刘子康　2022113416

第二章作业

1、(1)　$a=5$，$b=3$，$f(n)=n$

$n^{\log_b a} = \Theta(n^{\log_3 5})$　　令 $\varepsilon = \log_3 5 - 1$

则有 $f(n) = n = O(n^{\log_b a - \varepsilon})$

由master定理可知，$T(n) = \Theta(n^{\log_b a}) = \Theta(n^{\log_3 5}) = \Theta(n^{1.46})$

(2)　$a=2$，$b=2$，$f(n)=n^{\frac{1}{2}}$

$n^{\log_b a} = \Theta(n)$　　令 $\varepsilon = \frac{1}{2}$

则有 $f(n) = n^{\frac{1}{2}} = O(n^{\log_b a - \varepsilon})$

由master定理可知，$T(n) = \Theta(n^{\log_b a}) = \Theta(n)$

(3)
$$
\begin{aligned}
T(n) &= T(\lfloor \tfrac{n}{2} \rfloor) + T(\lfloor \tfrac{3}{4}n \rfloor) + n \\
&= T(\lfloor \tfrac{n}{4} \rfloor) + 2T(\lfloor \tfrac{3}{8}n \rfloor) + T(\lfloor \tfrac{9}{16}n \rfloor) + \tfrac{5}{4}n + n \\
&= T(\lfloor \tfrac{n}{8} \rfloor) + 3T(\lfloor \tfrac{3}{16}n \rfloor) + 3T(\lfloor \tfrac{9}{32}n \rfloor) + T(\lfloor \tfrac{27}{64}n \rfloor) + (\tfrac{5}{4})^2 n + \tfrac{5}{4}n + n \\
&= T(\lfloor \tfrac{n}{16} \rfloor) + 4T(\lfloor \tfrac{3}{32}n \rfloor) + 6T(\lfloor \tfrac{9}{64}n \rfloor) + 4T(\lfloor \tfrac{27}{128}n \rfloor) + T(\tfrac{81}{256}n) + (\tfrac{5}{4})^3 + (\tfrac{5}{4})^2 + \tfrac{5}{4}n + n \\
&= \cdots
\end{aligned}
$$

$$n \sum_{k=0}^{n} (\tfrac{5}{4})^k = \frac{1-(\frac{5}{4})^{n+1}}{1-\frac{5}{4}} n = [4(\tfrac{5}{4})^{n+1} - 4]n$$

第i次迭代，会增加一项 $(\frac{5}{4})^i n$，

由 $T(\frac{3}{4}n)$ 的迭代可知 i最大为 $\log_{\frac{4}{3}} n$，而进行到第 $\log_2 n$ 次迭代时出现 $T(1)$，

此后增加项小于 $(\frac{5}{4})^i n$

则代价和小于 $n + n\sum_{i=1}^{\log_{\frac{4}{3}} n} (\frac{5}{4})^i = n + 5n(n^{\log_{\frac{4}{3}} \frac{5}{4}} - 1)$，大于 $n + n\sum_{i=1}^{\log_2 n} (\frac{5}{4})^i = n + 5n(n^{\log_2 \frac{5}{4}} - 1)$

下证 $T(n)$ 的上界为 $O(n^2)$

$$
\begin{aligned}
T(n) &= T(\lfloor \tfrac{n}{2} \rfloor) + T(\lfloor \tfrac{3}{4}n \rfloor) + n \\
&\le c \cdot \tfrac{n^2}{4} + c \cdot \tfrac{9}{16}n^2 + n = \tfrac{13}{16}n^2 + n = cn^2 + n - c \cdot \tfrac{3}{16}n^2
\end{aligned}
$$

令 $n - c \cdot \frac{3}{16}n^2 \le 0$　即 $c \ge \frac{16}{3n}$　取 $c = \frac{16}{3}$　则 $T(n) \le cn^2$

综上所述：$T(n) = O(n^2)$

~~第二章作业~~

n=1时，

2、证明：① $F(3)=F(2)+F(1)=2F(1)+F(0)=3>(\frac{1+\sqrt{5}}{2})^1$ 成立

② 假设当 $n\le k$ 时，~~$F(k+2)>(\frac{1+\sqrt{5}}{2})^k$ 成立 $(k\ge1)$~~ $F(n+2)>(\frac{1+\sqrt{5}}{2})^n$ 成立 $(k\ge2)$

往证 $n=k+1$ 时 $F(n+2)>(\frac{1+\sqrt{5}}{2})^n$ 也成立

$$F(k+1+2)=F(k+3)=F(k+2)+F(k+1)>(\frac{1+\sqrt{5}}{2})^k+(\frac{1+\sqrt{5}}{2})^{k-1}$$

$$\because 1+\frac{2}{\sqrt{5}+1}=\frac{2\sqrt{5}+6}{2(\sqrt{5}+1)}=\frac{\sqrt{5}+1}{2}$$

$$\therefore (\frac{\sqrt{5}+1}{2})^k+(\frac{\sqrt{5}+1}{2})^{k-1}=(\frac{\sqrt{5}+1}{2})^{k+1}$$

$\therefore F(k+1+2)>(\frac{1+\sqrt{5}}{2})^{k+1}$ 即 $F(n+2)>(\frac{1+\sqrt{5}}{2})^n$ 成立

