算法设计与分析前四课月考试卷答案与讲评

一、合并重叠区间

1. 算法

先按左端点递增、左端点相同按右端点递增排序。若排序后区间为 $[l_i, r_i]$，扫描时维护结果最后一个区间 $[L, R]$：

```
sort(intervals, key = (left, right))
ans = []
for [l, r] in intervals:
    if ans 为空 或 l > ans[-1].right:
        把 [l, r] 加入 ans
    else:
        ans[-1].right = max(ans[-1].right, r)
return ans
```

闭区间相交当且仅当 $l \le R$，所以只有 $l > R$ 才开启新区间。样例中 $[1,3]$ 与 $[3,6]$ 在端点 3 相交，故合并。若另一个题目明确把相邻整数区间也视为可合并（例如 $[1,2]$ 与 $[3,5]$），判断才改为 $l \le R+1$。

2. 样例扫描

排序后：

$$[1,3],[3,6],[8,10],[15,18].$$

扫描轨迹：

当前区间	结果变化	原因
$[1,3]$	加入 $[1,3]$	结果为空
$[3,6]$	改为 $[1,6]$	$3 \le 3$，右端扩到 6
$[8,10]$	新开区间	$8 > 6$
$[15,18]$	新开区间	$15 > 10$

最终结果为 $[[1,6],[8,10],[15,18]]$。

3. 正确性与复杂度

扫描到当前区间时，结果中的每个区间已经合并了此前所有与它连通的输入区间，并按左端点递增排列。若当前左端点 $l > R$，它与最后结果区间不相交；由于所有更早结果区间的右端点不超过 R，它也不可能与更早区间相交，所以必须新开区间。若 $l \le R$，当前区间与最后结果区间相交，合并后的右端点取 $max(R,r)$，恰好覆盖两者并保持不漏元素。归纳可知扫描结束时所有且仅所有重叠连通的区间已合并。

排序耗时 $O(n \lg n)$，扫描耗时 $O(n)$，总时间 $O(n \lg n)$；输出数组之外的辅助空间为 $O(1)$（若把排序和输出存储计入，则为 $O(n)$）。

二、旋转升序数组中的二分查找

1. 算法

```
l = 0, r = n - 1
while l <= r:
    m = floor((l + r) / 2)
    if A[m] == s: return m

    if A[l] <= A[m]:                # 左半段有序
        if A[l] <= s < A[m]:        # 目标在左半段值域
            r = m - 1
        else:
            l = m + 1
    else:                            # 右半段有序
        if A[m] < s <= A[r]:
            l = m + 1
        else:
            r = m - 1
return -1
```

严格递增保证 $A[l] \le A[m]$ 时左半段一定有序；否则右半段一定有序。判断目标是否落在有序段的闭开值域内，就能保留可能含目标的一侧。

2. 两条轨迹

查找 $s=0$：

轮次	(l,m,r)	比较	保留区间
1	$(0,3,6)$	$A[3]=7$，左段 $[4,5,6,7]$ 有序，0 不在其值域	$[4,6]$
2	$(4,5,6)$	$A[5]=1$，右段有序，$0<1$	$[4,4]$
3	$(4,4,4)$	$A[4]=0$	返回 4

查找 $s=3$：第一轮同上保留 $[4,6]$；第二轮 $m=5, A[5]=1$，右段值域为 $[1,2]$，3 不在其中，保留左侧 $[4,4]$；第三轮检查 $A[4]=0\neq 3$，区间为空，返回 -1。

3. 正确性与复杂度

循环不变式：若目标存在，则它位于当前闭区间 $A[l..r]$。每轮先检查中点。由于数组严格递增且只发生一次旋转，中点左右至少有一侧有序。若左侧有序而 $s\in[A[l],A[m])$，只能保留左侧；否则目标不在左侧，丢弃左侧。右侧有序时同理。每轮至少删除一半候选位置，因而不变式保持，且区间最终为空或命中目标。

每轮常数次比较，区间长度至少减半，时间 $O(\lg n)$，辅助空间 $O(1)$。

三、递推式

令 $p=\log_b a$，比较 $f(n)$ 与 n^p。

1. $T(n)=2T(n/2)+n^4$：$p=\log_2 2=1$，n^4 比 n 大一个多项式因子，主定理第三种，
$$T(n)=\Theta(n^4).$$

2. $T(n)=T(7n/10)+n$：只有一个子问题，展开为
$$T(n)=n+\frac{7}{10}n+\left(\frac{7}{10}\right)^2 n+\cdots,$$
是收敛等比和，故 $T(n)=\Theta(n)$。

3. $T(n)=16T(n/4)+n^2$：$p=\log_4 16=2$，与 $f(n)=n^2$ 同阶，
$$T(n)=\Theta(n^2\lg n).$$

4. $T(n)=7T(n/3)+n^2$：$p=\log_3 7\approx 1.77<2$，$f(n)$ 大一个多项式因子且满足正则条件，
$$T(n)=\Theta(n^2).$$

