北 京 交 通 大 学 考 试 试 题（A 卷）

参 考 答 案 与 评 分 标 准

课程名称：算法设计与分析　学年学期：2022—2023 学年第 2 学期

课程编号：M210004B 开课学院：软件学院 出题教师：刘铎，吴睿智，李令昆

第一部分、单项选择题。（共 20 分）

1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
D	**A**	**C**	**B**	**D**	**B**	**D**	**C**	**B**	**D**

第二部分、计算题。（共 30 分）

11. （共 12 分）解：变量定义如图 1 所示。用 0-1 变量表示 $x_1, x_2, y_1, y_2, y_3, y_4$ 分别表示是否选择该有向边，1 表示选择，0 表示不选择。（4 分）

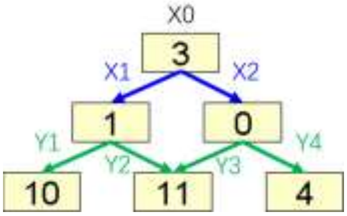


图 1

则可得到线性规划模型如下（目标 3 分、等式组 3 分、变量类型约束 2 分）。

MIN $3+1x_1+0x_2+10y_1+11y_2+11y_3+4y_4$

ST　$x_1+x_2-1=0$

　　$y_1+y_2-x_1=0$

　　$y_3+y_4-x_2=0$

　　$x_1, x_2, y_1, y_2, y_3, y_4 \in \{0, 1\}$

12. （共 10 分）解：

（1）（1+2+1 分）令 $S[1]S[2]S[3]...S[n]$表示输入序列，$L(i)$（$1 \le i \le n$）表示以 $S[i]$结束的最长单调递增子序列的长度，$P(i)$表示这个最长单调递增子序列中在 $S[i]$之前一项（即倒数第二项）的位置。如果 $L(i)=1$（即该子序列只包含 $S[i]$，而没有倒数第二项），则取 $P(i)=0$。

（2）（2+2 分）过程如表 1 所示。

（3）（2 分）最长单调递增子序列是[65 158 170 239 300 389]（65,158,170,239,300,389）。

表 1

i	1	2	3	4	5	6	7	8
$S[i]$	65	158	170	155	239	300	207	389
$L(i)$	1	2	3	2	4	5	4	6
$P(i)$	0	1	2	1	3	5	3	6

13. （共 8 分）解：

（1）$T_A(n)=k\times T_A(n/5)$　　$T(5)=k$　　$T(n)\in\Theta(n^{\log_5 k})$　　（2+1 分）

（2）$T_B(n)=t\times T_B(n/4)$　　$T(4)=t$　　$T(n)\in\Theta(n^{\log_4 t})$　　（2+1 分）

（3）当$\log_5 k < \log_4 t$时，即$\frac{\ln k}{\ln t} < \frac{\ln 5}{\ln 4} = \log_4 5$时，算法 $\boldsymbol{A}_{\mathrm{plus}}$ 的时间复杂性将优于 $\boldsymbol{B}_{\mathrm{plus}}$ 的。（2 分）

第三部分、综合分析题。（共 50 分）

14. （共 12 分）解：（过程不唯一）

参考解法一（by Li）。

（1）（9 分）

```
IS-MATCH (str, prefix)
    if (len(str) < len(prefix)) then return FALSE
    for i = 1 to m
        if str[i] ≠ prefix[i]
            return FALSE
    return TRUE
LOWER-BOUND (A, prefix)
    l ← 0, r ← n+1
    while (r - l > 1)
        mid ← (l+r) / 2
        if (A[mid] >= prefix) then r ← mid
        else l ← mid
    return r
```

```
UPPER-BOUND (A, prefix, index)
    l ← index, r ← n+1
    while (r - l > 1)
        mid ← (l+r) / 2
        if (IS-MATCH(A[mid], prefix)) then l ← mid
        else r ← mid
    return r
COUNT-PREFIX (A, prefix)
    l ← LOWER-BOUND(A, prefix)
    if (l=n+1 or (not IS-MATCH(A[l], prefix))) then return 0
    r ← UPPER-BOUND(A, prefix, l)
    return r - l
```

