 1.线性表

线性表是具有 n 个相同类型元素的有限序列（ n ≥ 0 ）  


a1 是首节点（首元素）， an 是尾结点（尾元素）

a1 是 a2 的前驱， a2 是 a1 的后继

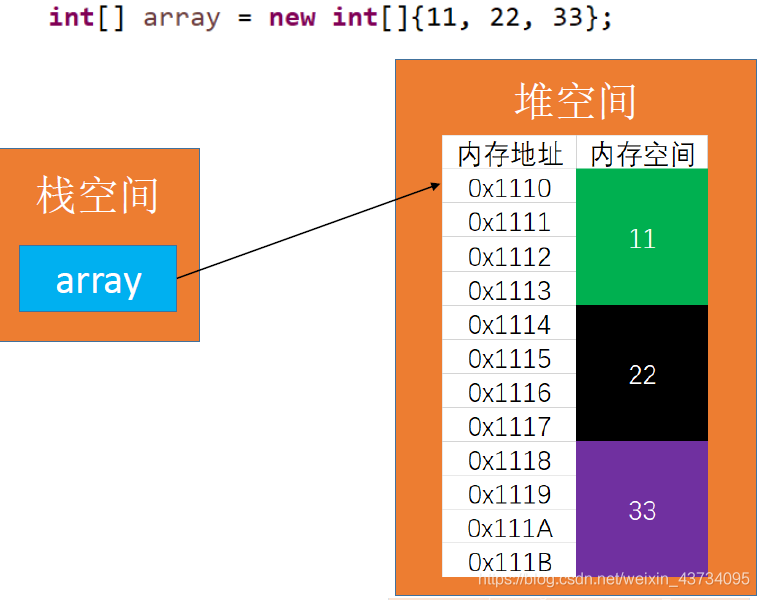
常见的线性表有：

* 数组
* 链表
* 栈
* 队列
* 哈希表（散列表）

# 线性表

## 数组

数组是一种顺序存储的线性表，所有元素的****内存地址是连续****的；



https://blog.csdn.net/weixin\_43734095/article/details/104598981