5. $T(n)=7T(n/2)+n^2$：$p=\log_2 7\approx 2.81>2$，递归树叶子主导，
$$T(n)=\Theta(n^{\log_2 7}).$$

6. $T(n)=2T(n/4)+\sqrt{n}$：$p=\log_4 2=1/2$，与 $\sqrt{n}$ 同阶，

$$T(n) = \Theta(\sqrt{n}\,\lg n).$$

7. $T(n) = T(n-2) + n^2$：展开为

$$T(n) = n^2 + (n-2)^2 + (n-4)^2 + \cdots + \Theta(1).$$

共有 $\Theta(n)$ 项，每项最大 $O(n^2)$，且前半段有 $\Theta(n)$ 项都为 $\Omega(n^2)$，所以

$$T(n) = \Theta(n^3).$$

四、快速排序与插入排序混合算法

1. 流程与块间顺序

对长度 m 的区间，若 $m < k$ 直接返回；否则随机选枢轴并划分，再递归处理左右两侧。快排阶段结束后，对整个数组执行插入排序。

截断递归得到的终止区间称为小块。任取两个不同小块，考虑它们第一次被同一个祖先划分分开的时刻。若左块在枢轴左侧，则左块所有元素都小于枢轴；右块所有元素都大于枢轴，因此左块任意元素都小于右块任意元素。沿递归树继续向下，这一关系不会被破坏。故跨块不存在逆序对，剩余逆序对只能位于单个小块内部。

2. 插入排序与逆序对

插入排序处理一个 key 时，每向右移动一个大于 key 的前驱元素，就恰好消除一个逆序对；每轮还要做常数级检查。因此总时间为 $\Theta(n + I)$。

设小块长度为 $m_1, \ldots, m_s$，则 $m_i < k$ 且 $\sum_i m_i \le n$。每块最多有 $\binom{m_i}{2}$ 个逆序对：

$$I \le \sum_i \binom{m_i}{2} \le \frac{k-1}{2} \sum_i m_i \le \frac{n(k-1)}{2} = O(nk).$$

所以插入排序阶段为 $O(n + nk) = O(nk)$。

3. 第一阶段期望代价

长度为 m 的一次划分花费 $\Theta(m)$。把这项代价平均分给该区间的 m 个元素，固定跟踪一个元素。若枢轴的秩落在中间一半，例如 $m/4$ 到 $3m/4$，称为好枢轴；随机选择好枢轴的条件概率至少为 $1/2$。每次好枢轴后，跟踪元素所在的子问题规模至多缩为 $3m/4$，直到小于 k。

需要的成功缩小次数为 $O(\log_{4/3}(n/k)) = O(\lg(n/k))$；每次成功前等待的坏枢轴次数期望至多为 1，所以一个元素参与划分的期望次数为 $O(1 + \lg(n/k))$。对 n 个元素用期望的线性性相加：

$$\mathbb{E}[T_{\text{quick}}] = O\left(n + n \lg \frac{n}{k}\right).$$

与插入阶段相加，得到：

$$\mathbb{E}[T(n,k)] = O\left(nk + n \lg \frac{n}{k}\right).$$

该结论依赖互异元素和均匀随机枢轴；最坏的随机路径仍可能达到二次时间。

4. 阈值代入与命题结论

k	**总上界化简**
2	$O(n \lg n)$
$\Theta(\lg n)$	$O(n \lg n + n \lg (n/\lg n)) = O(n \lg n)$
$\Theta(\sqrt{n})$	$O(n^{3/2} + n \lg n) = O(n^{3/2})$
n	$O(n^2)$

渐近分析只说明固定常数或至多对数阈值能保持 $O(n \lg n)$；真正最优整数阈值还取决于划分、递归、缓存、比较器和插入移动的常数，应在固定实现与硬件上实测。

命题思路总结：这四题为什么会放在同一张卷子

这套卷子覆盖四种互补能力：区间题考排序后扫描与不变式，旋转数组考“局部有序 + 二分排除”，递推式考递归树、主定理和展开法，混合排序考把算法结构、逆序对和随机期望合并分析。前两题与 LeetCode 56「Merge Intervals」及 LeetCode 33「Search in Rotated Sorted Array」属于同类经典题型；闭区间端点约定需以本卷题干为准。递推式来自课程的 CLRS 分治练习，混合排序对应 CLRS 7.4-5。

后续模拟卷可以沿用以下出题模板：

1. 先给一个熟悉算法，再改变一个关键条件，例如闭区间端点、重复值、非均匀递推或停止阈值；
2. 要求“算法 + 轨迹 + 正确性 + 复杂度”，避免只背最终结论；
3. 安排一个必须处理边界的样例，如空输入、全负数组、目标不存在、区间相接；
4. 把主结论与一个易错的近邻结论并列，例如期望 $\Theta(n \lg n)$ 与最坏 $\Theta(n^2)$，或 $O(nk)$ 与实际阈值选择；
5. 让不同题共享同一分析工具：排序、二分、递推、循环不变式、逆序对、概率和期望。