（2）（3 分）**IS-MATCH** 的时间复杂度为 $O(m)$，**LOWER-BOUND** 的时间复杂度为 $O(\log n)$，**UPPER-BOUND** 的时间复杂度为 $O(m \log n)$。因此 **COUNT-PREFIX** 的时间复杂度为 $O(m \log n)$。

参考解法二（by Liu）。

（1）（9 分）

```
IS-MATCH (str, prefix)
    if (lengthof(str) < m) then return FALSE
    for i = 1 to m
        if str[i] ≠ prefix[i] then
            return FALSE
    return TRUE
FindFirst (left, right)
    low←left, high←right, label←-1
    while (low≤high) do
        mid←(low+high)/2
        if (IS-MATCH(A[mid], prefix))
            label←mid, high←mid-1
        else if (Key<A[mid]) high←mid-1
        else low←mid+1
    return label
```

```
FindLast (left, right)
    low←left, high←right, label←-1
    while (low≤high) do
        mid←(low+high)/2
        if (IS-MATCH(A[mid], prefix))
            label←mid, low←mid+1
        else if (Key<A[mid]) high←mid-1
        else low←mid+1
    return label
COUNT-PREFIX (A, prefix)
    输入：A[0..n–1], prefix
    输出：A 中前缀为 prefix 的字符串个数
    from←FindFirst(0, n-1)
    to←FindLast(0, n-1)
    if (from<0) then output 0
    else output to-from+1
```

（2）（3 分）**IS-MATCH** 的时间复杂度为 $O(m)$，设 **FindFirst**/**FindLast** 的时间复杂度为 $T(n)$（二者流程类似），则 $T(n)=T(n/2)+O(m)$，$T(1)\in O(m)$，即 $T(n)\in O(m\log n)$。因此 **COUNT-PREFIX** 的时间复杂度为 $O(m\log n)$。

15. （共 12 分）解：

（1）（4 分）将怪兽按 T/D 的值由小到大排序，依次击败。

（2）（4 分）（参考示例伪代码，不唯一）

```
BEAT-MONSTER (Monster[])
    SORT(Monster)                    //将怪兽按Ti/Di按不减顺序排序
    return Monster[]
```

（3）（4 分）

（证明方式一）假设有一个怪兽的击败顺序为$A=(1',2',3',\cdots,n')$，则怪兽$1'$对村庄的破坏强度为 0（一开始就击败它，不会对村庄产生破坏），怪兽$2'$对村庄的破坏强度为$T_{1'}\times D_{2'}$，怪兽$3'$对村庄的破坏强度为$(T_{1'}+T_{2'})\times D_{3'}$……，以此类推，怪兽$k'$对村庄的破坏强度为$D_{k'}\times\sum_{i=1}^{k-1}T_{i'}$。

同理，怪兽$(k+1)'$对村庄的破坏强度为$D_{(k+1)'}\times\sum_{i=1}^{k}T_{i'}$

交换怪兽k'和$(k+1)'$对怪兽$i\in[1',(k-2)']$和怪兽$i\in[(k+2)',n']$对村庄产生的总破坏无影响。（8 分）

因此设顺序B为顺序A中交换怪兽k'和$(k+1)'$，对村庄产生的总破坏变小了，即有：

$$
\begin{aligned}
D(B)-D(A)&=D_{(k+1)'}\times\sum_{i=1}^{k-1}T_{i'}+D_{k'}\times\left(\sum_{i=1}^{k-1}T_{i'}+T_{(k+1)'}\right)\\
&-D_{k'}\times\sum_{i=1}^{k-1}T_{i'}-D_{(k+1)'}\times\sum_{i=1}^{k}T_{i'}\\
&=D_k'\times T_{(k+1)'}-D_{(k+1)'}\times T_k'<0
\end{aligned}
$$

即$D_{k'}\times T_{(k+1)'}<D_{(k+1)'}\times T_k'$，即$\frac{T_{k'}}{D_k'}>\frac{T_{(k+1)'}}{D_{(k+1)'}}$。

