分治算法作业

左逸龙 1120231863 07112303

1 PPT 上的作业

1.1 问题 1

最少需要 7 组，理由如下：

1. 首先将马分成 5 组，进行 5 场比赛，得到每一组的第一名；
2. 然后，让每一组的第一名组成一组，我们可以称这一组为胜者组。胜者组进行一场比赛，得到所有马中的第一名；
3. 随后，我们只需要再进行一场比赛就可以得到所有马当中的第二名与第三名了，理由如下：
   i. 第二名的候选者只有**胜者组当中的第二名与胜者组第一名原来所在组的第二名**；
   ii. 第三名的候选者只有**胜者组当中的第三名与胜者组第二名原来所在组的第二名与胜者组第一名原来所在组的第三名**；
   iii. 因此只需将这三匹马组成一组进行比赛，再将第二名候选者与第三名候选者分别排序，选取两组中的最快者作为胜者即可。

1.2 问题 2

答案： A.c，理由如下：
设仓库坐标为 x，距离之和为 S，则：

$$\begin{aligned} S &= |x-a|+|x-b|+|x-c|+|x-d|+|x-e| \\ &= (|x-a|+|x-e|)+(|x-b|+|x-d|)+|x-c| \end{aligned}$$

其中，由绝对值不等式：

$$\begin{cases} |x-a|+|x-e| \geq e-a & a \leq x \leq e \\ |x-b|+|x-d| \geq d-b & b \leq x \leq d \\ |x-c| \geq 0 & x = c \end{cases}$$

故当 $x=c$ 时，S 取最小值 $(e-a)+(d-b)$。

1.3 问题 3

证明中位数原理，上一问已经证明当 $n = 5$ 时，$x_p = x_3$，可以类似证明当 n 为奇数的时候，$x_p = x_{(\frac{n+1}{2})}$。

当 n 为偶数的时候，则 x_p 在最中间两点的闭区间上，该结论可仿照上面的推导由绝对值不等式得到。

2 乐学上的作业

2.1 问题 2.8

可以采用二分查找算法找到该下标。为了方便，不妨假设 T 为升序排列。

算法 1: 二分查找寻找下标 i

输入: 排好序后的整数数组 $T[1:n]$，且没有重复的数

输出: 下标 i，使得 $T[i]=i$

初始化 $left \leftarrow 1$，$right \leftarrow n$;
while $left \leq right$ do
    $mid \leftarrow left+(right-left)/2$;
    if $mid = T[mid]$ then return mid;
    else if $mid > T[mid]$ then $left \leftarrow mid+1$;
    else $right \leftarrow mid-1$;
end
return -1// 不存在下标 i

算法分析：

$$T(n)=\begin{cases}1 & n=1\\ T(n/2)+1 & n\geq 2\end{cases}$$

由分治主定理，此时 $a=1, b=2, c=1, k=0$，于是 $\log_b a=k$，因此有：

$$T(n)=O(n^{log_b a}logn)=O(logn)$$

2.2 问题 2.9

首先，关于主元素，有以下几个定理成立 (必要条件)：

1. **序列至多存在一个主元素**，因为一个主元素就会占据一半以上的数字；
2. **如果序列存在主元素，那么主元素与非主元素的个数差一定大于** 0；
3. **如果序列存在主元素，删除序列两个不同的元素，主元素不变**，因为有两种情况：
    i. 删去的两个元素都不是主元素，此时主元素不变；
    ii. 删去的两个元素中，一个是主元素，一个不是，此时主元素也不变；

我们的思路是：根据这些必要条件，寻找可能的主元素，然后再验证其是否是主元素即可。

算法 2: 减治法求序列主元素

```
输入: 数组 T[0 : n − 1]
输出: 确定 T 中是否存在一个主元素
初始化 candidate, count = 0;
for i = 0 to n − 1 do // 确定可能的主元素
    if count > n/2 then  return True ; // 提前确定主元素
    if count = 0 then
        candidate ← T[i];
        count = 1
    end
    else if candidate = T[i] then count ← count + 1;
    else count ← count − 1;
end
count ← 0;
for i = 0 to n − 1 do // 验证主元素
    if candidate = T[i] then count ← count + 1;
    if count > n/2 then return True;
end
return False
```

算法分析： 在最坏的情况下，只需要遍历数组两次即可，时间复杂度为 $O(n)$。

2.3 问题 2.25

输入元素划分成 7 个一组与 3 个一组，算法仍然是线性时间算法，理由如下：

当输入元素划分成 7 个一组的时候，算法时间复杂度：

$$T(n) = \begin{cases} O(1) & n < 147 \\ T(\frac{n}{7}) + T(\frac{3n}{4}) + O(n) & n \geq 147 \end{cases} = O(n)$$

解释如下：

为了寻找划分基准 x，我们需要将输入元素划分为 $\lceil \frac{n}{7} \rceil$ 组，并求取每组的中位数，这一步骤需要 $O(n)$；随后，我们需要求取中位数的中位数，采取递归调用 $Select$ 函数的方法，这一步骤需要 $T(\lceil \frac{n}{7} \rceil)$；当我们找到中位数的中位数作为基准 x 的时候，进行一次类似快速排序的划分，这一步骤需要 $O(n)$。

对于中位数的中位数 x，其至少比 $\frac{\lfloor \frac{n}{7} \rfloor - 1}{2} \geq \lfloor \frac{n-7}{14} \rfloor$ 个中位数要大，而每一个中位数又比 3 个元素要大，因此 x 至少比 $4\lfloor \frac{n-7}{14} \rfloor$ 个元素要大；同理，x 至少比 $4\lfloor \frac{n-7}{14} \rfloor$ 个元素要小。

当 $n \geq 147$ 的时候，有 $4\lfloor \frac{n-7}{14} \rfloor \geq \frac{n}{4}$，此时至少可以将递归区间缩小为原来的 $\frac{3}{4}$ 倍，因此无论选择左区间还是右区间递归查找，最坏情况都是需要 $T(\frac{3n}{4})$；而在 $n < 147$ 的范围内，使用任意一种排序算法，计算时间不超过一个常数，故时间复杂度为 $O(1)$。

采用分治主定理计算时间复杂度，考虑两组：$a_1 = 1, b_1 = 7, k = 1$ 与 $a_2 = 1, b_2 = \frac{4}{3}, k = 1$，而对于两组而言，均有 $\log_b a = 0 < k$，因此时间复杂度为 $O(n^k) = O(n)$。

当划分为 3 个元素的时候，类似可以得到时间复杂度为 $O(n)$。因此算法仍然是线性时间算法。