# 2.5寻找最大的K个数

# 参考

july博客：<http://blog.csdn.net/v_JULY_v/article/details/6370650>

官方wiki：<http://bop1.wikispaces.com/>

# 解法1

# 解法2

## 分析

## 时间复杂度

### 编程之美观点

### July观点

July：O(N)

### 实验分析

按照书中描述的算法进行实验。

[代码在这](2-src/solution_2_test.cc)

[输入数据在这](2-src/large-in)，数组大小N=1000000，K的值从2开始，每次递增1000，一直到N。

[输出数据在这](2-src/large-out-20140513-2)，其中各个字段的含义为：

* k：表示最大的K个数中的k
* height：表示Partition的次数，也即N被分割的深度
* count：表示S[i]>p?Sa.Append(S[i]) : Sb.Append(S[i])这行执行的次数，也就是实际的时间复杂度
* O(N\*logK)：官方时间复杂度
* O(N\*(logK))/count：官方时间复杂度是count的倍数
* cout/O(N)：count是July时间复杂度的倍数

# 解法3