也就是说，若任意一个顺序中存在相邻两个怪兽，它们满足关系$\frac{T_{k'}}{D_k'}>\frac{T_{(k+1)'}}{D_{(k+1)'}}$，那么我们交换击败它们的先后顺序，可以使得对村庄的总的破坏力变小。在新的击败顺序中，如果还有相邻的两个怪兽满足上述关系，再交换它们可以使得村庄受到的总破坏再变小，重复这个过程，当顺序无法产生任何交换时，村庄受到的总的破坏力是最小的，此时有击败顺序$(1',2',3',\cdots,n')$，满足：$\frac{T_{1'}}{D_{1'}}<\frac{T_{2'}}{D_{2'}}<\frac{T_{3'}}{D_{3'}}<\cdots<\frac{T_{n'}}{D_{n'}}$。

（证明方式二）

首先，最优方案中无空闲时间，且打倒所有怪兽的总时间是确定的。

定义：若在打怪兽顺序 $i_1, i_2, \cdots, i_n$ 中存在 $1\le a<b\le n$ 且$\frac{T_{i_a}}{D_{i_a}}>\frac{T_{i_b}}{D_{i_b}}$，则称此时$(i_a, i_b)$构成一个逆序。

如果一个（无空闲时间的）打怪兽顺序方案中有一对逆序，那么就一定存在两个相邻的任务形成逆序。（学生可直接使用此结论而不加证明。）

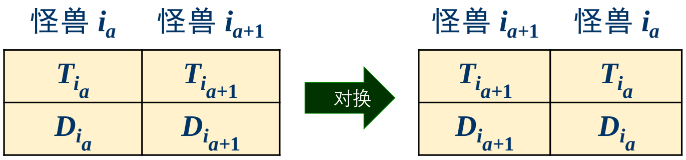
由算法可知贪婪解中不存在逆序。

假设 S^*是一个最优解，其打怪兽顺序为 $i_1, i_2, \cdots, i_n$。

①若 S^*中不存在逆序，则其就是贪婪解。

②如果 S^*中存在逆序(i_a, i_{a+1})，则对换打怪兽 i_a 和打怪兽 i_{a+1} 的次序后，

总破坏的变化值为：$D_{i_a}\times T_{i_{a+1}}-D_{i_{a+1}}\times T_{i_a}=D_{i_a}D_{i_{a+1}}\left(\frac{T_{i_{a+1}}}{D_{i_{a+1}}}-\frac{T_{i_a}}{D_{i_a}}\right)<0$，与 S^*的最优性产生矛盾。



因此最优解必定与贪婪解相同。

16. （共 13 分）**解**：（解法不唯一，采用不同思路亦可根据实际情况适当给分）

（1）（共 6 分）令$W=\lfloor \sum_{i=1}^{n} w_i/2 \rfloor$。设两队人的总体重为 A 和 B，$A\le B$，则$B-A=\sum_{i=1}^{n} w_i-2A$。要使得$B-A$尽量小，即等价于在$A\le\sum_{i=1}^{n} w_i$且 $A\le B$ 且 $A+B=\sum_{i=1}^{n} w_i$条件下使得 A 的值尽量大。而这又等价于在$A\le W$条件下使得 A 的值尽量大（于是类似于背包问题/装载问题）。

令从前 k 个人中选择总体重不超过 W 的若干人所能取得的最大总体重为 $F(k, y)$。

递推关系及初值为：

$$F(k,y)=\begin{cases} 0 & ,\text{if } k=0 \text{ or } y=0 \\ F(k-1,y) & ,1\le k\le n, 1\le y\le W, w_k>y \\ \min\{F(k-1,y), F(k-1,y-w_k)+w_k\} & ,1\le k\le n, 1\le y\le W, w_k\le y \end{cases}$$

伪代码为：

输入：人数 n, n 人的体重 $w_1, w_2, \cdots, w_n$

输出：所有分队方案中最小可能体重差

```
W ← ⌊∑_{i=1}^{n} w_i /2⌋
for y = 0 to W
    F(0, y)←0
for k = 1 to n
    F(k, 0)←0
for k = 1 to n
    for y = 1 to W
        F(k, y)←F(k–1, y)
        if w_k≤y then
            if F(k–1, y–w_k)+w_k>F(k, y) then
                F(k, y)←F(k–1, y–w_k)+w_k
return (∑_{i=1}^{n} w_i)/2 − 2F(n, W)
```

（2）（共 7 分）过程如表 2 所示（$W = \lfloor \sum_{i=1}^{n} w_i / 2 \rfloor = 13$），最小体重差为 2——分为 11、3 和 5、4、3 两队。