综上：由数学归纳法可知 $F(n+2)>(\frac{1+\sqrt{5}}{2})^n$ 成立

第三章作业

3. (1) 预处理：若 $n\le2$，则算法结束，否则把 S 中的点分别按 x-坐标和 y-坐标排序

(2) Divide：① 计算 S 中所有点 x 坐标的中位数 m

② 用 $x=m$ 把 S 划分为 2 个子集 S_L 和 S_R，S_L 中点在 $x=m$ 左侧，S_R 中点在 $x=m$ 右侧

③ 递归地在子集 S_L 和 S_R 上找出能构成三角形且周长最短的 3 个点，

$(p_1,p_2,p_3)\in S_L$，$(q_1,q_2,q_3)\in S_R$

④ 记 $d=\min\{Dis(p_1,p_2,p_3), Dis(q_1,q_2,q_3)\}$

(3) Merge：① 在临界区查找不在同一子集中且构成的三角形周长小于 d 的三个点

② 若找到，则该三点即为所求，否则为 (p_1,p_2,p_3) 和 (q_1,q_2,q_3) 中距离最小者为周长最小的三角形顶点

其中临界区为 $x_1=m-\frac{d}{2}$ 与 $x_2=m+\frac{d}{2}$ 之间区域

输入：平面上 n 个点构成的集合 S

输出：能构成三角形且周长最短的三个点

```
Min-Circumference(S, n)
If n≤2 Then
    return 0
m ← Middle(S); // 计算x坐标中位数
用x=m把S划分为两个子集S_L和S_R
d1 ← Min-Circumference(S_L, n/2);
d2 ← Min-Circumference(S_R, n/2);
d ← min(d1, d2);
return Merge(S_L, S_R, d)
```

算法时间复杂性：$T(n)=2T(n/2)+O(n$

由Master定理可求得：

$$T(n)=O(n\log n)$$

第四章作业

4\. 输入：整数序列 $a_1, a_2 \cdots a_n$

输出：总代价最大的合并方案

(1) 设 $S[i,j]=a_i+a_{i+1}+\cdots+a_j$，$m[i,j]$ 表示合并 $a_i, a_{i+1}\cdots a_j$ 的最大代价

则 $m[i,j]=0\ (i=j)$;

$$m[i,j]=\max_{i\le k<j}\{m[i,k]+m[k+1,j]\}+S[i,j]\quad (i<j);$$

而 $S[i,j]=a_i,\ i=j$

$S[i,j]=S[i,j-1]+a_j,\ i<j$

(2) $D[i,j]=k$ 记录合并 $a_i, a_{i+1}\cdots a_j$ 的最后一次合并在 $a_i \sim a_k$ 和 $a_{k+1}\sim a_j$ 之间

①
```
MaxMergePrice(a1, a2, ··· an)
For i←1 to n Do
    m[i,i]←0, S[i,i]←ai;
For l←2 To n Do
    For i←1 To n-l+1 Do
        j←i+l-1;
        S[i,j] ← S[i,j-1]+aj;
        m[i,j] ← -∞;
        For k←i To j-1 Do
            q← m[i,k] + m[k+1,j];
            If q > m[i,j] Then
                m[i,j]←q, D[i,j]←k;
```

②
```
PrintScheme(D, i, j) // 输出方案
If i=j Then
    Print ai;
Else Print "(";
    PrintScheme(D, i, D[i,j]);
    Print "+";
    PrintScheme(D, D[i,j]+1, j);
    Print ")";
```

调用 PrintScheme(D, 1, n) 即可输出合并 $a_1, a_2 \cdots a_n$ 代价最大方案

算法时间复杂性：计算代价时间：$O(n^3)$ ，（l, i, k 三层循环，每层至多 $n-1$ 步）
输出方案时间：$O(n)$
故总时间复杂度为 $O(n^3)$

第五章作业

5. 贪心思想：每次选择所需时间最短的任务进行处理，直至完成全部任务

(1) 贪心选择性：$A = \{$~~$a_1 \cdots a_n$~~$1, 2, 3 \cdots n\}$ ~~时~~是 n 个任务的集合，若这 n 个任务已按所需时间排序，即 $a_1 \le a_2 \le \cdots \le a_n$，则存在一个最优解将任务1安排在第1个处理

证明：~~若~~设 $i_1, i_2, \cdots i_n$ 是一个最优解

① $i_1 = 1$，则成立

② $i_1 \ne 1$，设 $i_1 = r\ (r \ne 1)$，$i_k = 1\ (k \ne 1)$

则将 i_1 与 i_k 调换位置，则任务 r 等待时间由 $e_r = a_r$ 变为 $e_r = \sum_{j=2}^{k-1} a_{i_j} + a_r + a_1$

任务1等待时间由 $e_1 = \sum_{j=2}^{k-1} a_{i_j} + a_1 + a_r$ 变为 $e_1 = a_1$

且任务 $i_2 \sim i_{k-1}$ 等待时间均减少 $a_r - a_1$

则新序列 $i_k, i_2, i_3 \cdots i_{k-1}, i_1, i_{k+1}, \cdots i_n$ 也是一个最优解，首先处理任务1

(2) 优化子结构：设 $A = \{1, 2, 3 \cdots n\}$ 是这 n 个任务的集合，这 n 个任务已按所需时间排序，即 $a_1 \le a_2 \le \cdots \le a_n$，设 $i_1, i_2, \cdots i_n$ 是一个最优解，则 $A' = A - \{1\}$ 的优化解 S' 包含于 A 的优化解~~为~~ S 中

(3) 若任务 $1, 2, \cdots n$ 已按所需时间排好序，即 $a_1 \le a_2 \le \cdots \le a_n$，则序列 $1, 2, \cdots n$ 为问题最优解.

```
TaskScheduling (A, a1, a2 ... an)
1. n = Len(A);
2. For i <- 1 To n Do
3.     Res[i] = i;
4. return Res;
```

时间复杂性：如果 $a_1, a_2 \cdots a_n$ 已排序，则 $T(n) = \Theta(n)$

如果 $a_1, a_2 \cdots a_n$ 未排序，则 $T(n) = \Theta(n\log n) + \Theta(n) = \Theta(n\log n)$