PS5

1.(a) 分治法　　读题！一共 k blocks

在 ~~$O(n)$~~ $O(n)$ 时间内可以把A分成最大的 $\lceil \frac{n}{2} \rceil$ 个数和最小的 $\lfloor \frac{n}{2} \rfloor$ 个数

$T(n) = 2T(\frac{n}{2}) + O(n)$，每次保证分治的两部分都是 $\frac{n}{k}$ 的倍数

分治直到最小部分是 $\frac{n}{k}$ 的长度，即分成了 k 个 blocks

递归树 logk 层，每层时间 $O(n)$

$\log k \, O(n) = O(n \log k)$

Correctness：每次分治都可保证前一半中所有数都小于后一半

(b)

一共 k 段，每次放入都需要至少 lgk 次比较才能确定所在的 block 是 k 个中的哪一个 (worst case)

一共要放 n 次

$\Rightarrow \Omega(n \log k)$

2. (a) 记一轻一重两币为 a、b

a、b 在左/右概率均为 $\frac{1}{2}$

则有 $\frac{1}{2}$ 概率在同一侧

(b) 随机分成两个号，称重，每次有 $\frac{1}{2}$ 概率不同

使号不同次数期望：$E=\frac{1}{2}\times1+\frac{1}{4}\times2+\frac{1}{8}\times3+\dots+\frac{n}{2^n}$

$2E=1+\frac{1}{2}\times2+\dots+\frac{n}{2^{n-1}}$

$E=2E-E=1+\frac{1}{2}+\frac{1}{4}+\dots=2$

此时得到两个号堆，每堆只包含 a 或 b 其中之一

分别对两个堆二分查找

① 对于含 a 的堆，每次留轻的那一半

② 对于含 b 的堆，每次留重的那一半

⇒ 都是 $\log_2 n-1$ 次，\~~~~~~~ 找出 a 或 b

综上，一共需要 $2+2(\log_2 n-1)=2\log_2 n$ 次

3. (a) ① 首先用堆排序在 $O(n\log n)$ 根据 value 排序，同时重排 $W[i]$

并用 $V[i]$ 代替 $S[i]$，

$V[i]$ 表示 i^{th} 小元素，$W[i]$ 表示这个元素的 weight

② 在 $O(n)$ 时间计算 $W(A)$

③ 从 $i=1$ 向后遍历，递增 i，直至 weight 的和大于 $\frac{1}{2}W(A)$，记共遍历 m 项

④ 从 $i=n$ 向前遍历，递减 i，直至和小于 $\frac{1}{2}W(A)$，记共遍历 $k+1$ 项

则 $x\in[m+1,n-k]$ 为满足要求的所有 x

时间：$O(n\log n)+O(n)=O(n\log n)$

(b) 二分法，$O(n)$ 时间可在 n 个中选 k 个最大的

首先令 $x = \frac{n}{2}$ ① 若 $w(S_{<x}), w(S_{>x}) < \frac{1}{2}W(S)$，则 x 满足

② 若有1个 $> \frac{1}{2}W(S)$，则不妨假设为 $S(<x)$

此时考虑 $S(<\frac{n}{2})$，$x = \frac{n}{4}$

① 若 $wS(<x) > \frac{1}{2}W(S)$，则继续二分查找 $> \frac{1}{2}W(S)$ 的部分，

直至得到二分之后均 $< \frac{1}{2}W(S)$，记此时 $x = \frac{n}{2^i}$

于是 $w(S_{<\frac{n}{2^i}}) > \frac{W(S)}{2}$，$w(S_{<\frac{n}{2^{i+1}}}) < \frac{W(S)}{2}$

接下来在 $[a,b] \in [\frac{n}{2^{i+1}}, \frac{n}{2^i}]$ 中二分查找

每次保留的小区间需满足 $w(S_{<a}) < \frac{W(S)}{2}$，$w(S_{<b}) > \frac{W(S)}{2}$

一定会有某一个 (a,b) 满足 $b-a=1$，则 $x=b$ 满足题意。

② 若 $wS(<x) < \frac{1}{2}W(S)$

则类似上面，在 $[a,b] \in [\frac{n}{4}, \frac{n}{2}]$ 中二分查找

Time: $\frac{n}{2} + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \dots + \frac{n}{2^i} + \dots = O(n)$

4. ~~任取一个~~ 取首元素与其它 $n-1$ 个元素比较

(a)

① 若 $x=y$，则把 y 推入栈A

② 若 $x<y$，则把 y 推入栈B、跳过 ④ ⑤

③ 都比较结束后，把 x 自己推入栈A

④ ~~x~~ 把栈A连接在栈B前

⑤ 把栈B连接在栈A前

(b) ① 创建26个一级桶，遍历所有单词首字母，放入a-z对应的26个桶中

② 在每个一级桶内创建26个二级桶，遍历所有非空一级桶内有第二位字母的单词，放入对应二级桶中

③ 以此类推，直到没有单词有第m位字母，则不再创建m级桶

④ 按桶的顺序输出所有单词

分析：因为一共有n个字母，所以一共有n次入桶操作

输出可在 $O(n)$ 内完成

5. (a) 易知 $n \le 5$ 时一定成立

假设 $n \le k$ 时均成立

当 $n = k+1$ 时，

擦掉任意叶结点a，剩余部分成立

① 如剩余部分可分为3子树

由于 $k \ge 5$，再加上被擦掉的叶结点a也不会超过 $\frac{n}{2}$

否则只需擦掉3子树中最小子树的叶结点b，再在剩余的k个点中找到 central vertex，此时还原b，必然满足

② 如剩余部分分为2子树，则只需由当前 central vertex 到a的路径上，将 central vertex 移动一个结点，必成立

由归纳假设，$\forall n$ 成立

Correct：如一子树超 $\frac{n}{2}$，则移动一步，另两个加起仍不超 $\frac{n}{2}$

(b) 首先遍历树，统计总结点数 n
由根结点先向下，每次走向子树大于 $\frac{n}{2}$ 的那个子结点
直到一个结点大于 $\frac{n}{2}$ 的子树，这个子树的至多2个子树结点
数都不超过 $\frac{n}{2}$，则这个子树的根即为所求

正确性：① 树连通，每次一定有下个结点可走
② 一定有一个子树结点 $>\frac{n}{2}$，再分割则小于 $\frac{n}{2}$
③ 以这个子树根为 central vertex，除这个子树外的部分
结点数 $<\frac{n}{2}$

6. (a) 分治法，令 $C = A \cup B$
比较 A、B 的 $\frac{k}{2}^{th}$，不妨设 A 的更小
则 A 的前 $\frac{k}{2}^{th}$ 个必是 C 的前 k^{th} 个中
于是可以删去 A 中这 $\frac{k}{2}^{th}$ 个，继续查找比较 A、B 的 $\frac{k}{4}^{th}$
特殊情况说明：① 若 A 中剩于小于所需的 $\frac{k}{2}$，则使用 A 中目前最大
(B 同理)
② 当 A、B 有一个被删空时，在另一个取所需即可

Time: $O(\lg k)$
$\because k \le m+n$
$\therefore O(\lg(m+n))$

(b) 中序遍历

关键：修改叶子结点左右空指针

当前为 cur

① 如果 cur.left = null：输出 cur

cur = cur.right

② 如果 cur.left ≠ null：寻找 cur 左子树最右端结点 mostright

① 如果 mostright.right = null

mostright.right = cur（便于返回 cur）

cur = cur.left（开始遍历左子树）

② 如果 mostright.right = cur

mostright.right = null

cur = cur.right（开始遍历右子树）

③ 重复 ①.②. 直至 cur = null