表 2

k \ y	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	11	11	11
2	0	0	0	0	0	5	5	5	5	5	5	11	11	11
3	0	0	0	0	4	5	5	5	5	9	9	11	11	11
4	0	0	0	3	4	5	5	7	8	9	9	11	12	12
5	0	0	0	3	4	5	6	7	8	9	10	11	12	12

17. （共 13 分）**解：**（解法不唯一，采用不同思路亦可根据实际情况适当给分）

问题分析同题 16。

（1）（3 分）依次考虑各个人。用 0-1 变量表示 x_1, x_2, x_3, x_4, x_5 分别表示是否选择第 i 个人加入体重和为 A 的队伍（即不是总体重“偏重”的一队），1 表示选择，0 表示不选择。

维护“到目前为止，在$A \le W$条件下，A 的最优值”*current_best*。

设已经考虑了 i 个人，用 $C_a(i)$表示目前已经选定的人的总体重，$C_e(i)$表示未考虑的人的总体重，则估界函数为 $C_a(i)+C_e(i)$。

若 $i=n$ 且 $C_a(n)>current\_best$，则更新 *current_best* 为 $C_a(n)$。

剪枝依据：

①若 $i \neq n$，且 $C_a(i)> W$，则进行剪枝。（事实上，先判断加第 $i+1$ 个人的体重后是否超过 W，如果是的话则不再考虑继续进行。）

②若 $i \neq n$，且 $C_a(i)+C_e(i) \le current\_best$ 则进行剪枝。

（2）（7 分）剪枝后的（部分）搜索树如图 2 所示。

（3）（3 分）选择体重分别为 11、3 的组合一队，其他人组成另一队，最小总体重差为 2。

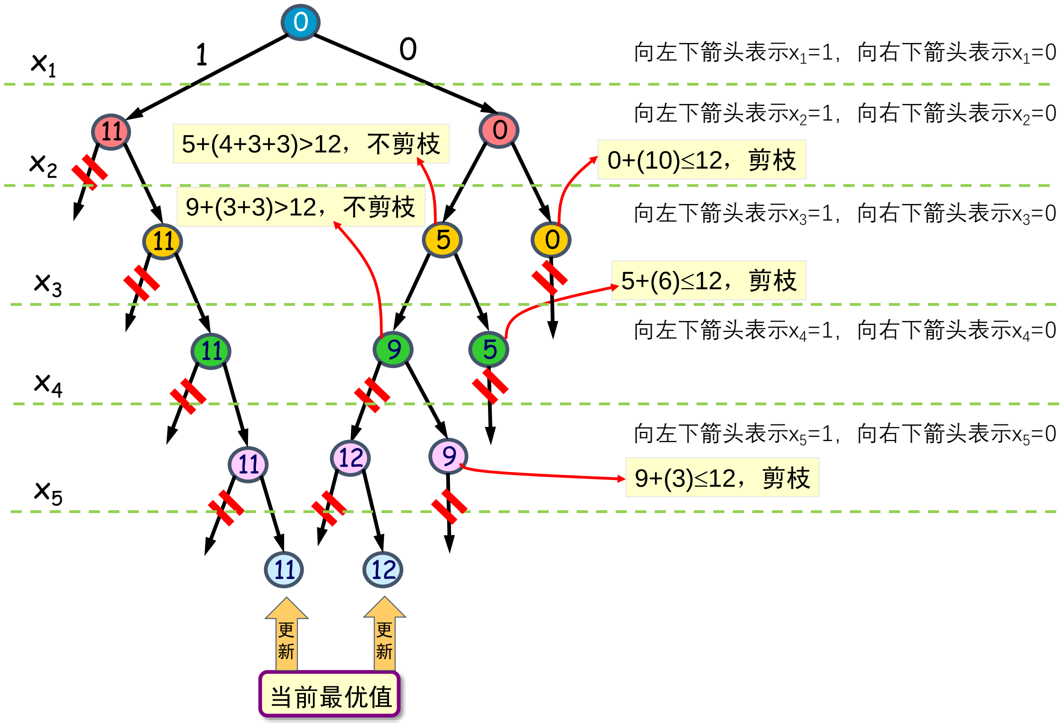


图 2 题目 15 解答用图 3